University of Staffordshire

A Staged Architectural Framework for Behaviourally Grounded and Adversarially Robust Insider Threat Detection

Anisa Mkuwu

ORCID: 0009-0003-6599-0287

A thesis submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
at the University of Staffordshire

Primary Supervisor: Dr Vahid Heydari Fami Tafreshi

School of Digital Technology, Innovation and Business
University of Staffordshire
2026

Declaration

I declare that this thesis and the work presented in it are entirely my own original research. Where information has been derived from other sources, this has been appropriately acknowledged and referenced.

Signature:_____ Date:_____

To my family.

Abstract

Insider cyber threats remain among the most operationally damaging and conceptually challenging categories of organisational risk. Because harmful actions often occur within legitimate trust boundaries and mimic normative workflows, they remain difficult to distinguish from benign behavioural variance. While machine learning (ML) has become central to detection strategies, much prior research relies on relatively reductionist models and evaluation protocols that may not fully reflect operational complexities, such as temporal drift, domain heterogeneity, and adaptive evasion. Consequently, high benchmark scores do not always guarantee reliable performance in live, non-stationary environments.

This thesis proposes a behaviourally grounded and evaluation-disciplined approach to insider-threat detection. Rather than seeking a singular algorithmic solution, the research treats *representational stability*, *semantic alignment*, and *robustness* as incremental design objectives. Four architectural contributions are presented. First, a Hybrid Self-Organising Map–Artificial Neural Network (SOM–ANN) framework explores the separation of unsupervised perceptual learning from supervised inference to help stabilise high-variance telemetry. Second, a Modular Artificial Neural Network (MANN) architecture addresses domain-specific semantics through structured decomposition and controlled fusion, aiming to reduce cross-domain masking. Third, a multi-layered hierarchical framework models behaviour across different temporal granularities to capture longer-horizon trajectories. Fourth, the R-Guard architecture formalises behaviourally constrained adversarial models to evaluate resilience under perturbations that preserve behavioural plausibility.

Using the CERT insider-threat benchmark, the thesis applies chronology-preserving validation and leakage-aware preprocessing to ensure a defensible comparison between architectures. The results suggest that structured representations and disciplined evaluation can improve stability and reduce certain error modes relative to monolithic baselines. While not a complete solution to the insider problem, this work provides a principled foundation for detection systems that better integrate behavioural reasoning with architectural design and adversarial awareness.

Keywords: insider threat detection; behavioural machine learning; self-organising maps; modular neural networks; hierarchical modelling; adversarial machine learn-

ing; temporal evaluation; concept drift; enterprise logs; robustness.

Acknowledgements

I would like to express my sincere appreciation to my supervisors, Dr Vahid Heydari Fami Tafreshi, Professor Mohamed Sedky, and Professor Claude Chibelushi, for their invaluable guidance, intellectual challenge, and unwavering support throughout this doctoral journey. Their insight, encouragement, and critical feedback have been instrumental in shaping the direction, rigour, and depth of this research. I am deeply grateful for their patience and for the academic freedom they afforded me to explore, refine, and strengthen the ideas presented in this thesis.

My deepest gratitude goes to my wife, Margaret, whose steadfast support, understanding, and encouragement sustained me through the demanding seasons of this work. Her faith in me never wavered, even during the most challenging moments. I am also profoundly thankful to our three beautiful daughters, Rachel, Naomi, and Abigail. Their joy, curiosity, and boundless energy have been a constant reminder of purpose beyond research, and a source of strength and motivation.

I would also like to express my heartfelt gratitude to my parents, Stanley and Gwen Irwin. They have been the foundational driving forces behind my education, instilling in me the value of perseverance, discipline, and lifelong learning. Their sacrifices, belief, and guidance have shaped not only my academic journey but also the person I have become.

Finally, I acknowledge all those colleagues, friends, and collaborators who contributed in various ways to this work, whether through discussion, encouragement, or professional support. This thesis is the culmination of collective influence, mentorship, and love.

List of Publications

The following publications are based on the work presented in this thesis:

1. Evaluative Survey on Machine Learning Approaches for Detecting Insider Threats. [Chapter 2]
2. Hybrid Machine Learning Strategies for Robust Detection of Insider Cyber Threats. [Chapter 3]
3. Enhancing Insider Threat Detection with Modular Artificial Neural Networks (MANN). [Chapter 4]
4. A Multi-Layered Machine Learning Framework for Hierarchical Pattern Analysis and Adaptive Threat Behavior Recognition. [Chapter 5]
5. Guarding the Gates: Adversarial Robustness of Machine Learning-Based Insider Threat Detection Systems. [Chapter 6]

Contents

Abstract i

Acknowledgements iii

List of Publications iv

List of Algorithms xxviii

1 Introduction 1

1.1 Background . . . 1

1.1.1 Operationalising Behavioural Semantics . . . 2

1.2 Problem Statement . . . 4

1.2.1 Behavioural Dependencies and Spurious Correlations . . . 6

1.3 Research Questions . . . 8

1.4 Aim and Objectives . . . 12

1.4.1 Research Aim . . . 12

1.4.2 Research Objectives . . . 12

1.4.3 Scope and Boundaries . . . 14

1.4.4 Rationale for the Staged Architectural Progression . . . 14

1.5 Methodological Approach . . . 18

1.5.1 Research Design . . . 18

1.5.2 Dataset and Experiments . . . 19

1.5.3 Data Processing and Feature Design . . . 19

1.5.4 Evaluation Protocol . . . 19

1.5.5 Robustness and Adversarial Evaluation . . . 20

1.5.6 Reproducibility and Reporting . . . 20

1.5.7 Ethical and Governance Considerations . . . 20

1.6 Contributions . . . 21

1.6.1 Technical and Architectural Inquiry . . . 21

1.6.2 Methodological Honesty and Transparency . . . 22

1.6.3 The Human-Centric and Governance Perspective . . . 22

1.7 Structure . . . 22

2 Literature Review 24

2.1 Overview . . . 24

2.2 Introduction . . . 25
2.3 The Insider Threat Problem . . . 26
2.4 Insider Threat Lifecycle . . . 27
2.4.1 Phase 1: Reconnaissance . . . 27
2.4.2 Phase 2: Misuse . . . 28
2.4.3 Phase 3: Escalation . . . 28
2.4.4 Linking the Lifecycle to Model Design . . . 29
2.4.5 Summary . . . 29
2.4.6 Contemporary Insider-Threat Case Illustrations . . . 30
2.4.6.1 Case 1: High-Value Intellectual Property Exfiltration 30
2.4.6.2 Case 2: Disgruntled Employees . . . 30
2.4.6.3 Case 3: Insider–Outsider Collusion . . . 31
2.4.6.4 Case 4: Sabotage . . . 31
2.4.6.5 Synthesis: Implications for Behavioural Modelling . . 31
2.4.7 The Nature of Insider Behaviour . . . 32
2.4.8 Epistemological Positioning . . . 35
2.5 Behavioural Foundations . . . 35
2.5.1 Human Behaviour and Insider Risk . . . 36
2.5.2 Role and Privilege Semantics . . . 37
2.5.3 Workload Rhythms . . . 38
2.5.4 Socio-Technical Interactions . . . 38
2.5.5 Behavioural Semantics and ML Limitations . . . 38
2.5.6 Behavioural Requirements for ML Architectures . . . 39
2.6 Methodological Limitations . . . 40
2.6.1 Dataset Realism . . . 40
2.6.2 Feature Engineering Limits . . . 41
2.6.3 Temporal and Context Misalignment . . . 42
2.6.4 Evaluation Flaws . . . 42
2.6.5 Reproducibility Issues . . . 42
2.6.6 Identity–Behaviour Confounding . . . 43
2.6.7 Data Leakage . . . 43
2.6.8 Concept Drift and Behavioural Evolution . . . 44
2.6.9 Methodological Bottlenecks . . . 45
2.7 Machine Learning Foundations . . . 47
2.7.1 AI and Machine Learning . . . 47
2.7.2 Where SOM and ANN Sit Within the ML Hierarchy . . . 48
2.7.3 Algorithm Families . . . 49
2.7.4 Common ML Workflow . . . 50
2.7.5 Positioning the Architecture . . . 50
2.8 Hybrid, Modular and Hierarchical ML . . . 51

2.8.1 Hybrid Architectures . . . 52
2.8.1.1 Lack of Joint Optimisation . . . 52
2.8.1.2 Limitations of SOM Topology and Stability . . . 53
2.8.1.3 Behavioural Shallowing in Feature Pipelines . . . 53
2.8.1.4 Pipeline Fragility and Leakage . . . 53
2.8.2 Modular Architectures . . . 54
2.8.2.1 Isolation and Missing Cross-Domain Reasoning . . . 54
2.8.2.2 Weak Semantic Fusion . . . 54
2.8.2.3 Behavioural Bias and Module Overfitting . . . 55
2.8.2.4 Lack of Behavioural Hierarchy . . . 55
2.8.3 Hierarchical Architectures . . . 55
2.8.3.1 Hierarchical Sequence Models . . . 55
2.8.3.2 Behavioural-State Modelling Constraints . . . 55
2.8.3.3 Hierarchical Representation . . . 56
2.8.3.4 Missing Semantic Constraints and Privilege Context 56
2.8.4 Architectural Fragility . . . 56
2.8.5 Synthesis and Contributions . . . 56
2.9 Relational Modelling and GNN Misalignment . . . 57
2.9.1 The Appeal of Relational Modelling . . . 58
2.9.2 Graphs and Behavioural Semantics . . . 58
2.9.3 Graph Construction and Structural Bias . . . 59
2.9.4 Temporal Dynamics . . . 60
2.9.5 Privilege and Role Semantics . . . 60
2.9.6 Message Passing . . . 61
2.9.7 Graph Sparsity, Density and Structural Instability . . . 61
2.9.8 Adversarial Vulnerabilities . . . 61
2.9.9 Limitations of Structure-Only Reasoning . . . 62
2.9.10 Insufficiency of Graphs . . . 62
2.9.11 Summary of Relational–Behavioural Misalignment . . . 62
2.10 Adversarial Modelling of Insider Threats . . . 63
2.10.1 Limits of Classical Adversarial ML . . . 63
2.10.2 Behaviourally Constrained Perturbation Operators . . . 64
2.10.2.1 Temporal Scheduling Operator $\mathcal{S}_\rho$ . . . 64
2.10.2.2 Gradual Drift Operator Δ_δ . . . 64
2.10.2.3 Sequence Reordering Operator $\mathcal{P}_\sigma$ . . . 65
2.10.2.4 Privilege-Aligned Substitution Operator $\mathcal{S}_{priv}$ . . . 65
2.10.3 Composite Behavioural Adversaries . . . 65
2.10.3.1 Gradual Shaper (Drift Adversary) . . . 65
2.10.3.2 Timing Manipulator . . . 66
2.10.3.3 Privilege Aligner . . . 66

2.10.3.4 Sequence Rewriter . . . 66
2.10.4 Behavioural Feasibility Manifold . . . 66
2.10.5 Architectural Vulnerabilities . . . 67
2.10.6 Adversarial Taxonomy . . . 67
2.10.7 Summary . . . 68
2.11 Synthesis of Unresolved Gaps . . . 68
2.11.1 Behavioural Gaps . . . 68
2.11.2 Temporal Gaps . . . 68
2.11.3 Conceptual Gaps in Relational Modelling . . . 69
2.11.4 Methodological Gaps . . . 69
2.11.5 Adversarial Gaps . . . 69
2.11.6 Architectural Gaps . . . 69
2.11.7 Integrated Research Gap . . . 69
2.11.7.1 Rationale for the Staged Architecture . . . 71
2.11.7.2 Why the Sequence Is Non-Arbitrary . . . 73
2.12 ANN as a Baseline . . . 75
2.12.1 ANNs for Insider Threat Detection . . . 75
2.12.2 ANN Families and Relevance . . . 76
2.12.2.1 Feedforward (Dense) Neural Networks . . . 76
2.12.2.2 Convolutional Neural Networks . . . 77
2.12.2.3 Recurrent Neural Networks (RNNs), LSTMs, and GRUs . . . 77
2.12.2.4 Transformer-Based and Attention-Based . . . 78
2.12.2.5 Hybrid and Structured ANN Extensions . . . 78
2.12.3 ANN and CERT Implementation . . . 78
2.12.3.1 Data Integration and Behavioural Feature Construction . . . 80
2.12.3.2 Preprocessing, Normalisation, and Leakage Control . 80
2.12.3.3 Class Imbalance and Label Scarcity . . . 80
2.12.4 Optimising, Tuning, and Evaluating ANN . . . 81
2.12.4.1 Architecture and Optimisation Tuning . . . 81
2.12.4.2 ANN Evaluation Metrics . . . 82
2.12.4.3 Baseline Reproducibility and Reporting Discipline . . 82
2.12.5 Case Study Baseline . . . 83
2.12.5.1 Case Study Objective and Scope . . . 83
2.12.5.2 Baseline Implementation Outline . . . 83
2.12.5.3 Expected Strengths and Limitations . . . 84
2.12.6 Implications . . . 84
2.12.7 Synthesis and Transition . . . 85
2.13 Insider-Threat Datasets . . . 86

2.13.1 The Role of Datasets . . . 86
2.13.2 Types of Datasets . . . 87
2.13.2.1 Synthetic and Simulated Datasets . . . 87
2.13.2.2 Real Organisational and Proprietary Datasets . . . 87
2.13.2.3 Public Benchmarks . . . 87
2.13.3 Criteria for Evaluating Insider-Threat Datasets . . . 88
2.13.3.1 Behavioural Realism and Contextual Plausibility . . . 88
2.13.3.2 Temporal Continuity and Sequential Integrity . . . 88
2.13.3.3 Multi-Modal / Multi-Log Coverage . . . 88
2.13.3.4 Label Availability, Granularity, and Class Imbalance 89
2.13.3.5 Accessibility and Reproducibility . . . 89
2.13.3.6 Ethical and Governance Suitability . . . 89
2.13.4 Critical Review of the CERT Dataset . . . 89
2.13.4.1 Strengths of CERT for Comparative Research . . . 91
2.13.4.2 Synthetic Artefacts and Architectural Validity . . . 92
2.13.4.3 Interpretive Implications . . . 95
2.13.5 Rationale for Selecting CERT . . . 95
2.13.6 Validity Boundaries . . . 96
2.13.7 External Validity and Organisational Adaptation . . . 96
2.13.8 Section Summary . . . 100
2.14 Evaluation Discipline and Research Design . . . 102
2.14.1 Research Design Philosophy . . . 103
2.14.2 Temporal Integrity and Leakage Prevention . . . 103
2.14.3 Identity–Behaviour Confounding . . . 103
2.14.4 Metric Selection Under Class Imbalance . . . 104
2.14.5 Adversarial Robustness . . . 104
2.14.6 Construct Validity . . . 104
2.14.7 Internal Validity . . . 105
2.14.8 External Validity . . . 105
2.14.9 Statistical Conclusion Validity . . . 105
2.14.10 Reproducibility and Artefact Transparency . . . 106
2.14.11 Summary of Validity Considerations . . . 106
2.15 Ethics . . . 106
2.15.1 Privacy and Behavioural Monitoring . . . 106
2.15.2 Transparency and Human Oversight . . . 107
2.15.3 Fairness, Role Sensitivity and Bias Risk . . . 107
2.15.4 Psychological and Organisational Impact . . . 108
2.15.5 Governance and Deployment Boundaries . . . 108
2.15.6 Ethical Implications . . . 109
2.16 Chapter Summary . . . 109

3 Hybrid Framework **111**
3.1 Overview . . . 111
3.2 Introduction . . . 112
3.3 Formulation of the SOMANN . . . 114
3.3.1 Definition of the Self-Organising Map . . . 114
3.3.2 Behavioural Interpretation . . . 115
3.3.3 SOM-Derived Representation . . . 116
3.3.4 ANN Discriminative Layer . . . 117
3.3.5 Integrated SOMANN Pipeline . . . 117
3.3.6 Theoretical Justification of SOMANN . . . 118
3.3.7 Plasticity, Stability, and Convergence . . . 119
3.3.8 Topology Preservation Metrics . . . 119
3.3.9 Variance Smoothing . . . 120
3.3.10 Theoretical Illustrations . . . 121
3.4 Methodology . . . 121
3.4.1 Experimental Setup and Data Partitioning . . . 122
3.4.2 Data Preprocessing and Feature Construction . . . 122
3.4.3 SOM Training Procedure . . . 123
3.4.4 Construction of SOM-Derived Representations . . . 124
3.4.5 ANN Training and Integration . . . 124
3.4.6 Baseline Comparison and Evaluation Protocol . . . 124
3.4.6.1 Unified Baseline and Cross-Chapter Protocol . . . 125
3.5 Experimental Evaluation . . . 128
3.5.1 Baseline Comparison: SOMANN versus ANN . . . 128
3.5.2 Repeated-Run Statistical Validation of SOMANN . . . 129
3.5.3 Identity Dependence and Generalisation . . . 130
3.5.3.1 Identity Dependence of the SOM Representation . . . 130
3.5.3.2 Generalisation to Previously Unseen Users . . . 131
3.5.3.3 Combined Interpretation . . . 132
3.5.4 Ablations and Sensitivity Analysis: SOM–ANN . . . 133
3.5.4.1 SOM Grid Size Sensitivity . . . 133
3.5.4.2 Neighbourhood Function and Learning Rate Schedule 133
3.5.4.3 Feature Smoothing Window Length Impact . . . 133
3.5.4.4 SOM Stabilisation vs. ANN-Only Comparison . . . 134
3.5.5 Error Profile and Behavioural Stability . . . 134
3.5.6 Impact of SOM-Derived Representation . . . 135
3.5.7 Diagnostic Visualisations and Supplementary Analysis . . . 135
3.5.8 Summary of Experimental Findings . . . 135
3.6 Discussion . . . 136
3.6.1 Behavioural Interpretation of Performance Gains . . . 136

3.6.2 Representation Stability and Variance Reduction 136
3.6.3 False-Positive Reduction and Operational Value 137
3.6.4 Interpretability and Diagnostic Value 137
3.6.5 Limitations of the SOMANN Approach 138
3.6.6 Implications for Modular and Hierarchical Extensions 138
3.6.7 Summary 139
3.7 Summary 139

4 Modular Artificial Neural Network 141
4.1 Chapter Overview 141
4.2 Introduction 142
4.3 Behavioural Domain Decomposition 144
4.4 MANN Architecture 146
4.4.1 Architectural Overview 146
4.4.2 Domain-Specific Encoders 147
4.4.3 Fusion Mechanisms 147
4.4.4 Decision Layer and Output 149
4.4.5 Relationship to Prior Architectures 149
4.4.6 Architectural Scope and Design Boundaries 149
4.5 Methodology 150
4.5.1 Dataset and Feature Construction 150
4.5.2 Temporal Partitioning and Evaluation Protocol 150
4.5.2.1 Temporal Shift and Leakage Validation 151
4.5.3 Training Procedure 153
4.5.4 Baseline Models and Comparative Setup 154
4.5.5 Evaluation Metrics 154
4.5.6 Validity and Scope Considerations 154
4.6 Experimental Evaluation 155
4.6.1 Primary Comparative Performance 155
4.6.1.1 Reconciled Comparative Results 155
4.6.2 Repeated-Run Statistical Validation of MANN 157
4.6.3 Domain-Level Contribution Analysis 158
4.6.4 Ablations and Sensitivity Analysis 158
4.6.4.1 Module Contribution Analysis 158
4.6.4.2 Fusion Operator Sensitivity 159
4.6.4.3 Cross-Domain Conflict Cases (Where Modules Disagree) 159
4.6.4.4 Compute Cost vs. Performance Trade-Off 159
4.6.5 Error Profile and Stability 159
4.6.6 Fusion and Ablation Effects 160
4.6.7 Summary of Experimental Findings 160

4.7 Theoretical Foundations of Modular Neural Architectures 160
4.7.1 Modularity in Learning Systems 161
4.7.2 Bias–Variance Perspective on Modular Learning 161
4.7.3 Cognitive and Organisational Parallels 162
4.7.4 Behavioural Non-Stationarity 162
4.7.5 Limits of Static Modularity 163
4.7.6 Implications for the Thesis Architecture Progression 163
4.8 Discussion 164
4.8.1 What Modular Decomposition Adds Beyond Hybrid Stabilisation 164
4.8.2 Error Characteristics and Behavioural Plausibility 164
4.8.3 Stability and Reproducibility Considerations 165
4.8.4 Fusion Mechanisms and Architectural Trade-offs 165
4.8.5 Validity Boundaries and Interpretation Limits 166
4.8.6 Implications for Chapter 5 166
4.9 Summary 166

5 A Multi-Layered Framework 168
5.1 Overview 168
5.2 Introduction 168
5.2.1 Temporal Blind-Spot Failure Mode 169
5.3 Behavioural Hierarchy in Insider Activity 172
5.3.1 Hierarchy as an Organising Principle 172
5.3.2 Conceptual Hierarchy and Implementation Pipeline 173
5.3.2.1 Three-Tier Conceptual Behavioural Hierarchy 174
5.3.2.2 Five-Stage Computational Pipeline 174
5.3.2.3 Mapping Between the Two Views 175
5.3.3 Relationship to Modularity 176
5.3.4 Behavioural Progression Across the Hierarchy 176
5.3.5 Implications for Evaluation 177
5.4 Formal Hierarchical Framework 177
5.4.1 Notation and Multi-Log Behavioural Observations 177
5.4.2 Hierarchical Representation Levels 178
5.4.3 Hierarchical Fusion and Risk Inference 179
5.4.4 Training Objective and Class-Imbalance Handling 180
5.4.5 Design Rationale and Complexity Considerations 180
5.4.6 Scope and Validity Boundaries 180
5.5 Methodology and Implementation 181
5.5.1 Dataset Configuration and Preprocessing 181
5.5.2 Chronology-Preserving Data Partitioning 181
5.5.3 Hierarchical Model Instantiation 182

5.5.4 Training Procedure and Optimisation . . . 182
5.5.5 Evaluation Metrics and Reporting . . . 182
5.5.6 Implementation Environment and Reproducibility . . . 183
5.5.7 Ethical and Methodological Considerations . . . 183
5.6 Experimental Evaluation . . . 183
5.6.1 Analysis of Performance Benchmarks . . . 183
5.6.2 Operational Drift Metrics . . . 185
5.6.2.1 Model Stability Index . . . 186
5.6.2.2 Concept Drift Rate . . . 186
5.6.2.3 Epoch-Based Retraining Efficiency . . . 187
5.6.2.4 Interpretive Scope . . . 188
5.6.3 Repeated-Run Statistical Validation of MLML . . . 189
5.6.4 Interpretation of Retraining Efficiency . . . 190
5.6.5 Evaluation Protocol and Baselines . . . 191
5.6.6 Aggregate Performance Comparison . . . 191
5.6.7 Temporal Error Analysis . . . 192
5.6.8 Ablation and Sensitivity Analysis . . . 192
5.6.8.1 Ablation Studies: Contribution of Hierarchical Levels 192
5.6.8.2 Sensitivity Analysis: Temporal Context Lengths (τ_2, τ_3) 192
5.6.8.3 Conditions Under Which Hierarchy Provides Limited Benefit . . . 193
5.6.8.4 Smoothing Effects on Detection Performance . . . 193
5.6.9 Comparative Interpretation Across Chapters . . . 194
5.6.10 Limitations of the Experimental Findings . . . 194
5.7 Discussion . . . 195
5.7.1 What Hierarchical Abstraction Contributes . . . 195
5.7.2 Hierarchy Versus Modularity . . . 195
5.7.3 Behavioural Plausibility and Operational Relevance . . . 196
5.7.4 Limits of Hierarchical Modelling . . . 196
5.7.5 Positioning Within the Thesis Architecture . . . 196
5.8 Summary . . . 197

6 Adversarial Machine Learning 198
6.1 Overview . . . 198
6.2 Introduction . . . 199
6.3 Related Work and Fundamental Limitations . . . 200
6.3.1 Standard AML Threat Models . . . 200
6.3.2 Mismatch with Insider-Threat Contexts . . . 201
6.3.3 Behavioural Constraints . . . 201
6.3.4 Implications for Robustness Evaluation . . . 202
6.4 Adversarial Threat Model . . . 202

6.4.1 Adversarial Objectives . . . 202
6.4.2 Adversarial Capabilities . . . 203
6.4.3 Behavioural and Organisational Constraints . . . 203
6.4.4 Temporal Structure and Sequential Adaptation . . . 204
6.4.5 Knowledge Assumptions . . . 204
6.4.6 Scope and Limitations of the Threat Model . . . 204
6.5 Adversarial Perturbation Operators . . . 204
6.5.1 Perturbation Operators as Behavioural Transformations . . . 205
6.5.2 Taxonomy of Behavioural Perturbation Operators . . . 205
6.5.3 Adaptive Insider Behaviour . . . 207
6.5.4 Constraint Space for Behavioural Perturbations . . . 207
6.5.5 Mapping Perturbation Operators . . . 208
6.5.6 Role Within the Thesis Architecture . . . 208
6.5.6.1 Perturbation Taxonomy Reconciliation . . . 208
6.5.6.2 Composite Perturbation Class . . . 210
6.5.6.3 Causal Validity of Sequence Permutation . . . 210
6.5.6.4 Relationship to IAA-Sequential . . . 210
6.5.7 Formal Definition of IAA-Sequential . . . 211
6.5.7.1 Sequential Evaluation Procedure . . . 212
6.6 Robust Detection Architecture . . . 213
6.6.1 Formal Specification of the Denoising Autoencoder . . . 214
6.6.2 Design Principles and Architectural Rationale . . . 215
6.6.3 Architectural Overview . . . 216
6.6.4 Mitigation of Behavioural Perturbation Operators . . . 216
6.6.5 Relationship to Prior Thesis Contributions . . . 217
6.6.6 Scope, Assumptions, and Limitations . . . 217
6.6.7 Interpretation and Governance Considerations . . . 218
6.7 Experimental Evaluation . . . 218
6.7.1 Repeated-Run Statistical Validation of R-Guard . . . 221
6.7.2 Experimental Design Rationale . . . 223
6.7.3 Datasets and Baseline Configurations . . . 223
6.7.4 Adversarial Behaviour Generation . . . 224
6.7.5 Evaluation Metrics and Robustness Interpretation . . . 224
6.7.6 Behaviour Under Individual Perturbation Operators . . . 224
6.7.7 Clean Versus Adversarial Performance Comparison . . . 225
6.7.8 Robustness–Utility Trade-offs . . . 226
6.7.9 Ablations and Sensitivity Analysis . . . 226
6.7.9.1 Robustness–Utility Curve Interpretation . . . 226
6.7.9.2 Operator Composition Order Effects . . . 226
6.7.9.3 Qualitative Example Trajectory . . . 227

6.7.10 Failure Modes and Boundary Conditions . . . 227
6.7.11 Summary of Experimental Findings . . . 227
6.8 Discussion and Synthesis . . . 227
6.8.1 From Predictive Performance to Behavioural Robustness . . . 228
6.8.2 Operator-Specific Robustness Insights . . . 228
6.8.3 Robustness–Utility Trade-offs and Operational Interpretation . 228
6.8.4 Relationship to the Thesis Architecture Progression . . . 229
6.8.5 Failure Modes and Interpretation Boundaries . . . 229
6.8.6 Implications for Research and Practice . . . 229
6.8.7 Summary . . . 230
6.9 Comparative Synthesis of Contributions . . . 230
6.9.1 Overview of Models and Contributions . . . 230
6.9.2 Comparing Model Performance Across Architectures . . . 231
6.9.2.1 Evaluation Framework . . . 231
6.9.2.2 Model Comparison: Clean vs Adversarial Performance 231
6.9.3 Trade-offs and Architectural Strengths . . . 232
6.9.3.1 Stability vs. Adaptability . . . 232
6.9.3.2 Complexity vs. Interpretability . . . 232
6.9.4 Failure Modes and Limitations . . . 233
6.9.4.1 Insider Adaptation and Failure Points . . . 233
6.9.4.2 Resource Efficiency . . . 233
6.9.5 Reflection . . . 233
6.10 Summary . . . 234

7 Conclusion and Future Directions 235
7.1 Research Context . . . 235
7.2 Summary of Contributions . . . 236
7.3 Central Thesis Claim . . . 237
7.4 Cross-Chapter Synthesis and Unifying Principle . . . 238
7.4.1 Empirical Failure Modes Motivating the Progression . . . 239
7.4.2 Cumulative Rather Than Independent Contributions . . . 241
7.4.3 Theoretical Implication . . . 242
7.5 Theoretical Implications . . . 242
7.6 Practical and Operational Implications . . . 243
7.7 Limitations of the Research . . . 244
7.8 Future Research Directions . . . 246
7.8.1 Automated and Semi-Automated Invariant Learning . . . 246
7.8.2 Explainable AI and Analyst-Centred Decision Support . . . 246
7.8.3 Cross-Organisational Transfer and Adaptation . . . 246
7.8.4 Expanded Adversarial Threat Models . . . 246
7.8.5 Longitudinal and Real-World Deployment Studies . . . 246

7.8.6 Privacy, Governance, and Responsible Deployment 247
7.9 Concluding Remarks 247

A Supplementary Figures **254**
A.1 Artefact: SOMANN Supplementary Figures 254
A.1.1 Baseline Comparison: SOMANN versus ANN 259
A.1.2 Representation Effect and Architectural Stability 259
A.1.3 Error Profile and Behavioural Stability 260
A.1.4 Sensitivity to Gradual Escalation 260
A.2 Artefact: MANN Supplementary Figures 260
A.2.1 Cross-Validation and Representational Stability 263
A.2.2 Mitigating Representational Entanglement 263
A.3 Hierarchical Supplementary Figures 268
A.3.1 Deconstructing the Incremental Value of Architectural Depth . 270
A.3.2 Operational Significance and Comparative Strength 270
A.3.3 Transitioning Toward Hierarchical and Robust Detection . . . 271
A.3.4 Quantifying Stability in Evolving Environments 271
A.3.5 Operational Sustainability and Retraining Efficiency 272
A.4 Artefact: R-Guard Supplementary Figures 272
A.4.1 Discussion of Adversarial Resilience 273
A.4.2 The Denoising Autoencoder as a Strategic Filter 273
A.4.3 Consistency and the Trade-off of Robustness 273

B Methodological Decision Framework **281**

C Algorithms Adapted **283**

D Implementation Artefacts **288**
D.1 Artefact—SOMANN Implementation 288
D.2 Artefact—MANN Implementation 305
D.3 Artefact—Multi-Layered Machine Learning (MLML) Implementation 320
D.4 Artefact—Guarding the Gates Implementation 338
D.5 Appendix D Artefact Summary 360

E List of Abbreviations **362**

F Terms and Terminology **364**

List of Figures

2.1 Reported average annual organisational cost associated with insider threat and insider-risk incidents across selected Ponemon Institute reporting years. Reported values are $11.45 million in 2020, $15.38 million in 2022, $16.2 million in 2023, and $19.5 million in 2026 [5], [6], [7], [29]. *Source: Author's visualisation based on the cited industry reports.* 32
2.2 Conceptual taxonomy of insider threats, distinguishing malicious, negligent and compromised insiders across behavioural, motivational and operational dimensions. *Source: Author's own conceptualisation based on the literature reviewed in this chapter.* 34
2.3 Overview of core behavioural foundations relevant to insider-threat modelling: role and privilege semantics, workload rhythms and temporal patterns, socio-technical interactions, and behavioural semantics. *Source: Author's conceptual synthesis of the literature reviewed in this chapter.* 36
2.4 Conceptual behavioural manifold illustrating how privilege assignments and role semantics shape the interpretation of behavioural deviation. 37
2.5 Socio-technical human–machine interaction loop illustrating how human intent, organisational context and system responses shape observable behaviour. 39
2.6 Positioning of SOM and ANN within the AI/ML hierarchy. SOMs support unsupervised topology-preserving behavioural structuring, while ANNs provide a supervised deep-learning baseline for non-linear discrimination. Their hybridisation motivates the SOM–ANN contribution and supports subsequent modular, hierarchical, and robust extensions. 48
2.7 Evolution of insider-threat detection architectures from monolithic models toward *hybrid* pipelines (unsupervised representation learning combined with supervised discrimination), *modular* multi-log designs (domain-specific models with fusion), and *hierarchical* frameworks (multi-level behavioural abstraction across time and context). 52

2.8 Behaviourally constrained insider adversary space. Each adversary operates within legitimate privilege, role semantics, and operational feasibility while manipulating different behavioural dimensions (temporal dynamics, privilege alignment, and event structure). Unlike overt attacks, these strategies preserve local plausibility and evade detectors that rely on magnitude-based anomalies or static behavioural baselines. 65
2.9 High-level roadmap linking behavioural foundations, methodological limitations and existing architectures to the unresolved gaps addressed by the SOMANN hybrid (Chapter 3), modular architecture (Chapter 4), multi-layered behavioural framework (Chapter 5) and adversarially robust R-Guard architecture (Chapter 6). 70
2.10 Synthesis of unresolved gaps motivating the architectural contributions of Chapters 3–6. 70
2.11 Thesis architecture progression from a conventional ANN baseline to progressively structured and robust insider-threat detection frameworks. Each stage addresses a specific limitation of the preceding modelling configuration, moving from general non-linear behavioural classification toward modularity, hierarchical behavioural abstraction, and adversarial robustness. 85

3.1 Conceptual SOMANN hybrid detection pipeline. The SOM performs unsupervised topology-preserving behavioural organisation to form a structured knowledge space. A knowledge-mapping interface derives topology-informed descriptors that are then consumed by an ANN for supervised risk inference and calibrated threat probability estimation. 113
3.2 Conceptual illustration of behavioural manifold compression in SOMANN. High-dimensional behavioural observations (constructed from multi-log features) are organised onto a two-dimensional SOM lattice that preserves neighbourhood relations. This topology-preserving compression stabilises representation geometry prior to supervised discrimination, supporting trajectory-aware interpretation and reducing sensitivity to noise-dominated variance. 121
3.3 Variance smoothing and behavioural regime structure induced by SOM organisation. The distribution of SOM-derived deviation measures (e.g., MID/quantisation-based distance) reveals clustered low-distance regions corresponding to routine behavioural regimes and higher-distance regions associated with atypical or transitional activity. These topology-informed descriptors form stability-oriented inputs for ANN risk inference in SOMANN. 122

4.1 Architectural progression from stabilised unified representation to modular behavioural decomposition. **Phase 1 (Chapter 3)** employs SOM-based perceptual representation learning to stabilise behavioural variance before supervised inference, producing a unified latent embedding. Empirical analysis reveals a key limitation: unified embeddings can dilute domain-specific semantics and mask subtle anomalies. **Phase 2 (Chapter 4)** addresses this limitation through modular decomposition, where behavioural domains are encoded independently and integrated via a controlled fusion layer to produce a unified risk assessment. This modular foundation enables subsequent hierarchical modelling (Chapter 5) and adversarial robustness mechanisms (Chapter 6). 143

4.2 MANN architecture illustrating behavioural domain decomposition and controlled fusion. Each behavioural domain (authentication, file system, communication, and device/network activity) is processed by a dedicated domain encoder to produce domain-aligned embeddings. These embeddings are integrated through a configurable fusion layer, supporting either *concatenation-based fusion* (static integration) or *adaptive/attention-based fusion* (dynamic domain weighting), as analysed in the ablation studies in Section 4.6.4. By learning domain-specific representations prior to fusion, the architecture mitigates semantic dilution and reduces cross-domain masking from high-volume benign activity, improving robustness and false-positive control. . . . 148

5.1 Computational pipeline of the Multi-Layered Modular Learning (MLML) framework. The complete implementation comprises five processing stages, within which three behavioural abstraction tiers represent progressively broader behavioural context. Stage 1 performs telemetry ingestion and normalisation; Stages 2–4 implement session-level, task/domain-level, and trajectory-level behavioural representation, respectively; and Stage 5 produces the final prediction and auxiliary outputs. The five-stage computational pipeline therefore contains, rather than competes with, the three-tier conceptual behavioural hierarchy. 173

5.2 Temporal resilience and adaptation behaviour of the MLML framework under the evaluated drift condition. The framework retains approximately 89% of its reference ROC-AUC at the final temporal block (MSI = 0.89), while the normalised latent-distribution displacement is CDR = 0.25. Fine-tuning requires 83% fewer training epochs than the corresponding from-scratch procedure (RE = 0.83), and the adapted model achieves a post-drift F1-score of 0.92. MSI, CDR, and RE are operational measures defined specifically for the evaluation protocol used in this thesis. 184
5.3 Incremental performance across the architectural layers of the MLML framework (L_1 through L_5). The transition to Level 3 (Parallel Domain Modules) is associated with the largest observed increase in F1-score (+5%), while Levels 4 and 5 provide further incremental improvements. The complete configuration achieves an AUC of 0.97 under the reported evaluation conditions. *Source: Author's own experimental results.* 184
5.4 Comparative performance of the MLML framework and the evaluated baselines on the CERT dataset. Under the reported evaluation protocol, MLML achieves the highest values across the displayed metrics, including Accuracy and ROC-AUC values of 0.97. The comparison provides aggregate evidence of improved predictive performance under the evaluated benchmark conditions. *Source: Author's own experimental results.* 185

6.1 Comparative robust accuracy of baseline models (RF, LSTM, Transformer) versus the $\mathcal{R}$-Guard framework under IAA_Sequential attacks on the CERT dataset. While traditional architectures suffer significant performance degradation when exposed to sequential behavioral manipulation, $\mathcal{R}$-Guard maintains a high robust accuracy of 89.0%. This demonstrates the framework's improvement ability to preserve detection integrity even when an adversary intentionally masks malicious intent within legitimate-looking activity streams. 219
6.2 Ablation study of $\mathcal{R}$-Guard components illustrating the incremental gain in robust accuracy. The baseline model's resilience (65%) is substantially enhanced by the integration of the Denoising Autoencoder (+DAE), which provides the single largest boost in protection. The addition of consistency regularization (+Reg) further stabilizes the model, culminating in the full $\mathcal{R}$-Guard's peak robustness. This progression confirms that a multi-layered defense strategy is essential for neutralizing sophisticated, time-evolving insider threats. 220

A.1 Legend for SOM mean inter-neuron distance (MID) values used to interpret behavioural stability on the map. Lower MID indicates dense, stable neighbourhood structure consistent with normal behaviour; intermediate MID reflects transitional regions where behaviour shifts between neighbourhoods; higher MID corresponds to sparse or highly separated neighbourhoods associated with anomalous or weakly supported behavioural patterns. 254
A.2 Continuous SOM U-matrix showing the behavioural topology of the CERT dataset. Lighter areas correspond to higher mean inter-neuron distance (MID), forming anomaly-prone boundaries, while darker regions represent cohesive, stable activity clusters. Note: The threshold values shown reflect intermediate valley estimates for illustration. All final analyses use the validated thresholds $t_{low} = 0.05$ and $t_{high} = 0.71$. 255
A.3 SOM U-matrix with KDE-derived MID behavioural zones overlaid. Low MID regions denote stable/normal behaviour; intermediate MID regions represent transitional drift; and high MID regions correspond to anomaly-prone activity. Zones are derived from KDE valley thresholds at $t_{low} = 0.05$ and $t_{high} = 0.71$. Note: The threshold values shown reflect intermediate valley estimates for illustration. All final analyses use the validated thresholds $t_{low} = 0.05$ and $t_{high} = 0.71$. . . 255
A.4 High-MID neurons selected as anomaly-bearing units for SOM→ANN supervised learning. Samples mapped to these neurons form the derived dependent variable used to train the ANN inference layer in the SOMANN hybrid framework. Note: The threshold values shown reflect intermediate valley estimates for illustration. All final analyses use the validated thresholds $t_{low} = 0.05$ and $t_{high} = 0.71$. 256
A.5 SOM dynamics output map illustrating normalised U-matrix MID values across neurons, with highlighted high-MID regions. These regions correspond to behavioural boundary zones where atypical or transitional activity concentrates, supporting anomaly localisation within the SOM topology. 256
A.6 Correlation analysis between SOM-derived mean inter-neuron distance (MID) and ANN-predicted malicious probability (per neuron mean). The weak and statistically insignificant correlation indicates that SOM structural deviation and ANN inference capture complementary information rather than redundant signals, supporting the hybrid SOMANN design. 257

A.7 Alternative SOM dynamics output map from an independent training run, showing consistent emergence of high-MID regions despite stochastic initialisation. This illustrates the robustness of behavioural boundary formation in the SOM representation. 257
A.8 Extraction of high-MID regions from the SOM U-matrix used to derive SOM-informed features for ANN training. Highlighted neurons represent boundary and sparse regions where behavioural deviation is most pronounced, forming a principled bridge between unsupervised structure discovery and supervised risk inference. 258
A.9 Annotated SOM U-matrix with labelled low-, intermediate-, and high-MID regions. This detailed diagnostic view supports analyst-oriented interpretation by explicitly mapping behavioural regimes and anomaly-prone boundary areas within the learned topology. 258
A.10 Standalone performance summary for the MANN implementation (selected metrics). Included as supporting material for the evaluation reported in Section 4.6. 261
A.11 Supplementary visual comparison of MANN and ANN performance across selected metrics. The figure provides an illustrative summary consistent with Chapter 4 results; interpretation should emphasise PR-oriented metrics under class imbalance. 262
A.12 Supplementary computational-time comparison between ANN and MANN implementations (illustrative). Timing depends on hardware, batching, and implementation choices and is provided as a transparency diagnostic. 262
A.13 Training dynamics diagnostic: learning curves for ANN and MANN across epochs. This figure supports discussion of convergence and potential overfitting/underfitting patterns under the chosen protocol. 263
A.14 Illustrative feature-importance summary for insider-threat detection under the MANN model. Feature importance should be interpreted cautiously due to feature correlations and potential dataset-specific artefacts. 265
A.15 SHAP-style summary bar chart (mean absolute contribution) for the top features. Included as supplementary interpretability evidence, not as proof of causal influence. 265
A.16 SHAP-style beeswarm visualisation showing the distribution of feature contributions across instances (illustrative). This supports interpretation of heterogeneity in feature influence under behavioural data. 266

A.17 Force-plot-style contribution chart for a single instance (illustrative), showing positive and negative feature contributions relative to a baseline. Included as supplementary instance-level explanation material. 266
A.18 Precision–recall curves for MANN and ANN under the temporally faithful evaluation protocol. PR curves are emphasised due to class imbalance, where ROC summaries can be less informative. 267
A.19 Receiver operating characteristic (ROC) curve included for completeness for MANN/ANN comparison. Primary interpretation in this thesis uses PR-oriented evaluation due to the rarity of insider events. 267
A.20 Illustrative relationship between modular complexity (number of modules) and performance, indicating diminishing returns beyond moderate modular granularity. This supports balanced decomposition rather than arbitrarily fine modularisation. 268
A.21 Flowchart representation of the R-Guard training and evaluation procedure. This implementation-oriented figure complements the conceptual architecture and clarifies the sequence of processing stages used for behaviourally constrained adversarial robustness evaluation. . 275
A.22 Overall system performance summary comparing clean accuracy, robust accuracy, and computational/processing overhead for baseline and R-Guard configurations. The figure provides a supplementary deployment-oriented view of the accuracy–robustness–overhead relationship. 276
A.23 ROC curves under clean and adversarial conditions for representative baseline and R-Guard-augmented models. The figure provides a supplementary discrimination-focused view of adversarial degradation and robustness recovery. 277
A.24 Radar-chart summary of selected performance dimensions (e.g., clean accuracy, robust accuracy, and efficiency/overhead) for baseline and R-Guard configurations. This figure is included as a compact synthesis visual and is supplementary to the primary quantitative plots in Chapter 6. 278
A.25 Representative confusion-matrix comparison for a baseline detector and its R-Guard-augmented counterpart under behavioural adversarial evaluation. The figure is included as a diagnostic illustration of error-pattern changes, particularly false-negative reduction under stealthy attack conditions. 279

B.1 Methodological decision framework adapted for insider-threat detection research. 281

List of Tables

1.1 Theoretical rationale for the staged architectural progression. 17

2.1 Contemporary insider-threat case patterns: indicative signals, modelling implications, and links to thesis contributions 33
2.2 Behavioural constructs, their characteristics and implications for insider-threat machine learning. 40
2.3 Taxonomy of data leakage in insider-threat detection research. 44
2.4 Summary of methodological limitations in insider-threat machine learning and their consequences for validity, robustness and operational deployment. 46
2.5 Common machine-learning tasks and algorithm families across domains, with implications for insider-threat detection 49
2.6 Behaviourally constrained adversarial insider taxonomy. 67
2.7 Summary - Integrated synthesis of unresolved gaps in insider-threat ML research. 71
2.8 Literature-derived rationale for the staged architectural progression. . 74
2.9 Summary of ANN families and their suitability for insider-threat detection 79
2.10 CERT dataset criteria and mitigation strategies: technical and methodological considerations 90
2.11 CERT dataset criteria and mitigation strategies: validity, reproducibility, and ethical considerations 91
2.12 Insider-threat datasets: public benchmarks and suitability for this thesis 101
2.13 Insider-threat datasets: real-world organisational datasets (restricted access) 101
2.14 Insider-threat datasets: research-limited logs and study-specific synthetic datasets 102

3.1 Performance of the common $\mathrm{ANN}_{\mathrm{std}}$ baseline under the principal evaluation conditions used across the thesis. 127
3.2 Repeated-run statistical comparison of the standard ANN and SOMANN using PR-AUC ($n = 7$ paired runs). Model performance is reported as mean ± standard deviation. 129

3.3 Performance under user-disjoint evaluation on previously unseen users. Values are reported as mean ± standard deviation. 132

4.1 Behavioural domain decomposition for insider-threat modelling, highlighting semantic distinctions that motivate modular representation learning. 145
4.2 Reconciled comparative performance under the chronology-preserving Chapter 4 evaluation protocol. 155
4.3 Repeated-run statistical validation of MANN using PR-AUC ($n = 7$ paired runs). Model performance is reported as mean ± standard deviation. 158

5.1 Mapping between the three-tier conceptual behavioural hierarchy and the five-stage MLML computational pipeline. 175
5.2 Repeated-run statistical validation of MLML using PR-AUC ($n = 7$ paired runs). Model performance is reported as mean ± standard deviation. 189

6.1 Mapping between the Chapter 2 primitive perturbation operators and the Chapter 6 experimental attack classes. 209
6.2 Ablation study of the $\mathcal{R}$-Guard architecture on the CERT dataset. The results demonstrate the critical role of each component in maintaining robustness against sequential adversarial attacks. While the full $\mathcal{R}$-Guard configuration achieves the highest Robust Accuracy (89.0%), it incurs a minor trade-off in clean accuracy (92.4%). The removal of the Denoising Autoencoder (DAE) results in the most significant drop in resilience ($\Delta RA = -11.2$), highlighting its role as the primary defense against behavioral manipulation. 218
6.3 Repeated-run statistical validation of R-Guard under adversarial attack ($n = 7$ paired runs). Robust accuracy is reported as mean ± standard deviation. Differences are expressed in percentage points (pp). CR denotes consistency regularisation. 222

A.1 Performance comparison between baseline models and the SOM–ANN framework. The hybrid architecture demonstrates a substantial 12% absolute improvement in accuracy over the standalone ANN and achieves a high F1-score of 0.931. These results highlight the model's superior sensitivity to insider-threat activity while maintaining a low false-positive rate (1.56%) essential for operational efficiency. 259

A.2 Comparative 5-fold cross-validation results for the Modular Artificial Neural Network (MANN) and traditional Artificial Neural Network (ANN). The results demonstrate the superior stability and reliability of the MANN architecture, which consistently maintains an accuracy range of 91%–93% across all folds. This minimal performance deviation across different data partitions highlights the robustness of the modular approach in capturing complex behavioural patterns compared to the more variable performance of the monolithic ANN. . 264
A.3 Performance Comparison on the CERT Insider Threat Dataset. The results illustrate the comparative advantage of the proposed MLML framework against four established baseline models. Under identical pre-processing and parameter tuning conditions, the MLML framework achieves up to a 5% improvement in F1-Score and a 6% increase in AUC relative to the Hybrid Ensemble, currently the strongest baseline. These improvements are statistically significant ($p < 0.01$) according to a paired t-test, suggesting that the architecture provides a more robust foundation for distinguishing subtle threats from routine organisational activity. 269
A.4 Layer-wise Incremental Performance on the CERT Dataset. To evaluate the contribution of each architectural component, performance was measured after progressively activating layers L_1 through L_5. The results demonstrate consistent gains across all metrics as the model complexity increases. Notably, the integration of Hybrid Modules (L_1–L_3) provides the largest individual boost to the F1-score (+5%), while the full MLML framework (L_1–L_5) achieves the peak performance of 0.97 Accuracy and AUC, confirming the value of the final hierarchical reasoning stage. 269
A.5 Adaptive Learning Performance under Concept Drift. The results demonstrate the superior resilience of the proposed MLML framework compared to established baselines when exposed to simulated feature distribution shifts. The MLML architecture achieves a significantly lower Concept Drift Rate (CDR) of 0.25 and a high Model Stability Index (MSI) of 0.89. Furthermore, a post-drift F1-score of 0.92 and improved Retraining Efficiency (RE) suggest that the framework can maintain operational effectiveness with approximately 20% lower retraining costs than traditional batch-based approaches. 271

A.6 Ablation Study of the $\mathcal{R}$-Guard Architecture on the CERT Dataset. The results quantify the contribution of individual components to both clean and robust accuracy. While the full $\mathcal{R}$-Guard configuration prioritises resilience—achieving a Robust Accuracy of 89.0%—it incurs a marginal reduction in Clean Accuracy (92.4%) compared to variants without denoising or consistency constraints. Notably, the removal of the Denoising Autoencoder (DAE) leads to the most significant degradation in robustness ($\Delta RA = -11.2$), confirming its critical role in filtering adversarial perturbations while preserving behavioural plausibility. 273

D.1 Appendix D overview of implementation artefacts (D1–D4) and their methodological and ethical scope. 361

List of Algorithms

1 SOMANN Training Procedure (Perception–Inference two-stage learning) 118

2 IAA-Sequential adversarial evaluation procedure 213

3 (C1) ANN Baseline Implementation for Insider-Threat Detection (CERT multi-log) 283

4 (C2) SOM–ANN (SOMANN) Hybrid Implementation for Representation Stabilisation 284

5 (C3) MANN Implementation (Domain-Modular ANN with Structured Fusion) 285

6 (C4) Hierarchical Multi-Layered Implementation for Behavioural Progression Modelling 286

7 (C5) R-Guard Implementation for Behaviourally Constrained Adversarial Robustness 287

Chapter 1

Introduction

1.1 Background

Organisational cybersecurity has undergone a profound transition over the past decade. Traditional perimeter-based security models, designed primarily to repel external intrusions, are increasingly inadequate in environments characterised by distributed workforces, cloud-native infrastructures, privileged automation, and zero-trust access paradigms [1]. As security boundaries dissolve and internal systems become highly interconnected, the locus of risk has shifted from spatial intrusion to *behavioural deviation* within trusted environments.

This shift is reflected in both research and practice. Large-scale breach reporting continues to show that human and organisational factors remain central to many security failures, while industry assessments emphasise that insider risk is persistent and operationally costly [2], [3], [4]. Recent cost-of-insider-risk reporting further indicates that the average annual organisational cost associated with insider risk has increased over time, reinforcing the need for detection approaches that are not only accurate in benign settings, but also resilient under organisational stress and evolving operational conditions [5], [6], [7]. Unlike many external threat scenarios, insiders operate within legitimate access boundaries using valid credentials and authorised workflows; their actions are therefore embedded within normal organisational telemetry and may appear superficially plausible.

These realities motivate what may be termed the *behavioural turn* in cybersecurity: the recognition that security violations increasingly manifest not as overt boundary breaches, but as subtle deviations within legitimate behavioural manifolds. Consequently, detection systems must move beyond static rule violations and reason about behaviour, context, temporal progression, and role-dependent plausibility. This is particularly important for insider threats, where harmful actions may be distributed over time, shaped by organisational context, and coupled across multiple activity domains (e.g., authentication, device use, file access, web activity, and communication) [8], [9], [10].

However, a fundamental challenge emerges at this juncture: the **semantic gap** between low-level telemetry and high-level behavioural intent. Organisational systems produce vast streams of discrete logs—authentication events, file interactions, email exchanges, process executions, and device activities—that constitute syntactic artefacts of activity. Insider-threat detection, by contrast, requires interpreting these signals in terms of behavioural meaning, organisational context, privilege semantics, and temporal plausibility. This gap is amplified by the socio-technical nature of insider behaviour, which is shaped by psychological stressors, workplace climate, policy interpretation, and interpersonal dynamics [8], [11], [12].

Many existing machine-learning (ML) systems attempt to bridge this semantic gap in a single monolithic step: high-dimensional log-derived features are fed directly into statistical or deep-learning classifiers, with the expectation that representation learning alone will infer behavioural intent. Systematic reviews suggest that such approaches are often undermined by weak behavioural grounding, limited cross-domain semantics, severe class imbalance, and evaluation fragility—including temporally invalid split strategies and leakage-prone preprocessing that can inflate performance [10], [13], [14]. In practice, these weaknesses can produce models that appear strong under benchmark conditions but behave unreliably when exposed to behavioural drift, role changes, or adaptive insider strategies.

Insider behaviour evolves through trajectories influenced by cognitive stressors, role transitions, workload rhythms, organisational restructuring, and contextual pressures. These dynamics introduce non-stationarity and semantic variability that challenge static, monolithic modelling approaches. The problem is therefore not merely one of predictive accuracy, but of *representational adequacy* and *evaluation validity*: models must learn behavioural structure in ways that remain interpretable, temporally faithful, and robust to plausible adaptation.

This thesis is motivated by the argument that insider-threat detection requires architectural designs explicitly aligned with behavioural structure. Rather than treating detection as a purely statistical classification problem, the research reframes it as *behavioural reasoning under non-stationarity and adversarial constraints*. Addressing this challenge motivates a staged progression of architectures and evaluation principles developed in subsequent chapters, including stability-oriented representation structuring, domain-aware modular decomposition, hierarchical behavioural abstraction, and behaviourally grounded robustness assessment.

1.1.1 Operationalising Behavioural Semantics

In this thesis, **behavioural semantics** refers to the contextual meaning carried by organisational activity when it is interpreted in relation to the functional domain in which an event occurs, the workflow or activity sequence to which it belongs, and its temporal position within a user's evolving behavioural trajectory. Accordingly,

behavioural semantics is not equated with user identity or with the frequency of individual events. Rather, it concerns whether an observed action is behaviourally meaningful in the context in which it occurs.

Let an organisational activity stream be represented as

$$\mathcal{X}_{1:T} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}, \tag{1.1}$$

where each $\mathbf{x}_t$ contains observable attributes associated with an activity at time t. A representation function

$$f_\theta : \mathcal{X}_{1:T} \to \mathcal{Z}_{1:T} \tag{1.2}$$

maps these observations to a latent behavioural representation $\mathcal{Z}_{1:T}$.

Within the framework developed in this thesis, the semantic content that this representation is intended to preserve is characterised along three complementary dimensions:

1. **Role and privilege context ($\mathcal{S}_{\text{role}}$):** whether an activity is consistent with the operational responsibilities, access privileges, and behavioural expectations associated with the user's functional context. This does not require preservation of individual user identity. Instead, the objective is to retain behaviourally relevant role context while reducing dependence on identity-specific characteristics.

2. **Domain and workflow context ($\mathcal{S}_{\text{flow}}$):** the functional relationship between activities occurring within and across behavioural domains, including authentication, file access, communication, web activity, and device usage. Individual events may appear benign when considered independently but acquire different behavioural meaning when interpreted as part of a cross-domain sequence.

3. **Temporal trajectory context ($\mathcal{S}_{\text{temp}}$):** the evolution of behaviour across multiple temporal horizons. This captures the distinction between isolated deviations and sustained behavioural change, including gradual drift, repeated anomalous activity, and longer-horizon escalation.

Conceptually, the behavioural-semantic content targeted by the learned representation can therefore be expressed as

$$\mathcal{S}_{\text{beh}} = \{\mathcal{S}_{\text{role}}, \mathcal{S}_{\text{flow}}, \mathcal{S}_{\text{temp}}\}, \tag{1.3}$$

with

$$\mathcal{Z}_{1:T} = f_\theta\left(\mathcal{X}_{1:T}\right). \tag{1.4}$$

The objective is for $\mathcal{Z}_{1:T}$ to preserve information associated with $\mathcal{S}_{\text{beh}}$ that is relevant to behavioural discrimination. The purpose is therefore not to reconstruct the raw telemetry exactly, nor to treat the semantic dimensions in Equation 1.3 as explicit additional inputs to the model. Rather, these dimensions characterise the contextual relationships that the learned representation is intended to retain.

In this thesis, **semantic preservation is treated as an empirical property rather than assumed from the architecture alone**. A representation is considered to provide stronger evidence of behavioural-semantic preservation when it satisfies three corresponding criteria:

1. behavioural structure remains informative while dependence on individual user identity is reduced;

2. removal or disruption of behaviourally relevant domain relationships produces measurable degradation in detection performance; and

3. disruption of meaningful temporal or sequential relationships changes the model response in a manner consistent with the loss of behavioural context.

Evidence for these properties is provided progressively across the experimental chapters. Chapter 3 evaluates the distinction between behavioural structure and identity-specific representation through SOM topology and out-of-user generalisation analysis. Chapter 4 examines the contribution of domain-specific representations and controlled fusion, including ablation of behavioural domains. Chapter 5 evaluates the contribution of temporal hierarchy to detecting behaviour that develops across multiple observation horizons. Chapter 6 subsequently examines whether behaviourally relevant representations remain useful when event timing, ordering, and other observable characteristics are deliberately perturbed.

Consequently, behavioural semantics in this thesis is not claimed to constitute a complete semantic model of human intent. It is an *operational definition*: the preservation of role-relevant, domain-dependent, workflow-dependent, and temporally structured behavioural relationships that remain informative for insider-threat discrimination while reducing reliance on incidental identity-specific or organisational artefacts. This definition provides the conceptual basis for the staged architectural progression developed in Chapters 3–6.

1.2 Problem Statement

Insider-threat detection remains a persistent challenge for organisational cybersecurity because it requires distinguishing harmful intent from legitimate activity within trusted environments. Unlike many external attacks, insider actions frequently occur through authorised credentials, privilege-consistent workflows, and role-aligned

behaviours, which makes malicious activity difficult to separate from benign organisational variation [8], [11], [15]. As a result, conventional security monitoring approaches that rely on static signatures, isolated anomaly flags, or single-stream indicators are often insufficient for identifying insider escalation trajectories that unfold gradually across time and multiple activity domains.

The core research problem addressed by this thesis is that many existing machine-learning (ML) approaches to insider-threat detection exhibit a mismatch between the *behavioural semantics* of insider risk and the *representational and evaluation assumptions* used to build and validate detection systems. In the literature, insider behaviour is commonly encoded as flattened feature vectors or short windows derived from heterogeneous logs. While such representations can support baseline classification, they frequently omit organisational context, privilege semantics, and cross-domain behavioural dependencies that are critical for interpreting intent and recognising subtle misuse [9], [10]. This representational limitation contributes to two recurring failure modes: inflated false-positive rates driven by benign deviations (e.g., workload surges, legitimate overtime, role transitions), and missed detections of privilege-consistent misuse embedded within plausible workflows.

A second and closely related problem concerns *methodological rigour and evaluation validity*. Systematic reviews and critical analyses of benchmark-based insider-threat studies highlight widespread issues, including temporally invalid split strategies, leakage-prone preprocessing, inconsistent reporting, and selective metric interpretation [10], [13], [14]. These weaknesses can produce models that appear to perform strongly on benchmark datasets yet do not provide reliable evidence of generalisation under realistic organisational conditions. In insider-threat detection, where behaviour evolves and operational constraints matter, evaluation discipline is not a peripheral concern; it is fundamental to whether reported improvements are meaningful.

A third problem is that insider-threat detection systems are increasingly exposed to *behavioural drift* and *adaptive behaviour*. Organisational behaviour is non-stationary due to evolving workloads, policy changes, personnel movement, and technology adoption. Moreover, insiders may adapt strategically in response to monitoring, attempting to remain within behavioural baselines while progressing toward harmful outcomes. These conditions imply that high nominal predictive performance under benign assumptions does not guarantee robustness when the data distribution shifts or when behaviour is manipulated in plausible ways [16], [17]. Robust insider-threat detection therefore requires not only improved predictive modelling, but also architectural and evaluation mechanisms that explicitly address behavioural plausibility, temporal coherence, and adaptation.

1.2.1 Behavioural Dependencies and Spurious Correlations

Building on the operational definition of behavioural semantics introduced in Section **??**, a central difficulty in behavioural insider-threat modelling is that statistical dependence does not necessarily imply behavioural relevance. Enterprise telemetry contains numerous correlations arising from organisational structure, including user identity, department membership, workstation allocation, shift patterns, job function, and differences in activity volume. Such variables may be highly predictive within a particular dataset while providing limited evidence of malicious or risky behaviour. A model that relies heavily on these correlations may therefore achieve strong within-distribution performance without learning relationships that generalise across users, roles, temporal conditions, or organisational settings.

This thesis distinguishes between *meaningful behavioural dependencies* and *spurious organisational correlations*. A meaningful behavioural dependency is defined here as a statistical relationship whose predictive relevance is associated with the context, interaction, or temporal evolution of user activity and remains informative when plausible sources of incidental organisational dependence are controlled, removed, or disrupted. By contrast, a spurious organisational correlation is a predictive association that primarily reflects incidental properties of the dataset or organisation rather than the behavioural process of interest.

The distinction cannot be established from predictive association alone. In particular, a feature or latent representation should not be regarded as behaviourally meaningful solely because it exhibits a strong statistical association with the insider-threat outcome. Such an association may arise from contextual or organisational variables, including user identity, role, activity volume, workstation allocation, or other dataset-specific regularities.

Accordingly, this thesis treats behavioural relevance as an empirical property requiring convergent evidence that learned relationships remain informative when plausible sources of incidental dependence are controlled, removed, or disrupted. The objective is not to establish causality between individual behavioural features and malicious intent. Rather, it is to examine whether the learned representation remains informative when reliance on identity-specific, domain-dominant, temporally incidental, or otherwise dataset-specific structure is reduced.

Four complementary forms of evidence are used across the experimental chapters to examine this distinction.

Identity and role dependence. A representation should not derive its discriminatory power primarily from recognising individual users. Identity-dependence analysis and out-of-user evaluation are therefore used to examine whether performance persists when the model is required to generalise beyond identities observed during training. Role information is treated differently from individual identity because

legitimate behavioural expectations may depend on functional responsibilities and privilege context. The objective is consequently not to remove all role-related information, but to reduce reliance on identity-specific artefacts while retaining behaviourally relevant contextual structure. Chapter 3 evaluates this distinction through analysis of the relationship between SOM representations and user identity, together with user-disjoint generalisation experiments.

Domain and interaction dependence. Behavioural meaning may arise from relationships between activity domains rather than from isolated feature magnitudes. For example, authentication activity or file-access volume may each be unremarkable when considered independently, whereas their relationship to other behavioural domains may provide stronger evidence of behavioural deviation. Chapter 4 examines this distinction through domain-specific representation, controlled fusion, and ablation analysis in which individual behavioural domains are removed to evaluate their contribution to the final decision.

Temporal persistence and sequence dependence. A further distinction concerns whether predictive information reflects persistent behavioural evolution or merely transient coincidence. Relationships that remain informative under chronology-preserving evaluation and across multiple temporal horizons provide stronger evidence of structured behavioural dependence than relationships that depend only on isolated local observations. Chapter 5 examines this issue through hierarchical multi-horizon modelling, including the temporal blind-spot analysis used to evaluate behavioural patterns whose discriminatory information develops across multiple observation periods.

Robustness to behaviourally constrained perturbation. A relationship that appears predictive under nominal conditions may still be fragile if small, behaviourally feasible modifications to observable activity substantially alter the model response while the underlying malicious objective remains unchanged. Chapter 6 therefore extends the evaluation by examining sensitivity to constrained changes in event timing, ordering, and other observable behavioural characteristics. This provides a further test of whether the learned decision structure depends on robust behavioural relationships or on brittle properties of the observed sequence.

Taken together, these analyses provide complementary rather than causal evidence. Identity-independent generalisation examines whether representations remain useful beyond known users; domain ablation evaluates dependence on functionally distinct behavioural sources; temporal evaluation examines whether longer-horizon behavioural structure contributes information beyond local observations; and controlled perturbation tests whether predictive relationships remain informative when observable behaviour is deliberately modified within the defined threat model.

Accordingly, this thesis does not assume that statistical predictiveness is equivalent to behavioural meaning. Meaningful behavioural dependency is instead treated as a property requiring convergent empirical support from identity-independent generalisation, domain ablation, chronology-preserving temporal evaluation, and behaviourally constrained perturbation. This distinction is particularly important for insider-threat detection because organisational datasets may contain stable regularities that machine-learning models can exploit without those regularities necessarily representing the behavioural process of interest.

The staged architectural investigation developed in this thesis progressively addresses this risk. SOMANN introduces topology-preserving representation learning intended to reduce sensitivity to isolated feature variation and identity-specific structure; MANN separates heterogeneous behavioural domains before controlled fusion; MLML introduces temporal abstraction to distinguish local deviations from longer-term behavioural progression; and $\mathcal{R}$-Guard evaluates whether the resulting behavioural relationships remain informative when observable activity is deliberately manipulated.

The objective of this progression is not to eliminate all organisational correlation, which would neither be feasible nor necessarily desirable. Rather, it is to make the basis of prediction progressively more behaviourally structured, empirically testable, and less dependent on incidental properties of a particular user population or observation period.

In summary, the problem addressed in this thesis can be stated as follows: *how can insider-threat detection systems be designed and evaluated to achieve reliable performance when insider behaviour is multi-domain, temporally progressive, semantically constrained by roles and privileges, and subject to behavioural drift and adaptive manipulation?* Addressing this problem motivates the thesis's staged architecture progression. The research begins by establishing a disciplined ANN baseline and evaluation protocol, then develops structured learning designs that progressively strengthen behavioural alignment: (i) representation stabilisation through SOM–ANN hybridisation, (ii) domain-aware modular decomposition through MANN, (iii) multi-level behavioural abstraction through hierarchical modelling, and (iv) behaviourally grounded robustness mechanisms and adversarial evaluation through R-Guard.

1.3 Research Questions

To address the research problem articulated in Section 1.2, this thesis is guided by a set of interrelated research questions that progressively structure the architectural development undertaken in subsequent chapters. The questions are formulated to reflect the socio-technical and temporal characteristics of insider behaviour, the

multi-domain nature of enterprise telemetry, and the increasing need for robustness against adaptive manipulation. Together, they define a staged investigation in which each chapter makes a distinct contribution while maintaining a coherent comparative reference through shared dataset assumptions and evaluation discipline.

RQ1: Behavioural Representation and Variance Stabilisation

> *How can behavioural representations be stabilised to preserve semantic meaning while mitigating high intra-user variance in insider activity logs?*

A persistent difficulty in insider-threat detection is that legitimate behaviour exhibits substantial intra-user variance across time, workload conditions, and organisational context. In many settings, benign activity can be highly irregular (e.g., deadline-driven bursts, role transitions, exceptional projects), while malicious behaviour may remain privilege-consistent and locally plausible. As a result, representations derived directly from raw logs or short aggregation windows can be unstable and can cause models to learn variance patterns rather than behaviourally meaningful structure. This motivates the need for representation learning strategies that stabilise behavioural manifolds while retaining semantic information relevant to intent and policy deviation.

This question motivates the thesis's first architectural contribution: the use of a Self-Organising Map (SOM) as an unsupervised topological structuring mechanism prior to supervised ANN classification (Chapter 3). The hypothesis is that topology-preserving clustering can reduce sensitivity to noise and variance by mapping high-dimensional behavioural activity into structured neighbourhoods, thereby supporting more robust downstream discrimination. Evaluation therefore focuses not only on predictive performance, but also on whether stabilised representations improve generalisation under temporally faithful protocols and reduce false positives arising from benign behavioural fluctuation.

RQ2: Domain Decomposition and Cross-Domain Interaction

> *Can modular neural architectures improve detection performance and generalisation by decomposing insider behaviour into semantically distinct domains while preserving cross-domain coherence?*

Insider activity is intrinsically multi-domain. Authentication behaviour, file interactions, device usage, web activity, and communications each reflect different operational semantics, statistical structure, and levels of behavioural variability. Monolithic models that fuse all features into a single representation often suffer from feature dominance and semantic dilution, where high-volume or noisy domains

overshadow subtler but risk-relevant signals. Moreover, the interpretability of model outputs is reduced when domain contributions are not separable. These limitations motivate architectural designs that preserve domain semantics while still allowing evidence to be integrated into a coherent risk estimate.

This question motivates the Modular Artificial Neural Network (MANN) contribution (Chapter 4). The core idea is to train domain-specialised modules that learn representations aligned with the semantics of each log source, followed by principled fusion to preserve cross-domain coherence. Evaluation addresses whether modular decomposition improves generalisation, reduces overfitting to dominant domains, and supports more interpretable risk reasoning. In addition, this question supports later synthesis by providing architectural foundations for cross-domain consistency constraints that become important under adversarial settings.

RQ3: Multi-Level Behavioural Abstraction

> *How can hierarchical modelling capture behavioural evolution across events, sessions, tasks, and organisational roles without collapsing semantic structure?*

A defining characteristic of insider threats is that they often unfold as trajectories across multiple temporal and semantic levels. Individual events may appear benign in isolation, while risk emerges through longer-horizon progression and changes in behavioural state relative to role expectations and organisational context. Flat feature representations and single-level models can obscure this progression, collapsing higher-level behavioural meaning into short-window statistics. This motivates the need for hierarchical abstraction mechanisms that preserve behavioural meaning across nested levels (event $\rightarrow$ session $\rightarrow$ task $\rightarrow$ role/context), while maintaining temporal validity and interpretability.

This question motivates the hierarchical / multi-layered modelling framework presented in Chapter 5. The focus is on whether hierarchical abstraction improves sensitivity to gradual behavioural drift, supports earlier detection of escalation trajectories, and enhances interpretability by mapping detected risk to behavioural levels rather than to unstructured feature aggregates. Evaluation therefore includes both predictive metrics and trajectory-oriented interpretation, ensuring that claimed improvements are attributable to multi-level modelling rather than to artefacts of split design or leakage.

RQ4: Behaviourally Constrained Adversarial Robustness

> *How can insider-threat detection architectures be designed to remain robust against behaviourally constrained adversarial manipulation while respecting organisational feasibility constraints?*

Insider-threat detection is increasingly exposed to adaptive behaviour. A motivated insider can attempt to evade detection by keeping actions privilege-consistent, temporally plausible, and aligned with role expectations while still achieving harmful objectives. Classical adversarial machine learning often assumes unconstrained perturbations in feature space (e.g., norm-bounded noise), but insider settings impose stronger constraints: perturbations must remain behaviourally feasible and operationally plausible. Robustness in this domain therefore requires threat models that reflect feasible action spaces and defences that preserve detection sensitivity under such constraints.

This question motivates Chapter 6, which introduces a behaviourally constrained adversarial threat model and evaluates robustness under sequential, plausible perturbations rather than purely synthetic feature-space manipulations. The R-Guard framework is positioned as a response to the gap between nominal accuracy and robustness under adaptive insider behaviour. Evaluation focuses on robustness metrics (including robustness curves and time-to-detection under drift/attack) and emphasises interpretive boundaries appropriate for benchmark-driven research.

RQ5: Architectural Integration and Operational Validity

> *What architectural principles enable the integration of behavioural grounding, domain decomposition, hierarchical abstraction, and adversarial robustness into a coherent insider-threat detection framework suitable for real-world deployment?*

While each of the preceding questions addresses a specific limitation observed in the literature, operationally useful insider-threat detection requires coherent integration rather than isolated component improvements. In practice, models must balance predictive sensitivity with false-positive control, maintain interpretability for analyst-centred decision support, and remain robust under non-stationarity and adaptive behaviour. Moreover, deployment must be governed by ethical and organisational constraints, including proportionality, privacy, accountability, and the risk of harmful overreach.

This question therefore synthesises insights from Chapters 3–6 to identify design principles that support integration. It focuses on how behavioural grounding (context and plausibility), modular semantics (multi-log decomposition), hierarchical abstraction (trajectory modelling), and robustness mechanisms (behaviourally constrained adversarial evaluation) can be combined into a coherent framework. The outcome is a set of principles and validated trade-offs that clarify what the thesis contributions enable and what validity boundaries remain.

Summary. Collectively, these research questions structure the thesis as a progressive architectural investigation. Each chapter addresses a distinct dimension

of the insider-threat problem while maintaining a coherent comparative reference through consistent dataset assumptions and disciplined evaluation protocols. This structure supports both scientific contribution (new architectural frameworks and robustness mechanisms) and methodological contribution (evaluation discipline and validity-bounded interpretation aligned with insider behavioural realities).

1.4 Aim and Objectives

1.4.1 Research Aim

The overarching aim of this thesis is to design, implement, and rigorously evaluate a behaviourally grounded machine-learning framework for insider-threat detection that remains reliable under multi-domain behavioural telemetry, temporal progression, behavioural drift, and behaviourally constrained adaptive manipulation. The aim is pursued through a staged architectural progression, establishing a disciplined ANN baseline and then introducing structured extensions that improve representation stability, preserve domain semantics, capture multi-level behavioural abstraction, and strengthen robustness under realistic adversarial constraints.

1.4.2 Research Objectives

To achieve this aim, the thesis addresses the following objectives:

1. **Establish a rigorous baseline and evaluation discipline for insider-threat detection.** Develop a reproducible baseline using ANN-based modelling and define temporally faithful evaluation protocols (including chronology-preserving splits and imbalance-aware metric reporting) suitable for CERT-style insider behavioural datasets. This objective provides a consistent comparative reference for interpreting subsequent architectural improvements and mitigates common evaluation pitfalls such as leakage and optimistic reporting.

2. **Improve behavioural representation stability under high intra-user variance.** Design and evaluate a SOM–ANN hybrid representation strategy that stabilises behavioural manifolds prior to supervised discrimination. The objective is to assess whether topology-preserving unsupervised structuring reduces sensitivity to noise and benign behavioural fluctuation, improves generalisation, and contributes to more interpretable behavioural mapping under temporally valid evaluation conditions (RQ1; Chapter 3).

3. **Develop a modular modelling framework that preserves domain semantics while enabling cross-domain reasoning.** Propose and evaluate a Modular Artificial Neural Network (MANN) architecture that decomposes

insider behaviour into semantically distinct domains (e.g., logon, file, device, web, and communication) and integrates evidence through principled fusion. This objective assesses whether modularity reduces feature dominance, improves generalisability, supports domain-level interpretability, and provides a stronger foundation for cross-domain consistency under adaptation (RQ2; Chapter 4).

4. **Capture multi-level behavioural progression through hierarchical abstraction.** Design and evaluate a hierarchical / multi-layered modelling approach capable of representing behavioural evolution across nested temporal and semantic levels (event → session → task → role/context). This objective examines whether hierarchical abstraction improves sensitivity to gradual escalation, supports drift-aware interpretation, and yields more behaviourally meaningful explanations than flat representations (RQ3; Chapter 5).

5. **Formulate and test behaviourally constrained adversarial robustness in insider settings.** Develop a behaviourally constrained adversarial threat model that reflects feasible insider action spaces and evaluate robustness under sequential, plausible perturbations rather than purely synthetic feature-space noise. This objective introduces and assesses robustness mechanisms (R-Guard) designed to preserve detection performance, maintain temporal coherence, and support bounded interpretability under adaptive manipulation (RQ4; Chapter 6).

6. **Synthesize architectural principles, validity boundaries, and deployment-relevant implications.** Integrate findings from the staged progression to identify design principles and trade-offs that support operationally meaningful insider-threat detection, balancing predictive performance, false-positive control, interpretability, and robustness. This objective also clarifies validity boundaries associated with benchmark-driven experimentation (CERT), highlights ethical and governance constraints, and defines directions for future validation and deployment-oriented research (RQ5; Chapters 6–7).

Alignment with research questions and thesis structure. The objectives map directly onto the research questions introduced in Section 1.3 and are implemented through the thesis architecture progression in Chapters 3–6. This alignment ensures that each contribution is both methodologically grounded (through disciplined evaluation and dataset rationale) and technically justified (through structured model design responding to identified insider-threat detection failure modes).

1.4.3 Scope and Boundaries

This thesis addresses insider-threat detection as a behavioural security analytics problem under organisational constraints. The scope is intentionally bounded to support rigorous comparative architectural evaluation and validity-bounded interpretation.

Data scope. Experimental evaluation is conducted primarily using the CERT insider-threat dataset as a reproducible benchmark backbone. This choice supports controlled comparison across the staged architecture progression, but it also imposes validity boundaries associated with synthetic/simulated data. Accordingly, the thesis interprets results as benchmark-grounded evidence of relative architectural behaviour and robustness trends rather than as definitive estimates of deployment-ready performance.

Threat scope. The thesis focuses on insider-risk behaviours observable through enterprise activity logs and related telemetry, including patterns consistent with misuse, policy violations, and adaptive behaviours that remain privilege-consistent. The work does not attempt to infer psychological intent directly; instead, it models behavioural risk indicators and trajectory patterns under declared assumptions.

Method scope. The thesis emphasises temporally faithful evaluation, imbalance-aware metrics, and robustness-oriented testing under behaviourally constrained adversarial manipulation. While multiple modelling families are discussed, the thesis contributions centre on ANN-based pipelines and structured extensions (hybrid, modular, hierarchical, and robust) that are evaluated under consistent protocols.

Operational and ethical scope. The proposed methods are framed as decision-support tools rather than autonomous judgement systems. The thesis does not advocate ungoverned employee surveillance or punitive automation; instead, it emphasises proportionality, governance, and responsible interpretation, and it highlights the risks of false positives and misuse in operational settings.

1.4.4 Rationale for the Staged Architectural Progression

The progression from the standard ANN baseline through SOMANN, MANN, MLML, and ultimately $\mathcal{R}$-Guard is not intended to imply that these architectures constitute the only possible sequence for insider-threat detection, nor that each later architecture is mathematically dependent upon its predecessor. Rather, the progression is *theoretically motivated and empirically cumulative*: each stage addresses a distinct modelling limitation that remains unresolved after the preceding investigation and introduces an additional structural requirement for behaviourally grounded detection.

The sequence can therefore be understood as a progression across four increasingly demanding modelling requirements:

$$\underbrace{\mathcal{R}_s}_{\text{stability}} \rightarrow \underbrace{\mathcal{R}_d}_{\text{domain grounding}} \rightarrow \underbrace{\mathcal{R}_t}_{\text{temporal structure}} \rightarrow \underbrace{\mathcal{R}_a}_{\text{adversarial robustness}}, \tag{1.5}$$

where $\mathcal{R}_s$ denotes the requirement for stable behavioural representation, $\mathcal{R}_d$ the preservation of domain-specific behavioural structure, $\mathcal{R}_t$ the representation of behaviour across multiple temporal horizons, and $\mathcal{R}_a$ robustness to intentional behavioural manipulation. The ordering reflects increasing representational and threat-model complexity rather than an assertion of mathematical uniqueness.

Stage 1: From raw classification to stable behavioural perception. The standard ANN baseline operates directly on aggregated behavioural features and therefore provides no explicit mechanism for separating transient behavioural variability from more persistent structure. Chapter 3 addresses this representational problem through SOMANN, which separates unsupervised perceptual organisation from supervised inference. The SOM introduces a topology-preserving representation in which behaviourally similar observations are mapped to neighbouring prototypes before supervised classification. This stage therefore investigates whether stabilising behavioural perception can improve downstream discrimination without relying exclusively on the raw feature space.

The contribution of this stage is consequently not that SOMANN eliminates behavioural noise, but that it introduces an architectural bias towards locally smooth and topology-consistent representation. The experiments in Chapter 3 evaluate whether this representational bias improves stability and generalisation relative to the common ANN baseline.

Stage 2: From stable representation to domain-grounded representation. Representational stability alone does not resolve the heterogeneity of enterprise telemetry. Authentication, file access, communication, web, and device activity represent behaviourally different processes, and their statistical distributions and operational meanings are not interchangeable. A unified representation can therefore remain stable while still allowing high-volume domains to dominate weaker but potentially informative behavioural signals.

Chapter 4 addresses this residual limitation through MANN. Rather than treating heterogeneous telemetry as a single undifferentiated feature space, MANN learns domain-specific representations before controlled fusion. The theoretical progression from SOMANN to MANN therefore reflects a change in modelling requirement: from asking whether behavioural observations can be represented more stably to asking whether the representation also preserves the functional distinctions between the behavioural domains from which those observations originate.

Importantly, MANN is not claimed to be mathematically dependent upon SOMANN. The dependency is conceptual and experimental: the investigation in

Chapter 3 establishes the importance of representation quality, while Chapter 4 examines the additional hypothesis that heterogeneous behavioural domains should be represented separately before controlled integration.

Stage 3: From domain grounding to temporal behavioural progression. Domain decomposition introduces functional structure but does not, by itself, provide a representation of how behaviour develops across longer temporal horizons. An action that appears benign within a single observation window may acquire different significance when it forms part of a sustained pattern of behavioural drift, repeated policy deviation, or progressive escalation.

Chapter 5 therefore extends the modelling problem from domain structure to temporal structure through the MLML hierarchy. Short-horizon behavioural representations are progressively integrated into higher-level states representing intermediate and longer-term behavioural trajectories. This stage addresses a temporal blind spot that remains possible in non-hierarchical architectures: locally plausible observations may collectively form a behaviourally significant trajectory even when no individual observation is sufficiently anomalous to trigger detection.

The hierarchy is therefore introduced not simply to increase model capacity, but to represent a form of information that is not explicitly modelled by the preceding architectures: relationships between behavioural states occurring across different temporal scales.

Stage 4: From predictive representation to adversarial robustness. The first three stages primarily evaluate behaviour under nominal or naturally changing conditions. This assumption becomes increasingly restrictive in the insider-threat setting because an informed adversary may adapt the timing, ordering, frequency, or other observable characteristics of activity while remaining within legitimate privilege and workflow constraints.

Chapter 6 consequently changes the threat model. $\mathcal{R}$-Guard evaluates whether behaviourally grounded representations remain reliable when observations are intentionally perturbed under behavioural feasibility constraints. Robustness is therefore treated as a distinct requirement from predictive performance: a representation may be stable, domain-aware, and temporally informative while nevertheless remaining vulnerable to strategic manipulation.

This final stage does not imply that adversarial defence could not be applied to a simpler classifier. Rather, the thesis deliberately evaluates adversarial robustness after establishing the behavioural structures whose preservation under attack is of interest. This ordering enables robustness to be interpreted with respect to behavioural structure rather than solely as resistance to unconstrained feature perturbation.

Taken together, the staged progression is therefore defended on the basis of

Table 1.1: Theoretical rationale for the staged architectural progression.

Stage	Primary Requirement	Residual Limitation	Architectural Response
ANN baseline	Predictive discrimination	No explicit mechanism for topology-preserving behavioural representation, leaving prediction sensitive to heterogeneous feature variation.	SOMANN introduces topology-preserving perceptual organisation before supervised inference.
SOMANN	Representation stability	A unified representation does not explicitly preserve distinctions between heterogeneous behavioural domains.	MANN introduces domain-specific representation followed by controlled fusion.
MANN	Domain-grounded representation	Domain-aware representations may remain temporally local and fail to represent longer-horizon behavioural progression.	MLML introduces hierarchical temporal abstraction across multiple behavioural horizons.
MLML	Temporal trajectory modelling	Predictive representations remain potentially vulnerable to deliberate manipulation of observable behaviour.	$\mathcal{R}$-Guard introduces robustness-aware evaluation and defence under behaviourally constrained perturbations.

residual modelling requirements rather than architectural novelty alone. Each stage retains the principal modelling concern established by the preceding investigation while introducing additional structure required to examine a limitation that the preceding architecture does not explicitly address. The progression is cumulative in research logic, but it is not asserted to be the unique mathematically necessary ordering of all possible insider-threat architectures.

This distinction is important. The thesis does not claim that SOMANN, MANN, MLML, and $\mathcal{R}$-Guard form an indivisible computational pipeline. Instead, they constitute a sequence of experimentally investigated architectural responses to increasingly demanding properties of insider-threat detection: stable representation, behavioural-domain grounding, temporal progression, and robustness under strategic manipulation.

Taken together, these requirements provide the basis for what this thesis terms the *Staged Behavioural Grounding and Robustness Principle*: reliable insider-threat detection requires behavioural information to be progressively structured at the representational, domain, and temporal levels before robustness to intentional behavioural manipulation can be meaningfully evaluated. Chapter 7 returns to this principle and evaluates it against the combined empirical evidence from Chapters 3–6.

1.5 Methodological Approach

This thesis adopts a design-science and experimental methodology in which architectural design choices are treated as testable hypotheses about behavioural representation, generalisation, and robustness in insider-threat detection. The overall approach is comparative and progressive: a disciplined baseline is established first, and subsequent chapters introduce structured architectural refinements designed to address specific failure modes identified in the literature (e.g., representational instability, domain heterogeneity, lack of behavioural hierarchy, and vulnerability under adaptive manipulation). Throughout, the emphasis is on reproducibility, temporally faithful evaluation, and validity-bounded interpretation rather than on maximising headline accuracy under potentially optimistic protocols.

1.5.1 Research Design

The research is organised as a staged architecture progression aligned with the research questions in Section 1.3 and the objectives in Section 1.4. The progression follows a principled sequence:

1. **Baseline modelling:** establish an ANN-based discriminative reference and evaluation discipline for insider-threat detection.

2. **Representation stabilisation:** introduce SOM–ANN hybridisation to reduce sensitivity to intra-user variance and to impose topology-preserving structure prior to classification.

3. **Domain semantics and modularity:** introduce modular decomposition (MANN) to preserve behavioural domain semantics and mitigate feature dominance across heterogeneous logs.

4. **Hierarchical behavioural abstraction:** introduce multi-layered modelling to capture behavioural progression across nested temporal and semantic levels.

5. **Behaviourally grounded robustness:** introduce R-Guard and behaviourally constrained adversarial evaluation to assess and improve resilience under adaptive manipulation.

This progression ensures that each contribution is evaluated relative to a consistent baseline, and that observed improvements can be interpreted as architectural effects rather than artefacts of changing datasets, inconsistent preprocessing, or evaluation leakage.

1.5.2 Dataset and Experiments

All experimental chapters use the CERT insider-threat dataset as a common benchmark. The decision to use CERT throughout is motivated by its multi-log structure, public availability, and suitability for reproducible comparative evaluation across multiple modelling stages, while recognising explicit validity boundaries associated with synthetic/simulated benchmarks [10], [13], [18]. Using a constant dataset supports internal comparability across Chapters 3–6 and reduces confounding effects that would arise from switching datasets between contributions. At the same time, the thesis interprets findings as benchmark-grounded evidence of relative architectural behaviour and robustness trends, rather than as definitive claims of deployment-ready performance.

1.5.3 Data Processing and Feature Design

Insider activity logs are transformed into behaviourally meaningful representations suitable for ML learning. The thesis follows a structured data-processing principle: transformations should preserve chronology, avoid leakage-prone fitting on future data, and retain interpretable behavioural semantics (e.g., workload rhythm indicators, after-hours activity ratios, cross-domain coupling features). Where sequential or windowed representations are used, windowing choices are justified in relation to behavioural plausibility and evaluation validity.

Because feature and representation design can strongly influence results, the thesis reports feature construction assumptions and preprocessing decisions alongside model configurations where appropriate. This reporting discipline supports reproducibility and ensures that architectural comparisons remain interpretable.

1.5.4 Evaluation Protocol

A core methodological contribution of the thesis is its evaluation discipline. Insider-threat detection is highly susceptible to performance inflation through unrealistic split strategies and leakage-prone preprocessing [13]. To mitigate this, evaluation protocols are designed around the following safeguards:

- **Chronology-preserving splitting:** training, validation, and test partitions are constructed to prevent future information from influencing model training, supporting time-faithful generalisation assessment.
- **Blocked temporal folds (where applicable):** when cross-validation is used, temporal blocking is applied to preserve chronology and reduce leakage risk.

- **Training-only fitting of preprocessing:** normalisation, encoding, and feature transformations are fitted on training data only and then applied to validation/test partitions.
- **Imbalance-aware metrics:** evaluation prioritises metrics appropriate to rare-event detection (e.g., precision, recall, F_1, PR–AUC), and interprets accuracy cautiously under severe imbalance [10], [19].
- **Threshold calibration:** decision thresholds are calibrated using validation data to reflect operational trade-offs, particularly the false-positive burden on analysts.

These safeguards provide a consistent interpretive basis for comparing baseline models with structured architectures and for assessing robustness under drift and adversarial manipulation.

1.5.5 Robustness and Adversarial Evaluation

Beyond nominal predictive performance, the thesis evaluates robustness under behavioural drift and behaviourally constrained adversarial manipulation. Classical adversarial ML commonly assumes unconstrained feature-space perturbations; however, insider settings impose feasibility constraints because actions must remain role-consistent, privilege-consistent, and behaviourally plausible. Accordingly, the thesis adopts threat modelling and evaluation designs that treat adversarial manipulation as a sequence of plausible behavioural perturbations rather than arbitrary noise [16], [17]. Robustness is assessed using comparative degradation analyses, attack success metrics under bounded perturbation budgets, and time-to-detection indicators where relevant.

1.5.6 Reproducibility and Reporting

To support reproducibility, the thesis reports model configurations, preprocessing assumptions, evaluation protocol details, and metric definitions. Where practical, implementation artefacts are provided in appendix material and include code structure suitable for re-execution and adaptation under appropriate ethical and governance controls. The aim is to enable transparent comparison and to ensure that the reported findings can be scrutinised and reproduced without relying on undocumented implementation choices.

1.5.7 Ethical and Governance Considerations

Although the thesis uses a benchmark dataset, insider-threat detection research has significant ethical and organisational implications. Behavioural monitoring can produce false positives, affect trust, and create risks of disproportionate intervention

if misapplied. The thesis therefore treats ethical and governance considerations as integral to methodological quality: results are framed as decision-support evidence rather than autonomous punitive judgement, validity boundaries are stated explicitly, and deployment-oriented claims are made cautiously. These considerations complement the ethics discussion in Chapter 2 and inform the interpretation of results throughout the thesis.

1.6 Contributions

This thesis seeks to contribute toward a more nuanced and operationally grounded approach to insider-threat detection. Rather than proposing a singular automated solution, the research focuses on improving the **structural alignment** between machine learning architectures and the complex reality of human behaviour in organisational settings. The contributions follow a staged progression, where each chapter explores a specific challenge identified in current literature while maintaining an honest, comparative reference through disciplined evaluation.

1.6.1 Technical and Architectural Inquiry

1. **Exploring Representation Stabilisation (Chapter 3):** This work introduces a hybrid SOM–ANN framework designed to manage the high "noise" of daily human activity. By separating initial behavioural perception from final risk inference, the architecture attempts to reduce the false alarms caused by routine changes in work patterns.

2. **Respecting Domain Semantics (Chapter 4):** Recognising that different digital actions (e.g., sending an email vs. accessing a file) carry distinct meanings, this contribution proposes a Modular ANN (MANN). This modularity is a structural attempt to ensure that subtle, meaningful signals are not "drowned out" by more frequent, routine events.

3. **Capturing Behavioural Progression (Chapter 5):** To better reflect how human risk often emerges over time rather than in a single moment, this research develops a hierarchical framework. It focuses on capturing the "trajectory" of behaviour across events and sessions, moving toward a more longitudinal understanding of user activity.

4. **Assessing Resilience Under Plausible Change (Chapter 6):** The R-Guard architecture investigates how detection systems might hold up when faced with "plausible" adversarial adaptations. This contribution provides a way to evaluate model robustness without assuming unrealistic or "perfect" attackers.

1.6.2 Methodological Honesty and Transparency

1. **A Disciplined Evaluation Protocol:** This thesis advocates for "temporally faithful" evaluation, ensuring that models are tested only on the "future" relative to their training data. This commitment to chronological validity aims to provide a more realistic—and perhaps more modest—picture of how these systems might perform in a live environment.

2. **Navigating Benchmark Limitations:** By using the CERT dataset as a consistent reference while explicitly acknowledging its gaps, the research provides a transparent comparative programme. This approach prioritises defensible interpretation over the pursuit of "perfect" benchmark scores.

1.6.3 The Human-Centric and Governance Perspective

At its core, this thesis treats insider-threat detection as a sensitive socio-technical task where model outputs directly affect individuals. Consequently, the proposed methods are framed strictly as **decision-support mechanisms** designed to assist human analysts rather than replace them with autonomous judgment. This contribution emphasises proportionality and transparency, highlighting that the "cost" of a false positive is not just a metric, but a potential human impact.

1.7 Structure

This thesis is organised as a staged investigation into behaviourally grounded and robustness-aware insider-threat detection. The chapters are structured to move from foundational concepts and methodological justification toward progressively more structured architectures, culminating in adversarially robust evaluation and synthesis. A consistent benchmark backbone and disciplined evaluation principles are maintained throughout the experimental chapters to support coherent comparison and validity-bounded interpretation.

Chapter 1 introduces the research context and motivation, defines the research problem, and presents the research questions, aims, objectives, methodological approach, and thesis contributions. It also outlines the overall structure of the thesis and the staged logic that connects the contributions across chapters.

Chapter 2 provides a critical literature review and establishes the conceptual, methodological, dataset, and ethical foundations for the thesis. It frames insider threats as socio-technical phenomena characterised by behavioural plausibility, temporal progression, and multi-domain evidence. The chapter includes contemporary case illustrations to emphasise scale and operational relevance, reviews behavioural foundations, and critiques methodological limitations in existing insider-threat detection research. It then positions the thesis within the broader AI/ML landscape,

clarifies where SOM and ANN sit in the learning-paradigm hierarchy, and reviews hybrid, modular, and hierarchical architectures as structured responses to insider detection requirements. The chapter further establishes ANNs as a unifying baseline paradigm, evaluates insider-threat datasets, and justifies the CERT dataset as the consistent benchmark used in subsequent experimental chapters with explicit validity boundaries and mitigation strategies. Finally, it consolidates ethical and governance considerations relevant to responsible insider-threat detection research and deployment.

Chapter 3 presents the first technical contribution: a SOM–ANN hybrid architecture designed to stabilise behavioural representations under high intra-user variance. The chapter details the proposed hybrid design, feature construction, temporally faithful evaluation protocol, and comparative results, establishing a stability-oriented foundation for subsequent architectural refinement.

Chapter 4 introduces the second contribution: the Modular Artificial Neural Network (MANN) framework. This chapter decomposes insider behaviour into semantically distinct behavioural domains and evaluates domain-specialised modules alongside structured evidence fusion. The chapter analyses how modularity influences generalisation, interpretability, and cross-domain coherence under consistent evaluation assumptions.

Chapter 5 presents the third contribution: a multi-layered / hierarchical behavioural modelling framework. The chapter reframes insider risk as behavioural progression across nested temporal and semantic levels and evaluates hierarchical abstraction under chronology-preserving protocols. It provides trajectory-sensitive interpretation and examines how hierarchy influences detection capability under long-horizon behavioural evolution.

Chapter 6 develops the final technical contribution: R-Guard, a behaviourally grounded robustness framework for insider-threat detection under adaptive manipulation. The chapter reviews adversarial machine learning limitations in insider contexts, formalises a behaviourally constrained adversarial threat model, defines perturbation operators, and evaluates robustness using degradation analyses and time-to-detection indicators. This chapter provides an explicit bridge between predictive modelling and robustness-aware operational validity.

Chapter 7 concludes the thesis by synthesising the contributions and findings across the staged architecture progression. It articulates the central thesis claim, summarises contributions and limitations, reflects on validity boundaries and ethical considerations, and outlines future research directions, including broader dataset validation and deployment-oriented evaluation under governed organisational settings.

Chapter 2

Literature Review

2.1 Overview

This chapter establishes the conceptual and methodological foundation for the thesis by critically reviewing insider-threat detection literature across behavioural science, organisational cybersecurity, machine learning, and adversarial reasoning. It first frames insider threats as socio-technical phenomena in which legitimate access, role context, and behavioural plausibility complicate detection, and it uses contemporary case illustrations to emphasise scale, impact, and evolving attack pathways. The chapter then identifies recurring limitations in prior work, including fragmented problem definitions, weak behavioural grounding, temporally invalid evaluation practices, inconsistent reporting, dataset artefacts, and limited robustness under behavioural drift and adaptive manipulation.

To address this fragmentation, the chapter positions the thesis within the broader AI/ML landscape and clarifies where key modelling families used in this work (notably SOM and ANN) sit within the learning-paradigm hierarchy. It reviews hybrid, modular, and hierarchical architectures as structured responses to insider detection requirements and motivates the thesis's staged progression from an ANN baseline toward SOM–ANN hybridisation, modular decomposition (MANN), hierarchical behavioural abstraction, and behaviourally grounded robustness. The chapter also evaluates insider-threat datasets, defines selection criteria, and provides a critical justification for using the CERT dataset as a consistent benchmark across the experimental chapters, with explicit validity boundaries and mitigation strategies for known evaluation pitfalls.

Finally, the chapter consolidates ethical and governance considerations, emphasising responsible interpretation, proportional monitoring, and safeguards against misuse. Collectively, the review functions both as a critical synthesis of existing knowledge and as a principled rationale for the architectural and evaluation choices developed in Chapters 3–6.

2.2 Introduction

Insider threats remain one of the most persistent and organisationally damaging forms of cyber risk because they originate *within* an organisation's trusted perimeter. Unlike external attackers, insiders act through authorised access pathways, legitimate credentials, and accumulated familiarity with workflows, systems, and security controls. Foundational work in human-centric cybersecurity has consistently shown that insider threats emerge from a socio-technical combination of psychological stressors, organisational context, behavioural drift, and technical affordances [8], [11], [15]. These interactions create behavioural trajectories that are difficult to capture through traditional security analytics or machine-learning (ML) models operating solely on technical data.

Early conceptualisations framed insider threats through misuse taxonomies distinguishing sabotage, fraud, espionage, or data theft [20], [21]. More recent studies, however, emphasise that insider risk is rarely defined by isolated malicious actions; instead, it is shaped by *progressive behavioural change* influenced by emotional states, organisational pressures, and contextual stressors [8], [12], [22]. This shift from static misuse categories toward dynamic behavioural trajectories underscores the need for ML models capable of representing evolving patterns, workload rhythms, contextual triggers, and socio-organisational dynamics.

At the same time, contemporary reporting highlights that insider risk is also shaped by third-party exposure, distributed operational ecosystems, and increasing digital complexity. Large-scale breach reporting continues to show that human and organisational factors remain central drivers of security incidents, while industry assessments emphasise the persistence of insider events in organisational risk registers [2], [3], [4]. These realities motivate the inclusion of contemporary case illustrations in this chapter to emphasise the practical scale and complexity of insider threats and to ground the modelling requirements that follow.

Despite sustained research activity and advances in anomaly detection, deep learning, hybrid pipelines, and behavioural analytics, insider-threat detection remains fragmented. Recurring weaknesses include conceptual inconsistency, narrow behavioural assumptions, dataset artefacts, temporally invalid evaluation practices, inconsistent reporting, and limited robustness under drift and adaptive manipulation [9], [10], [13], [14]. These limitations motivate the need for an integrated behavioural–ML–adversarial perspective, which this thesis develops through a staged architecture progression from baseline modelling to structured and robust detection.

This chapter develops the foundations required for the thesis contributions by establishing (i) an insider-threat framing grounded in socio-technical behavioural dynamics, (ii) a critical account of methodological and evaluation limitations in the

literature, (iii) a positioning of model families and structured architectures within the AI/ML landscape, (iv) an ANN-centred baseline and comparative reference for subsequent chapters, (v) a defensible dataset rationale centred on the CERT benchmark with explicit validity boundaries, and (vi) ethical and governance considerations for responsible insider-threat detection research.

2.3 The Insider Threat Problem

Understanding insider threats requires moving beyond the simplistic image of "trusted employees who become malicious." Contemporary research frames insider threats as a spectrum of behaviours mediated by intent, opportunity, stress, organisational context, and socio-technical affordances [8], [11]. Insider actions may be malicious, negligent, coerced, or unintentional, often emerging from psychological strain, policy misinterpretation, misaligned expectations, or weak security culture [12], [15]. Importantly, insider risk is also shaped by modern operational ecosystems—including contractors, third-party support channels, and outsourced workflows—which expands the practical definition of "insider" beyond permanent employees and increases the number of trusted access pathways that may be misused [2], [9].

Insider behaviour rarely appears as a single anomalous event. Instead, it often evolves across behavioural phases such as dissatisfaction, boundary testing, opportunity discovery, planning, and execution [22]. These trajectories unfold within constraints defined by organisational roles, privilege boundaries, workload rhythms, interpersonal dynamics, and policy enforcement. The detection challenge is therefore not simply to identify statistically rare events, but to recognise *behavioural trajectories* that remain locally plausible while gradually shifting toward harmful intent or policy violation.

From a detection standpoint, the central difficulty is that insider behaviour must remain *plausible*. Malicious insiders typically operate within authorised privileges, align actions with expected role semantics, and embed harmful steps within legitimate workflows. This makes the behavioural signal subtle and entangled with normal organisational variation, and fundamentally different from many external intrusion patterns. As a result, conventional anomaly detection methods that rely on high-magnitude deviations or signature-like indicators can struggle to capture the socio-technical cues that characterise insider escalation, particularly when malicious activity is temporally distributed and cross-domain in nature [8], [10], [15].

These properties have direct implications for ML-based detection. First, insider-threat detection is naturally multi-domain: meaningful evidence often spans authentication, file access, device usage, web activity, communications, and organisational context. Second, insider risk is temporally structured: detection requires models and evaluation protocols that respect chronology, capture long-horizon dependen-

cies, and remain interpretable under behavioural drift. Third, insider datasets are typically imbalanced and labels may be noisy or scenario-derived, demanding careful metric choice (e.g., PR-oriented evaluation), threshold calibration, and cautious interpretation of accuracy-centric claims [10], [19]. Finally, insiders may adapt strategically in response to monitoring or organisational controls, implying that nominal performance under benign evaluation protocols is insufficient without robustness-aware validation and adversary-informed reasoning [16], [17].

The next subsection (Section 2.4.6) grounds these challenges in contemporary case illustrations that highlight scale, complexity, and operational impact. This is followed by a behavioural synthesis (Section 2.4.7) that clarifies why insider risk is best modelled as a context-dependent trajectory. Together, these foundations motivate the chapter's subsequent focus on methodological limitations, structured architectures (hybrid, modular, hierarchical), and the thesis-wide emphasis on disciplined evaluation and defensible dataset selection. In particular, they justify the adoption of an ANN baseline as a consistent discriminative reference, the use of CERT as a reproducible benchmark with explicit validity boundaries, and the staged architecture progression toward stability (SOM–ANN), domain semantics (MANN), behavioural abstraction (hierarchical modelling), and robustness (R-Guard).

2.4 Insider Threat Lifecycle

Insider threats evolve through distinct phases, each characterised by specific behaviours and contextual triggers. Understanding this lifecycle is crucial for identifying relevant telemetry and aligning detection systems with the nature of insider risk at various stages. This section maps the lifecycle of insider threats to the various detection architectures developed in this thesis, particularly highlighting how each architecture targets key phases of insider behaviour: reconnaissance, misuse, and escalation.

2.4.1 Phase 1: Reconnaissance

The first phase of insider threat behaviour typically involves reconnaissance, where the insider gathers information to identify weaknesses in systems, identify valuable data, or map out potential attack vectors. This phase is characterised by exploratory actions, such as excessive access requests, browsing, and querying of sensitive data. Importantly, reconnaissance is often conducted under the guise of legitimate activity, making it challenging to detect using traditional anomaly detection methods.

Relevant Telemetry: - Login Patterns: Failed login attempts, logon times, and login frequency can indicate attempts to gain unauthorised access. - Access Patterns: Unusual access to high-value or non-work-related files. - System Queries: Unusual database queries or searches that do not align with the user's role.

Architectural Alignment: - **SOM–ANN** stabilises high-variance behaviour across time and offers a means of detecting unusual access patterns within defined bounds. However, due to the instability of early-stage exploratory behaviour, **SOM–ANN** may face difficulties in detecting subtle reconnaissance activities. - **MANN** is more adept in detecting reconnaissance by segmenting behavioural patterns into distinct domains (e.g., file access, system queries) and then identifying anomalies within these domains. - **Hierarchical Behavioural Modelling** benefits from temporal abstraction, allowing it to track exploratory behaviour over time and detect subtle shifts in behaviour that might indicate reconnaissance activities.

2.4.2 Phase 2: Misuse

Once the reconnaissance phase has been completed, an insider may begin to move into misuse of their access privileges. This phase involves more direct and deliberate violations of policy, such as data theft, sabotage, or espionage. Misuse typically involves actions like downloading sensitive data, accessing restricted files, or sending confidential information to unauthorised recipients.

Relevant Telemetry: - File Access: Large or unusual file access volumes, especially for high-value or sensitive information. - Data Transfers: Transfers to external locations or unapproved devices. - Device Usage: Using personal devices or non-standard channels to access or transfer sensitive data. - Communication Patterns: Unusual or suspicious email and chat activity involving confidential data.

Architectural Alignment: - **SOM–ANN** can detect deviations in file access patterns and file transfer behaviour. However, it requires sufficient data to reliably segment 'normal' from 'abnormal' activity, which may be hindered in early stages of misuse. - **MANN** offers a better solution by using modular decomposition to isolate misuse-related actions into specific domains (e.g., file access, data transfers), making it more effective at detecting policy violations and malicious activity within specific contexts. - **Hierarchical Modelling** is particularly well-suited to detecting misuse across multiple time-scales and across different levels of abstraction (e.g., recognising that frequent, isolated access to high-risk systems over time could indicate data theft).

2.4.3 Phase 3: Escalation

The final phase in the insider threat lifecycle is escalation, where the insider prepares to execute a more substantial attack, such as sabotage, large-scale data theft, or destruction of evidence. During this phase, insiders may attempt to escalate their privileges or pivot across systems to evade detection.

Relevant Telemetry: - Privilege Escalation: Requests to elevate access privileges or attempt to gain administrative rights. - Cross-System Behaviour: Activity

that spans across different systems or networks, indicating preparation for large-scale attacks. - Multiple Domain Cross-Interaction: Behaviour that touches multiple domains (e.g., email, file access, and database queries), often an indicator of preparation for malicious action.

Architectural Alignment: - **R-Guard** is especially effective at addressing the challenges of escalation. By enforcing behavioural invariants and monitoring cross-domain consistency, **R-Guard** helps prevent insiders from successfully escalating their attacks by detecting and flagging anomalous patterns across multiple domains. - **SOM–ANN** and **MANN** are capable of detecting escalations to some extent, but R-Guard's robustness layer significantly strengthens the overall system, ensuring that escalation events are caught before they result in major security breaches. Additionally, temporal coherence is key in this phase, and **Hierarchical Behavioural Modelling** ensures that long-term patterns are properly contextualised.

2.4.4 Linking the Lifecycle to Model Design

The models developed in this thesis each offer a contribution that addresses a specific stage of the insider threat lifecycle:

- **SOM–ANN** provides behavioural stabilisation and noise reduction, which are essential in detecting reconnaissance and early-stage misuse.
- **MANN** offers domain-aware decomposition, improving detection sensitivity and reducing cross-domain bias in insider behaviour during the misuse and escalation stages.
- **Hierarchical Modelling** adds temporal reasoning, strengthening the detection of behavioural progression across time and across different threat phases.
- **R-Guard** integrates robustness through behavioural invariants, providing resilience against escalation and adversarial manipulation.

In combination, these models offer a comprehensive solution to detecting insider threats across their lifecycle, from initial reconnaissance to the execution of malicious actions.

2.4.5 Summary

In summary, this section has mapped the insider threat lifecycle to the models and architectures proposed in this thesis. By framing insider threat detection in terms of behavioural phases and telemetry alignment, this chapter establishes a foundation for understanding how each architecture targets specific stages of insider behaviour.

Through modular decomposition, hierarchical abstraction, and adversarial robustness, the models developed here address critical weaknesses identified in prior work and provide a more holistic solution to detecting insider threats across time and space.

2.4.6 Contemporary Insider-Threat Case Illustrations

Recent reporting and industry assessments continue to emphasise that insider risk is shaped not only by malicious intent, but also by operational pressures, third-party exposure, and the increasing complexity of modern digital ecosystems [2], [3]. In addition, industry surveys indicate that many organisations report experiencing insider incidents annually, reinforcing the continued operational relevance of insider-threat research [4].

To emphasise the contemporary nature of insider threats and motivate the modelling requirements developed in this thesis, this subsection provides brief case illustrations that highlight scale, complexity, and practical detection challenges. The purpose is not to provide an exhaustive historical catalogue, but to foreground recurring patterns that directly motivate the thesis architecture progression: multi-log evidence integration, temporal progression modelling, interpretability, and robustness under adaptive behaviour.

2.4.6.1 Case 1: High-Value Intellectual Property Exfiltration

A recurring insider-threat pattern involves the covert extraction of high-value intellectual property (IP) by a trusted employee with legitimate access to internal repositories and development workflows. Publicly documented prosecutions illustrate how exfiltration can occur through authorised access pathways and may be staged over time, thereby reducing the effectiveness of purely rule-based controls that assume malicious behaviour must appear as overt access violations [23]. Such cases highlight a key modelling requirement for insider-threat detection: identifying subtle deviations in access trajectories and cross-domain behaviour (e.g., authentication patterns aligned with unusual data access or movement), rather than relying solely on high-magnitude outliers.

2.4.6.2 Case 2: Disgruntled Employees

Another commonly reported pattern involves insiders who leak sensitive information during employment transitions, disciplinary processes, or post-termination conflict. Reported incidents have linked large-scale data exposures to former employees and emphasised the operational consequences of insider-enabled disclosure [24]. For behavioural modelling, this pattern is important because it often occurs during identifiable organisational windows (e.g., offboarding) and may combine normal work

activity with targeted misuse. This motivates temporally faithful evaluation (to avoid leakage and inflated performance), and threshold calibration strategies that reflect the operational trade-off between early detection and false-positive control under legitimate role change.

2.4.6.3 Case 3: Insider–Outsider Collusion

Recent disclosures also highlight insider risk arising from collusion, particularly in environments where support operations have privileged access to sensitive customer systems. In 2025, a widely discussed case described external actors bribing and recruiting support personnel to access customer information, enabling downstream social engineering and fraud [25], [26]. This class of incident demonstrates that insider threats are not limited to core employees and may arise through extended organisational perimeters, contractor ecosystems, and customer-support tooling. It also illustrates that behavioural signatures of misuse may be low-amplitude at the event level (e.g., repeated legitimate-looking support queries) but high-impact at the outcome level. For ML design, this motivates multi-domain evidence fusion and behavioural consistency reasoning, since single-log anomaly measures can struggle in high-volume support environments.

2.4.6.4 Case 4: Sabotage

Insider threats may also involve sabotage and destructive actions, including deletion of systems, corruption of data, or disruption of operational services. Public reporting in 2025 described an incident involving destructive actions against databases and associated data assets in a government-contractor context, with significant operational consequences [27], [28]. These cases highlight that insider threats can produce rapid high-impact outcomes (burstiness), but may still be preceded by smaller behavioural precursors such as changes in access frequency, unusual sequence composition, privilege-use patterns, or deviations in time-of-day activity. For modelling, sabotage scenarios motivate time-sensitive detection measures (including time-to-detection) and robustness evaluation under behavioural drift or stressor-driven change.
Note: These figures are reported estimates from a recurring survey series; year-to-year comparability may be influenced by sampling and methodological updates, so the trend is interpreted as indicative rather than definitive.

2.4.6.5 Synthesis: Implications for Behavioural Modelling

Across these illustrations, three thesis-critical insights emerge. First, insider threats are rarely expressed as isolated anomalies; they often emerge as behaviourally plausible trajectories distributed across time and organisational context. Second, evidence is frequently multi-domain (e.g., authentication, file actions, device use, communications, and support tooling), meaning that robust detection requires integration

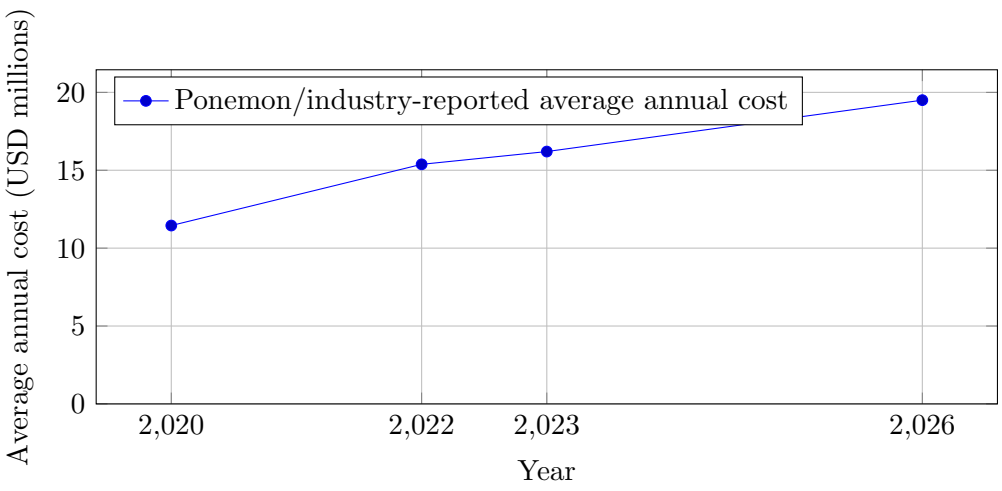


Figure 2.1: Reported average annual organisational cost associated with insider threat and insider-risk incidents across selected Ponemon Institute reporting years. Reported values are $11.45 million in 2020, $15.38 million in 2022, $16.2 million in 2023, and $19.5 million in 2026 [5], [6], [7], [29]. *Source: Author's visualisation based on the cited industry reports.*

across heterogeneous logs rather than reliance on single-stream features. Third, many incidents involve adaptive behaviour—whether gradual exfiltration, collusion that mimics legitimate workflows, or destructive actions triggered by contextual stressors—which implies that high nominal accuracy on benign evaluation protocols does not guarantee resilience under realistic insider adaptation [2], [9], [10].

These observations motivate the modelling requirements and architecture progression developed in this thesis. A well-defined ANN baseline establishes nonlinear discriminative capability; SOM–ANN hybridisation supports representation stabilisation and behavioural structuring; modular architectures preserve domain semantics and reduce feature dominance across heterogeneous logs; hierarchical modelling captures multi-level behavioural progression over long horizons; and behaviourally grounded robustness mechanisms address adaptive insider behaviour under constrained perturbations.

2.4.7 The Nature of Insider Behaviour

Behavioural cybersecurity research consistently shows that insider threats cannot be understood through system logs alone. Insider behaviour arises from the interplay of cognitive and affective factors (e.g., motivation, stress, impulsivity, perceived injustice), organisational structures (e.g., team dynamics, workplace climate, supervision and enforcement norms), contextual conditions (e.g., deadlines, workload surges, role change), and technical affordances (e.g., privilege breadth, system exposure, data accessibility) [8], [11], [12]. The key implication is that insider risk is not merely an outcome of access, but an emergent property of how access is exercised under evolving socio-organisational conditions.

Table 2.1: Contemporary insider-threat case patterns: indicative signals, modelling implications, and links to thesis contributions

Case pattern	**Indicative signals (multi-log)**	**Key modelling implications**	**Thesis linkage**
IP/data exfiltration by trusted insider	Elevated access to sensitive repositories; unusual file reads/exports; atypical device usage; abnormal timing; cross-domain coupling of logon and file activity	Requires non-linear discrimination and cross-domain evidence fusion; benefits from stable behavioural representation and interpretability to distinguish legitimate work from covert extraction	ANN baseline (Ch. 2); SOM–ANN stability (Ch. 3); MANN domain semantics (Ch. 4)
Disgruntled employee leakage	Behaviour change around offboarding windows; increased copying/export; spikes in access outside normal hours; shifts in device/media usage	Motivates temporally faithful evaluation, drift-aware interpretation, and threshold calibration to control false positives under legitimate role transition	Chronology-preserving evaluation (Ch. 3–6); hierarchical progression (Ch. 5)
Insider–outsider collusion (bribed support channels)	Legitimate-looking support queries; repeated access to customer records; unusual query sequences; social engineering downstream indicators	Highlights difficulty of separating authorised access from misuse; requires multi-log correlation and behavioural consistency checks across channels	MANN fusion logic (Ch. 4); invariant-s/consistency (Ch. 6)
Sabotage / destructive insider actions	Privilege-use anomalies; rapid deletion/alteration; unusual admin operations; abrupt activity bursts preceded by subtle precursors	Requires time-sensitive detection (time-to-detection) and robustness under behavioural stress; supports the need for early-warning drift indicators	Hierarchical modelling of trajectories (Ch. 5); robustness and time-to-detection (Ch. 6)

Machine-learning models typically encode behaviour as numerical features, sequences, or embeddings. While these representations capture technical activity signatures, they often omit the socio-technical grounding needed for reliable interpretation in organisational settings. In practice, this omission contributes to two recurring failure modes. First, models may overreact to benign deviations (e.g., legitimate overtime, exceptional projects, temporary role coverage), producing inflated false-positive rates and degrading analyst trust. Second, models may underdetect privilege-consistent misuse, where malicious actions remain locally plausible because they are aligned with role semantics and embedded in legitimate workflows [9], [13]. Systematic reviews emphasise that without behavioural context and evaluation discipline, reported accuracy can be misleading and sensitivity to subtle insider behaviours may be weak, particularly when malicious activity is temporally distributed or cross-domain in nature [8], [10].

Figure 2.2 summarises a conceptual taxonomy that integrates behavioural, motivational, and operational dimensions. This framing is used in the thesis not as a static categorisation scheme, but as a reminder that insider risk spans multiple insider archetypes (malicious, negligent, compromised/coerced) and that each archetype may produce different observable traces and different detection trade-offs (e.g., false-positive tolerance versus early-warning sensitivity).

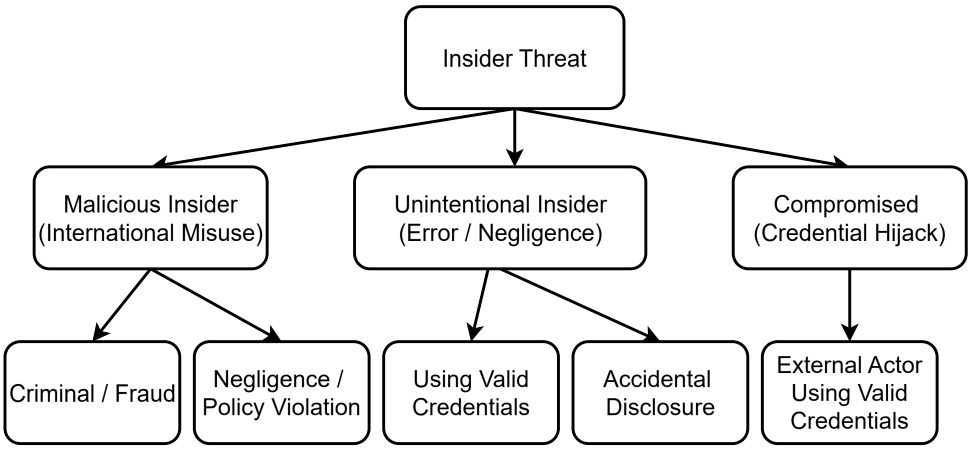


Figure 2.2: Conceptual taxonomy of insider threats, distinguishing malicious, negligent and compromised insiders across behavioural, motivational and operational dimensions. *Source: Author's own conceptualisation based on the literature reviewed in this chapter.*

In synthesis, insider behaviour is best understood as a context-dependent trajectory rather than a single anomalous event. The contemporary case illustrations in Section 2.4.6 reinforce that insider incidents often unfold gradually, combine actions across multiple organisational systems, and remain superficially plausible due to legitimate access and role alignment. These properties motivate the thesis emphasis on temporally faithful evaluation, multi-log modelling, and structured architectures capable of representing behavioural progression and cross-domain dependencies, while remaining resilient to adaptive manipulation and behavioural drift.

This behavioural framing also provides the conceptual bridge to the methodological limitations reviewed next and the structured ML responses (hybrid, modular, and hierarchical) developed throughout the thesis.

2.4.8 Epistemological Positioning

Insider-threat research has historically developed along two largely independent epistemological trajectories. The behavioural–organisational tradition conceptualises insider threats as emergent phenomena arising from individual psychology, organisational context and socio-technical interaction patterns [11], [15]. In contrast, the machine-learning tradition approaches insider detection as a pattern-recognition task driven by statistical inference, anomaly scoring, deep representation learning and log analysis [30], [31].

While each tradition provides valuable insights, both are incomplete in isolation. Behavioural models offer explanatory richness but lack scalability, whereas ML models provide scalability but lack semantic grounding. To bridge this divide, this thesis adopts a hybrid epistemological stance combining behavioural science, organisational theory, representation learning and adversarial modelling. The core claim is that insider threat detection is fundamentally a socio-technical machine-learning problem: behavioural trajectories must be computationally represented, and ML systems must incorporate semantic, contextual and adversarial reasoning to interpret deviations meaningfully.

This epistemological position shapes the structure of the literature review: Section 2.5 examines behavioural and organisational foundations; Section 2.6 critiques dataset validity and methodological limitations; Section 2.8 evaluates hybrid, modular and hierarchical ML architectures; Section 2.9 analyses relational and GNN misalignment; Section 2.10 introduces a formal behavioural adversarial framework; and Section 2.11 synthesises the unresolved gaps motivating the thesis contributions.

This hybrid epistemological scaffolding enables a multi-level understanding of insider threats and provides the conceptual foundation for the adversarially robust behavioural architectures developed in Chapters 3–6.

2.5 Behavioural Foundations

Insider threat is not fundamentally a technical anomaly problem; it is a behavioural phenomenon manifesting through socio-technical interaction within organisational systems. Human behaviour emerges from cognitive, emotional, organisational and contextual forces that shape how individuals interpret, respond to and navigate their working environments. Behavioural and organisational cybersecurity research emphasises that insider actions cannot be decoupled from the psychological, situational and structural factors that give rise to them [11], [12], [15]. These behavioural

foundations are essential for interpreting observable system activity and form the conceptual grounding required for machine-learning models to operate meaningfully and robustly.

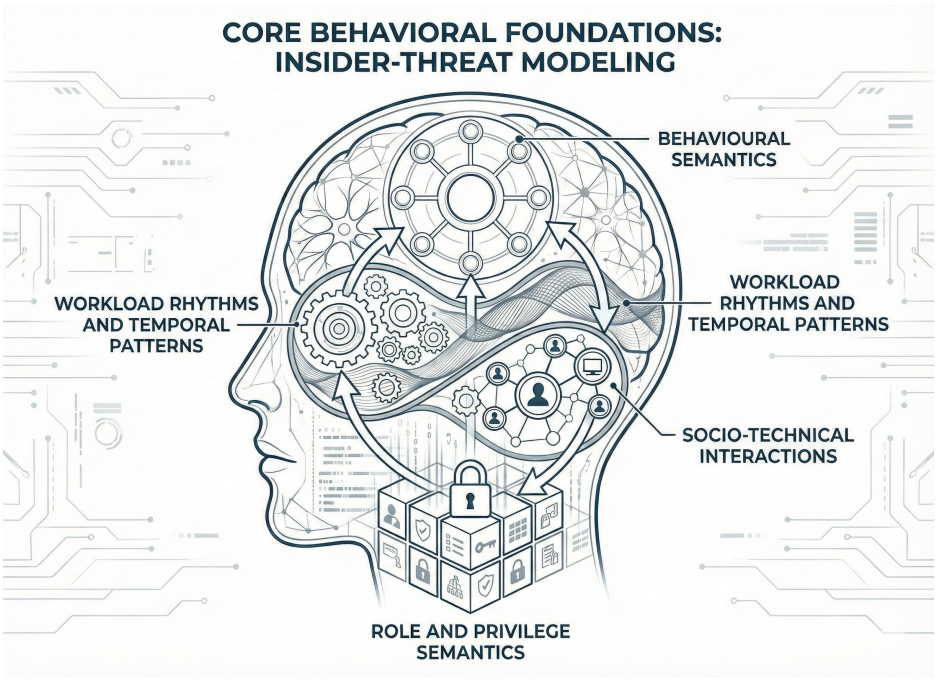


Figure 2.3: Overview of core behavioural foundations relevant to insider-threat modelling: role and privilege semantics, workload rhythms and temporal patterns, socio-technical interactions, and behavioural semantics. *Source: Author's conceptual synthesis of the literature reviewed in this chapter.*

The discussion that follows synthesises key behavioural constructs underpinning insider-risk development and demonstrates how these constructs reveal structural blind spots in contemporary ML-based detection systems. These foundations directly motivate the need for multi-level, context-aware and adversarially robust models, as developed in later chapters.

2.5.1 Human Behaviour and Insider Risk

Human behaviour in organisations is shaped by motivation, cognitive load, emotional state, fatigue, perceived injustice and situational stressors. Behavioural studies show that harmful insider actions typically emerge through precursors—such as dissatisfaction, declining performance, unusual communication patterns or deviation from routine habits—rather than appearing abruptly [15], [22]. Yet such precursors may be subtle, gradual and context-dependent. Not all anomalies reflect malicious intent; conversely, many harmful actions appear statistically normal.

Detected behaviour is also heterogeneous: what appears anomalous for one role may be routine for another. A data engineer transferring gigabytes of logs is normal; for a compliance officer, it may be cause for concern. This role-dependent heterogeneity limits the effectiveness of global anomaly thresholds and highlights the import-

ance of behavioural semantics and contextual interpretation [8]. Machine-learning systems that treat behaviour as independent events—ignoring behavioural histories and trajectories—struggle to capture the gradual, psychologically driven escalation that characterises many insider incidents.

2.5.2 Role and Privilege Semantics

Organisational structure shapes what behaviour is expected, permissible or suspicious for a given user. Role semantics encode behavioural norms, while privilege assignments define which actions a user is authorised to perform. Research in organisational cybersecurity consistently shows that insider detection must be evaluated relative to a user's job function, responsibilities and workflow context [8], [21].

The same observable action can have different meanings depending on who performs it and under what conditions. Logging in at night may be atypical for a junior clerk but entirely normal for system administrators during maintenance periods. Large file transfers may indicate exfiltration for some roles yet be routine for others. Machine-learning systems lacking explicit role and privilege semantics misinterpret contextual behaviour, leading to excessive false positives and systematic blind spots in privilege-consistent malicious behaviour [13].

Figure 2.4 illustrates how insider behaviour occupies a behavioural manifold where privilege level, organisational role and behavioural deviation intersect. Movement within this manifold may signal legitimate change (e.g., promotion, project rotation) or early-stage risk. Distinguishing between the two requires behavioural awareness and adaptive modelling, particularly in the presence of concept drift [32], [33].

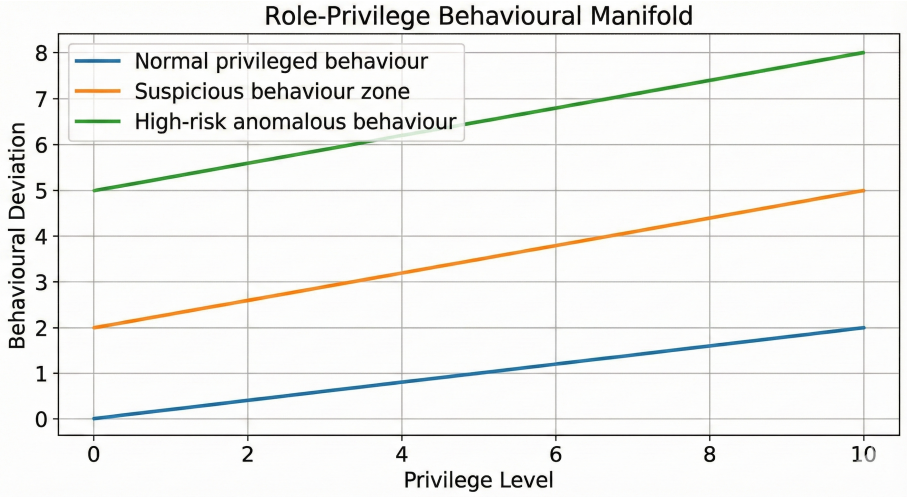


Figure 2.4: Conceptual behavioural manifold illustrating how privilege assignments and role semantics shape the interpretation of behavioural deviation.

2.5.3 Workload Rhythms

Work activity is inherently structured by temporal rhythms—daily routines, weekly cycles, monthly reporting periods and seasonal patterns. Organisational ethnography and behavioural psychology show that these temporal regularities strongly influence behavioural expectations [8]. Patterns such as morning logins, midday inactivity, end-of-day reporting, financial-quarter deadlines and periodic administrative cycles define a user's behavioural cadence.

Yet many ML-based insider-threat systems employ static or arbitrarily sized temporal windows that ignore such rhythms. This disconnect enables timing-based adversarial behaviour, where malicious actions are embedded within workload peaks or high-volume system events, reducing their anomaly scores—a vulnerability documented in recent adversarial temporal-model studies [34], [35]. Temporal semantics must therefore form a foundational component of behavioural modelling, capturing both short-term patterns and long-range behavioural evolution.

2.5.4 Socio-Technical Interactions

Insider behaviour emerges from interactions between humans and the technical systems they use. Socio-technical theory emphasises that system affordances, interface usability, automation scripts, monitoring practices and organisational workflows shape behavioural patterns. Users adapt behaviour in response to system design, often creating informal workarounds when policies or tools impede productivity. Such adaptive behaviours may appear anomalous but are not malicious [12].

Conversely, high monitoring pressure can cause behavioural suppression or deliberate reshaping to avoid triggering alerts. These human–machine interaction dynamics create feedback loops (Figure 2.5) that influence observable activity but are opaque to purely technical ML models. Behavioural irregularities arising from socio-technical friction differ fundamentally from those emerging from malicious intent, and ML models require this contextual grounding to separate the two.

2.5.5 Behavioural Semantics and ML Limitations

Traditional ML approaches assume that behavioural meaning can be inferred from statistical deviation, embedding distance or sequence irregularities. Yet behavioural semantics—the meaning, intent and contextual interpretation of an action—cannot be inferred from low-level system events alone. Three structural limitations are consistently observed across insider-threat ML research:

Semantic blind spots. Privilege-consistent malicious actions often appear statistically normal and evade detection when models lack role or workflow semantics [13].

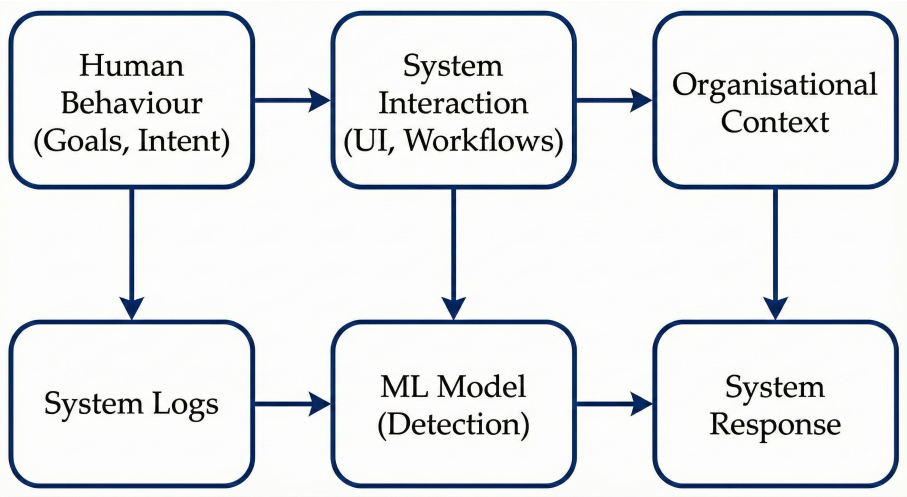


Figure 2.5: Socio-technical human–machine interaction loop illustrating how human intent, organisational context and system responses shape observable behaviour.

Identity–behaviour conflation. Deep and statistical models frequently cluster users by identity rather than behavioural meaning, a pitfall repeatedly documented in systematic reviews [8].

Lack of organisational interpretability. Without behavioural and contextual grounding, models produce anomalous scores devoid of operational meaning. This undermines trust, interpretability and incident-response relevance.

Table 2.2 summarises key behavioural constructs, their relevance to ML modelling and common failure modes. These limitations demonstrate why ML-only pipelines consistently struggle with insider threats.

2.5.6 Behavioural Requirements for ML Architectures

The behavioural insights outlined above establish critical requirements for ML architectures intended to model insider threats. Effective detection requires hierarchical interpretation across event-, session-, task- and role-level contexts; temporal sensitivity to behavioural trajectories and organisational rhythms; semantic modelling of privilege, workflow and intent; adaptive responses to legitimate drift; and robustness against behaviourally feasible manipulation strategies (Section 2.10). These requirements are absent in much of the existing literature and underpin many of the methodological weaknesses examined in Section 2.6.

Behaviourally grounded modelling therefore forms the conceptual basis for the hybrid, hierarchical and adversarially robust frameworks developed in subsequent chapters. The next section reviews the methodological limitations that have prevented the field from achieving reliable, generalisable and behaviourally aligned insider-threat detection.

Table 2.2: Behavioural constructs, their characteristics and implications for insider-threat machine learning.

Behavioural Construct	Description	ML Relevance	Typical Failure Modes
Role Semantics	Expected behaviour defined by responsibilities, privilege boundaries and organisational constraints.	Establishes behavioural baselines.	Role change misclassified; privilege-consistent misuse undetected.
Temporal Patterns	Cyclical work rhythms and activity sequences.	Essential for time-series and sequence models.	Long-range semantics lost; drift ignored.
Socio-Technical Interactions	Behaviour shaped by system affordances and workflows.	Supports context-aware modelling.	Models assume feature independence; context omitted.
Behavioural Semantics	Meaning and intent behind actions.	Required for semantic-rich detection.	ML misinterprets meaning; maps surface-level anomalies only.

2.6 Methodological Limitations

Despite considerable progress in applying machine learning to insider-threat detection, the methodological foundations of the field remain fragile. Many widely cited studies rely on synthetic or weakly realistic datasets, opaque feature-engineering pipelines, flawed evaluation protocols and assumptions that fail under temporal, cross-user or adversarial scrutiny. These weaknesses undermine validity, reproducibility and operational relevance, and they help to explain the persistent gap between strong reported performance and limited real-world success.

This section synthesises the main methodological limitations in insider-threat ML, drawing on critical analyses of dataset realism [13], concept drift [32], [33], adversarial ML [34], [36] and behavioural cybersecurity research [8], [11]. The discussion focuses on dataset realism, feature semantics, temporal structure, evaluation methodology, reproducibility, identity–behaviour confounding, data leakage and the neglect of drift and behavioural evolution.

2.6.1 Dataset Realism

Insider-threat detection research is heavily dependent on a small number of public datasets, most notably the synthetic CERT datasets. While these corpora provide

structured logs for experimentation, detailed analyses show that they encode over-simplified behavioural scripts, deterministic malicious scenarios and weak organisational context [13]. As a result, many models trained on CERT learn artefacts of scenario design rather than robust indicators of insider behaviour.

Synthetic users in CERT follow constrained and repetitive patterns with limited variability, which artificially inflates the separability between benign and malicious activity. Malicious actions are often injected as abrupt, temporally localised deviations, making them easy targets for even simple anomaly detectors. Users are only weakly differentiated by role, team or privilege structure; project cycles, realistic workload rhythms and team interactions are largely absent. Ground truth labels are also synthetic, reflecting scenario scripts rather than organically emerging socio-technical incidents.

Similar limitations affect other datasets, including LANL [31], ADFA-LD [35] and RIT-IDEAL [31]. Although they improve scale or diversity, none fully captures the behavioural and organisational complexity described in Section 2.5. Consequently, the literature often optimises for performance on narrowly defined benchmarks with limited ecological validity, rather than for robustness in real organisational environments.

2.6.2 Feature Engineering Limits

Feature engineering remains central in insider-threat ML, but many pipelines inadvertently destroy behavioural meaning or encode spurious artefacts. Common transformations such as one-hot encoding, token indexing, frequency counts and aggregated statistics compress rich behavioural sequences into coarse numerical representations.

This leads to three structural problems. First, behavioural meaning is lost: actions such as accessing a critical system, exfiltrating sensitive files or making unusual communications are reduced to feature vectors that omit context, intent and privilege semantics. Second, identity artefacts are encoded directly into features. One-hot encodings and categorical indices make it easy for models to learn user-ID signatures instead of behaviour, a problem exacerbated under class imbalance and limited user diversity [13]. Third, over-reliance on frequency-based statistics biases models toward "busy" users, while subtle malicious activity embedded within high-volume legitimate behaviour remains under-emphasised.

These distortions weaken both interpretability and generalisability. Without behaviourally meaningful features, even sophisticated models struggle to transfer across users, roles, projects or organisations.

2.6.3 Temporal and Context Misalignment

Temporal structure is a defining feature of insider behaviour, yet many studies treat logs as temporally weak or even time-agnostic. Fixed-size sliding windows, shuffled sequences and coarse aggregation strategies fragment behavioural trajectories and suppress long-range dependencies. This makes it difficult to detect gradual escalation, workload-aligned timing manipulation or multi-stage insider campaigns.

Window-based fragmentation breaks natural behaviour into arbitrary segments, obscuring the continuity of sessions, tasks and workflows. Cyclical patterns—diurnal, weekly or monthly rhythms—are often ignored, despite evidence that such periodicity underpins behavioural normality [8]. Few models explicitly integrate multi-scale temporal reasoning, in which short-term event sequences are interpreted in the context of longer-term behavioural evolution. Legitimate behavioural drift—caused by promotions, project changes or new responsibilities—is rarely modelled, despite well-established evidence that static classifiers perform poorly in non-stationary domains [32], [33].

The result is temporal misalignment: legitimate changes are misclassified as anomalies, while adversaries exploit unmodelled structure by embedding malicious activity within workload peaks or unobserved intervals [17], [34].

2.6.4 Evaluation Flaws

Evaluation methodology in insider-threat ML is frequently optimistic and sometimes fundamentally flawed. A recurring issue is the use of train/test splits that ignore temporal order, allowing future information to leak into training. When models are evaluated on data reshuffled across time, reported performance no longer reflects realistic deployment scenarios.

Other flaws include inadequate treatment of class imbalance, where accuracy and ROC-AUC are reported without appropriate baselines or cost considerations; unrealistic sampling strategies where malicious events are oversampled or artificially balanced; and evaluation protocols that do not test generalisation to unseen users or roles. Robustness to drift or adversarial manipulation is rarely assessed, despite its central importance in realistic insider settings.

Critical evaluations have shown that many deep-learning approaches degrade substantially when evaluated under temporally faithful splits or when tested on previously unseen users [13]. Without rigorous, behaviourally aligned evaluation, claims of superiority for particular architectures or techniques remain weakly substantiated.

2.6.5 Reproducibility Issues

Reproducibility remains a substantial challenge in insider-threat research. Complex pipelines spanning log preprocessing, feature extraction, model training and

evaluation are rarely documented in full. Key details such as hyperparameters, normalisation strategies, window sizes, label construction rules, random seeds and code versions are often omitted or only partially described. In some cases, datasets themselves are restricted, further limiting independent verification.

The lack of experimental transparency makes it difficult to reproduce reported results, compare methods fairly or build cumulative scientific knowledge. This is particularly problematic in security ML, where small methodological differences can produce large performance swings. Without reproducible baselines, it is difficult to distinguish genuine architectural advances from artefacts of experimental setup.

2.6.6 Identity–Behaviour Confounding

A pervasive methodological weakness is the confounding of identity and behaviour. Many models, especially deep architectures and embedding-based methods, inadvertently learn to recognise users rather than to interpret behaviour. This occurs when features, sequences or graphs encode stable user-specific patterns that dominate the learned representation [8], [13].

When identity and behaviour are entangled, models can achieve apparently high performance metrics while failing to detect truly malicious change. Malicious actions that superficially resemble a user's historical profile may be misclassified as benign, while benign deviations by high-variance users are over-reported. Cross-user generalisation deteriorates because the model effectively memorises users rather than learning transferable behavioural rules. This confounding is especially acute when one-hot encodings, user IDs or stable structural positions in graphs are used as input features.

2.6.7 Data Leakage

Data leakage is one of the most serious and under-recognised threats to methodological validity in insider-threat ML. Leakage arises when information from outside the legitimate training scope influences learning or evaluation—often inadvertently through preprocessing steps, splitting strategies or feature construction.

Table 2.3 summarises the main leakage types observed in the literature. Temporal leakage occurs when future events influence current training examples, for example when windows are constructed over full sequences before splitting. Identity leakage arises when user identity is directly encoded in features or indirectly preserved in embeddings. Contextual leakage appears when timestamp patterns or synthetic injection rules inadvertently reveal ground truth. Pipeline leakage emerges when normalisation, feature selection or dimensionality reduction is performed on the full dataset prior to splitting.

Leakage can be subtle and difficult to detect, especially in multi-stage pipelines.

Table 2.3: Taxonomy of data leakage in insider-threat detection research.

Leakage Type	Definition	Leakage Channel	Consequences
Temporal Leakage	Future events incorporated into training or feature extraction.	Non-temporal splits; window aggregation over full sequences.	Inflated accuracy and unrealistic predictive foresight.
Identity Leakage	Identity encoded in features or embeddings.	User IDs; one-hot encodings; stable graph positions.	Models learn identity rather than behaviour; poor cross-user generalisation.
Contextual Leakage	Contextual artefacts that implicitly reveal labels.	Injection scripts; timestamp regularities; scenario markers.	Trivial detection of synthetic anomalies; over-optimistic results.
Pipeline Leakage	Cross-stage contamination during preprocessing or feature selection.	Normalisation and feature selection performed on the full dataset.	Severe overfitting and exaggerated performance claims.

Its presence severely distorts evaluation, producing performance estimates that collapse when models are deployed or re-implemented under stricter protocols.

2.6.8 Concept Drift and Behavioural Evolution

Insider-threat environments are non-stationary. User roles change, projects start and end, policies evolve, technologies are introduced and broader organisational shifts reshape behaviour. Concept drift research demonstrates that static models deployed in such environments will progressively misalign with the underlying data-generating process [32], [33].

In practice, many insider-threat ML systems either assume stationarity or rely on occasional retraining without systematic drift detection or management. Legitimate behavioural evolution is then misclassified as anomalous, while adversaries exploit drift blind spots by gradually adjusting behaviour to evade detection. Over time, models that were tuned to historical distributions become unreliable, with increasing false positives and missed detections.

Neglect of drift is particularly dangerous in the adversarial setting formalised in Section 2.10, where gradual shaping of behaviour is itself a primary evasion strategy. Without drift-aware and behaviourally grounded adaptation, long-term deployability remains questionable.

2.6.9 Methodological Bottlenecks

Taken together, these methodological limitations constitute a core bottleneck for insider-threat ML. Unrealistic datasets, behaviourally weak feature engineering, temporal misalignment, optimistic evaluation, poor reproducibility, identity–behaviour confounding, data leakage and drift neglect collectively undermine the credibility of reported results and the robustness of deployed systems.

Table 2.4: Summary of methodological limitations in insider-threat machine learning and their consequences for validity, robustness and operational deployment.

Limitation	Description	Consequence for ML Models	Implication for Research
Data Leakage	Overlapping users, sessions or temporal segments during training, validation or testing.	Inflated accuracy; misleading predictive power.	Generalisation and operational readiness overstated.
Unrealistic Sampling	Artificial class balancing or oversimplified dataset splits.	Overfitting to narrow behavioural distributions.	Poor transferability to heterogeneous organisations.
Temporal Mixing	Loss of chronological ordering through shuffling or window collapse.	Sequence semantics destroyed; degraded temporal modelling.	Models unsuitable for real-time insider detection.
Concept Drift Ignored	Assumption of static user behaviour or static threat landscape.	Rapid performance decay under behavioural evolution.	Long-term deployment becomes infeasible.
Label Noise / Ambiguity	Weak, inconsistent or retrospective labelling practices.	Incorrect pattern learning; unstable model training.	Higher false positives and false negatives.
Adversarial Blindness	Models ignore constraint-respecting insider manipulations.	Evasion via drift, timing and sequence perturbations.	Increased deployment risk without adversarial robustness.

Bridge to architectural responses. The methodological limitations identified in Section 2.6 do not merely reflect implementation errors; they indicate structural misalignment between common machine-learning assumptions and the sociotechnical nature of insider behaviour. In particular, the literature reveals a repeated tension between (i) behavioural realism and temporal validity, (ii) heterogeneous multi-log evidence and single-vector modelling, and (iii) high nominal accuracy and

limited robustness under drift or adaptive behaviour. These limitations motivate a progression toward more structured learning designs that explicitly encode representation stability, domain semantics, hierarchical behavioural abstraction, and robustness constraints. The next section therefore positions the thesis within the broader machine-learning landscape and clarifies where the chosen modelling families (including SOM and ANN) sit within that hierarchy before introducing hybrid, modular, and hierarchical architectural responses.

2.7 Machine Learning Foundations

This section provides a focused bridge between the methodological limitations identified in Section 2.6 and the architecture-centric literature reviewed in the following section. The purpose is not to provide a general machine-learning tutorial, but to clarify (i) where the modelling families used in this thesis sit within the wider AI/ML landscape, (ii) how model choices relate to the requirements of insider-threat detection, and (iii) why increasingly structured architectures (hybrid, modular, hierarchical, and robust) are a principled response to the limitations observed in the literature. This framing also establishes a shared vocabulary for interpreting the ANN baseline section and the architecture progression developed in Chapters 3–6 [9], [10], [14].

2.7.1 AI and Machine Learning

Machine learning (ML) can be viewed as a subset of artificial intelligence (AI) concerned with learning patterns from data to support prediction, classification, clustering, or decision-making. Within ML, learning paradigms are commonly grouped into supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. Although these categories are standard, their implications in insider-threat detection are domain-specific because insider data are behavioural, temporally structured, and typically imbalanced, with labels that may be incomplete, ambiguous, or scenario-derived [8], [10], [15], [22].

Supervised approaches are widely used when labelled data are available and when the objective is to distinguish malicious or policy-violating behaviour from benign activity. Unsupervised approaches are often adopted when labels are scarce or unreliable and when the objective is anomaly detection or behavioural clustering. Semi-supervised approaches attempt to leverage limited labelled data alongside a larger pool of unlabelled behavioural traces. Reinforcement-learning approaches are less common in insider-threat detection but may be relevant to adaptive policy control, response optimisation, or attacker–defender interaction modelling [9], [14].

Within this thesis, supervised and unsupervised paradigms are both central: the SOM component provides unsupervised topological structuring and behavioural

representation stabilisation, while ANN-based components provide supervised discriminative modelling and multi-stage architectural learning. This combination is aligned with the socio-technical challenge identified earlier: insider threats require both behavioural structure discovery and risk-sensitive classification under operational constraints.

2.7.2 Where SOM and ANN Sit Within the ML Hierarchy

Figure 2.6 provides an architectural positioning view that locates the modelling families used in this thesis within the broader AI/ML landscape. This hierarchy is used here for conceptual clarity and to establish an explicit rationale for hybridisation. Self-Organising Maps (SOMs) are placed within unsupervised learning for clustering and topology-preserving representation, while Artificial Neural Networks (ANNs) are positioned within supervised learning and deep learning as the primary baseline family for non-linear discriminative modelling [37], [38].

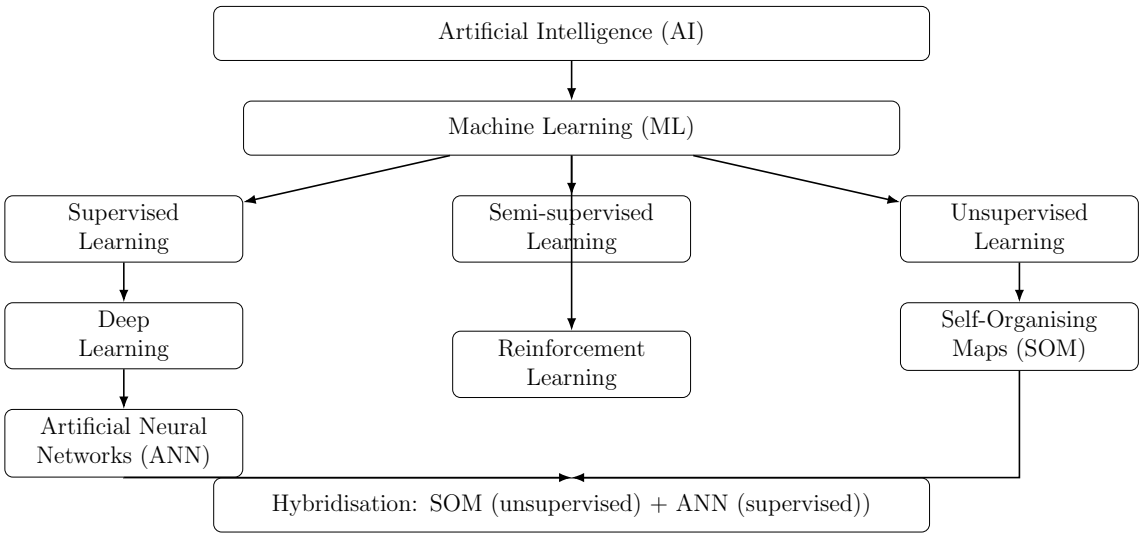


Figure 2.6: Positioning of SOM and ANN within the AI/ML hierarchy. SOMs support unsupervised topology-preserving behavioural structuring, while ANNs provide a supervised deep-learning baseline for non-linear discrimination. Their hybridisation motivates the SOM–ANN contribution and supports subsequent modular, hierarchical, and robust extensions.

This positioning is important for insider-threat detection because it clarifies the complementary roles of structure discovery and discriminative learning. In the insider domain, unsupervised models can capture behavioural manifolds and discover latent structure in multi-log data, while supervised models can provide operationally useful risk classification under controlled labels. Hybrid architectures aim to combine these benefits while mitigating the limitations of either paradigm used alone [10], [14], [30].

2.7.3 Algorithm Families

Although insider-threat detection has unique socio-technical constraints, model choice can be contextualised by considering how ML families are typically selected across application domains. Table 2.5 provides a compact cross-domain view that highlights common algorithm families and tasks in business, finance, gaming, and cybersecurity. The purpose is not to suggest that insider detection should mirror other domains, but to provide interpretive grounding for why ANN-based modelling (and structured ANN extensions) is a reasonable choice for complex behavioural telemetry that is heterogeneous, temporal, and non-linear [9], [14], [38].

Table 2.5: Common machine-learning tasks and algorithm families across domains, with implications for insider-threat detection

Domain	**Common tasks**	**Common ML families**	**Implications for this thesis**
Business / operations	Classification, forecasting, segmentation, process optimisation	Linear models, tree ensembles, clustering, shallow and deep neural networks	Insider-threat detection shares heterogeneity and noisy observational data; motivates robust feature integration and non-linear modelling (ANN baseline)
Finance	Fraud detection, anomaly detection, risk scoring, time-series prediction	Tree ensembles, anomaly detection, sequence models (RNN/LSTM), deep learning	Similar imbalance and rare-event detection; supports PR-focused evaluation and sequence-aware modelling under temporal validity constraints
Gaming / interactive systems	Behaviour modelling, sequence prediction, agent policy optimisation	RNN/Transformer, representation learning, reinforcement learning	Highlights value of temporal progression modelling; motivates hierarchical behavioural abstraction for long-horizon trajectories
Cybersecurity (incl. insider threat)	Intrusion/anomaly detection, behavioural analytics, correlation across logs, robustness under adversaries	Anomaly detection, sequence models, hybrid pipelines, modular fusion, adversarial robustness methods	Directly motivates the thesis progression: ANN baseline → SOM–ANN stabilisation → MANN modular semantics → hierarchical modelling → R-Guard robustness

2.7.4 Common ML Workflow

Machine-learning systems for behavioural security analytics are typically developed through a pipeline comprising data preparation, feature construction, model training, tuning, and evaluation. In insider-threat detection, the pipeline must be interpreted through additional constraints: temporal validity, behavioural plausibility, label ambiguity, and the risk of misleading performance estimates under leakage or identity memorisation [9], [10], [13].

To avoid repetition with later experimental chapters, the discussion here is framed as a set of principles that guide the implementation of all contributions in this thesis:

- **Preprocessing and integration:** Align heterogeneous logs into coherent behavioural representations while preserving chronology and avoiding leakage-prone transformations.
- **Feature construction:** Combine activity intensity measures, temporal indicators, and cross-domain ratios in ways that remain interpretable and defensible for behavioural reasoning.
- **Model selection and training:** Choose models whose inductive biases match the data regime (tabular, sequential, multi-domain) and apply regularisation to mitigate overfitting under imbalance.
- **Tuning and calibration:** Perform hyperparameter tuning under temporally faithful validation to avoid optimistic estimates; calibrate thresholds to reflect operational trade-offs.
- **Evaluation discipline:** Prefer chronology-preserving splits, PR-oriented analysis under imbalance, and robustness checks under drift/adaptation where relevant.
- **Monitoring and validity boundaries:** Interpret results as benchmark-grounded evidence with declared transferability limits; treat deployment as a governed socio-technical process.

These principles serve as the methodological bridge between the limitations discussed in Section 2.6 and the architecture-centric solutions reviewed in Section 2.8. In later chapters, they are instantiated in the concrete experimental protocols used for the ANN baseline and the thesis architecture progression.

2.7.5 Positioning the Architecture

The preceding subsections clarify that model choice in insider-threat detection must respond to behavioural structure, temporal progression, cross-domain evidence, and

adversarial conditions. Within this thesis, ANN-based modelling provides the baseline non-linear discriminative substrate, while structured architectures address increasingly specific limitations: SOM–ANN hybridisation improves representation stability; MANN introduces domain-aware modular semantics; hierarchical modelling captures multi-level behavioural progression; and R-Guard addresses behavioural adversarial robustness.

This progression is summarised in Figure 2.11 (introduced in the ANN baseline section) and provides a coherent interpretive framework for linking literature limitations to the architectural solutions reviewed next. The following section therefore reviews hybrid, modular, and hierarchical ML architectures through the lens of insider behavioural modelling requirements and positions the thesis contributions relative to prior work.

Section summary. This section has positioned the work within the broader AI/ML landscape and clarified where SOM and ANN sit within common learning paradigms. By linking machine-learning foundations to the specific constraints of insider-threat detection—including temporal validity, multi-log heterogeneity, imbalance, and adversarial adaptation—it has provided a coherent bridge from the methodological limitations in Section 2.6 to the structured architectural responses reviewed next. The following section therefore examines hybrid, modular, and hierarchical ML architectures as principled mechanisms for aligning model design with behavioural security requirements.

2.8 Hybrid, Modular and Hierarchical ML

Building on the methodological limitations identified in Section 2.6 and the machine-learning positioning developed in Section 2.7, this section reviews hybrid, modular, and hierarchical ML architectures as structured responses to insider-threat detection challenges. The literature shows that architectural sophistication has increased considerably over the past decade—moving from rule-driven anomaly detectors toward deep learning, hybrid pipelines, and multi-level reasoning systems—yet many foundational socio-technical challenges remain unresolved. In particular, behavioural semantics, identity–behaviour confounding, temporal drift, data leakage, and adversarial manipulation continue to limit operational validity even when nominal benchmark performance appears high [8], [9], [10], [14].

This section critically evaluates three dominant architectural families in insider-threat ML research:

1. **Hybrid architectures**, which combine unsupervised and supervised components;

2. **Modular architectures**, which specialise modelling by behavioural domain and fuse evidence across logs;

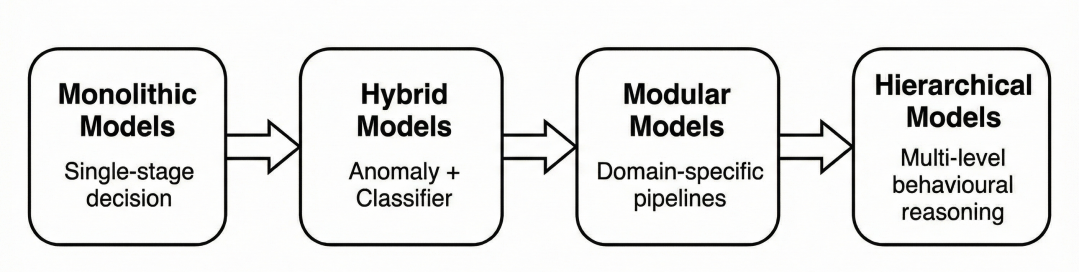


Figure 2.7: Evolution of insider-threat detection architectures from monolithic models toward *hybrid* pipelines (unsupervised representation learning combined with supervised discrimination), *modular* multi-log designs (domain-specific models with fusion), and *hierarchical* frameworks (multi-level behavioural abstraction across time and context).

3. **Hierarchical architectures**, which attempt to model behaviour across multiple temporal and semantic levels.

Although each family introduces valuable ideas, all exhibit structural limitations that motivate the integrated hybrid, modular, and hierarchical contributions developed in Chapters 3–5 and the robustness mechanisms introduced in Chapter 6.

2.8.1 Hybrid Architectures

Hybrid architectures combine complementary ML components—typically an unsupervised representation learner with a supervised classifier. The most established hybrid pattern integrates a Self-Organising Map (SOM) with a downstream classifier, a design used in early insider-threat systems and still influential in contemporary pipelines [14], [30], [37]. Variants include SOM+SVM, SOM+Random Forest, and autoencoder+classifier pipelines, alongside broader hybrid deep-learning designs that combine convolutional, recurrent, and attention-based components for threat detection in complex environments [39].

Hybrid approaches offer several advantages. They decompose learning into interpretable stages (e.g., clustering vs. discrimination), provide partial behavioural visualisation (especially via SOM topologies), and support component-level retraining when distributions shift. On synthetic benchmarks, hybrids can outperform monolithic classifiers by stabilising noisy representations and reducing sensitivity to raw-feature variance.

However, closer examination reveals structural weaknesses that constrain real-world viability.

2.8.1.1 Lack of Joint Optimisation

A central limitation of many hybrid designs is the absence of joint optimisation. SOMs cluster input data using unsupervised learning (often Euclidean distance in a projected topology), whereas the downstream classifier learns discriminative

boundaries optimised for malicious–benign prediction. Because these components are trained independently:

- SOM embeddings may not align with discriminative structure;
- classifiers can overfit to SOM artefacts (e.g., BMU coordinates) rather than behavioural meaning;
- behaviourally meaningful gradients cannot propagate across the pipeline.

This misalignment can produce brittle decision boundaries that deteriorate under behavioural drift, role change, or realistic organisational variation.

2.8.1.2 Limitations of SOM Topology and Stability

Although SOMs provide intuitive 2D maps, topology preservation does not guarantee behavioural semantic preservation. High-dimensional behavioural manifolds can collapse during projection, benign variability can trigger unstable BMU activation, and small perturbations can reorganise neighbourhood structure. Recent work on improving SOM stability (e.g., virtual winning neuron mechanisms) reinforces that representation stability remains a non-trivial challenge for anomaly detection and clustering in complex data [40]. In insider contexts, topology instability is particularly problematic because behavioural baselines evolve over time and because adversarially subtle changes may exploit representational fragility (see Section 2.10).

2.8.1.3 Behavioural Shallowing in Feature Pipelines

Many hybrid pipelines operate on flattened feature vectors or engineered summaries that omit behavioural semantics. As argued in Section 2.5, insider detection requires reasoning over privilege context, workflow structure, role expectations, and temporal patterns. Hybrid models often reduce these cues to abstract numeric vectors, creating pipelines that are technically elegant but behaviourally shallow.

2.8.1.4 Pipeline Fragility and Leakage

Hybrid systems typically involve multi-stage preprocessing (feature extraction, windowing, normalisation, clustering, classification). Multi-stage pipelines increase the likelihood of leakage (e.g., fitting scalers on the full dataset before splitting) and can amplify overfitting to intermediate representations rather than genuine behavioural patterns. This contributes to a common pattern in the literature: strong benchmark metrics paired with weak ecological validity [13].

2.8.2 Modular Architectures

Modular architectures attempt to overcome hybrid limitations by decomposing insider behaviour into domain-specific pathways. Typical modules include authentication behaviour, file-system interactions, email activity, device usage, and web/network activity. Each module processes its own subset of logs, learning domain-specific representations before a fusion layer integrates outputs into a unified risk estimate.

This design aligns naturally with enterprise telemetry and operational security monitoring (e.g., SIEM-integrated pipelines), where heterogeneous log sources and behavioural biometrics may be integrated to detect anomalous insider activity [41]. The Modular ANN (MANN) paradigm—developed further in Chapter 4—embodies this approach by explicitly separating behavioural domains that exhibit different statistical structure and semantic meaning. For example, authentication logs often encode temporal regularity and privilege usage, whereas file-system logs capture task-level semantics, and email logs reflect social and workflow interactions.

Modular systems are more behaviourally sensitive than many flat hybrids, but they also exhibit structural deficiencies.

2.8.2.1 Isolation and Missing Cross-Domain Reasoning

Insider behaviour often spans multiple domains. Malicious insiders may coordinate actions across authentication, file access, device usage, and communication (e.g., privilege escalation followed by anomalous file access). Modular architectures frequently treat domains as independent and fuse only at a late stage, limiting the model's ability to capture cross-domain dependencies essential for detecting coordinated workflows.

2.8.2.2 Weak Semantic Fusion

Fusion mechanisms commonly concatenate embeddings or combine probabilities without ensuring:

- temporal synchronisation across domains;
- semantic alignment of representations;
- comparable feature scales and uncertainty calibration;
- robustness to targeted manipulation of weaker modules.

This can create exploitable seams: adversaries may maintain benign authentication patterns while manipulating file or email behaviour in domains that are noisier or under-modelled.

2.8.2.3 Behavioural Bias and Module Overfitting

Different domains exhibit different levels of variability and signal-to-noise ratio. Rich or highly variable modules may dominate, suppressing semantically important signals from sparse domains. This imbalance can hide malicious behaviour that exploits under-modelled channels and inflate benchmark performance.

2.8.2.4 Lack of Behavioural Hierarchy

Modularity separates domains, but it does not automatically encode behavioural hierarchy: events form sessions, sessions form tasks, tasks form behavioural states, and states are shaped by privilege and organisational context. Without hierarchical scaffolding, modular systems risk detecting fragments of behaviour rather than behavioural meaning.

2.8.3 Hierarchical Architectures

Hierarchical models represent a conceptually advanced direction in insider-threat ML because they recognise that behavioural meaning emerges at higher levels of abstraction than raw events. Hierarchical architectures include multi-resolution sequence models (event-to-session encoders), task-level behavioural models, multi-granularity transformers, and role-aware behavioural abstraction systems [42], [43]. This direction aligns with recent research emphasising multi-granularity user anomaly detection, where abnormality may only become visible when behaviour is represented at multiple resolutions [44].

Despite their promise, hierarchical insider-threat modelling remains underdeveloped and faces significant limitations.

2.8.3.1 Hierarchical Sequence Models

Hierarchical RNNs, attention models, and multi-granularity transformers can represent micro- and macro-temporal structure, but many implementations lack organisational semantics, privilege constraints, workflow context, and behaviour-state transitions. As a result, models may be mathematically hierarchical but behaviourally shallow.

2.8.3.2 Behavioural-State Modelling Constraints

Behavioural-state approaches (e.g., HMMs, CRFs, dynamic Bayesian networks) conceptualise insider behaviour as transitions between latent risk states. While theoretically appealing, many implementations rely on simplistic state definitions and Markovian assumptions that do not match real insider trajectories. Without integration of role semantics and workflow context, behavioural-state models can have limited operational interpretability.

2.8.3.3 Hierarchical Representation

Deep hierarchical systems (multi-level transformers, domain-to-user unification models) remain rare partly because realistic datasets are scarce, labels are limited, and evaluation pitfalls can obscure true performance. Without rigorous temporal validity and behavioural grounding, learned embeddings may capture statistical regularities rather than behavioural meaning [13].

2.8.3.4 Missing Semantic Constraints and Privilege Context

Perhaps the most critical limitation in hierarchical insider models is the absence of formal semantic constraints. Even advanced architectures often omit privilege semantics, role expectations, organisational policies, and workflow constraints. Without these:

- anomalies are detected without interpretable meaning;
- privilege-consistent malicious behaviour remains difficult to detect;
- stability under behavioural drift and role change is limited.

This motivates the semantics-aware hierarchical framework introduced in Chapter 5.

2.8.4 Architectural Fragility

Across all three paradigms, architectural fragility becomes more evident under behaviourally constrained adversarial perturbations (Section 2.10). Hybrid SOM–ANN pipelines can exhibit drift elasticity and vulnerability to poisoning or representation manipulation. Modular architectures can be subverted through domain-targeted manipulation, especially when fusion is late and cross-domain consistency is weak. Hierarchical systems can fail when adversaries preserve surface-level patterns while altering behavioural meaning or intent.

These observations reinforce a core insight that informs Chapter 6: architectural complexity does not equate to robustness. Without behaviourally grounded threat models, constraint-aware evaluation, and robustness mechanisms, even advanced architectures can remain brittle under adaptive insider behaviour.

2.8.5 Synthesis and Contributions

Hybrid, modular, and hierarchical architectures each contribute important ideas: hybrids improve representation stabilisation and partial interpretability; modular systems reflect multi-log domain specificity; and hierarchical models capture multi-level behavioural structure. However, none fully resolves key insider-threat requirements, including behavioural semantics, privilege context, cross-domain dependency

modelling, temporally faithful evaluation under drift, and robustness under behaviourally plausible adversarial adaptation [8], [9], [10].

These limitations motivate the integrated architectural designs developed in the thesis. Chapter 3 introduces SOM–ANN hybridisation as a stability-oriented representation refinement. Chapter 4 introduces a modular ANN framework (MANN) to preserve domain semantics and reduce feature dominance across logs. Chapter 5 introduces a semantics-aware hierarchical modelling framework for multi-level behavioural progression. Chapter 6 then extends the architecture progression with adversarial robustness through R-Guard, explicitly addressing behaviourally constrained adaptive threats that can undermine high-performing detectors when evaluation is limited to benign benchmark conditions.

Section summary. In summary, the literature supports a shift from monolithic modelling toward structured learning systems. This section has shown that hybrid, modular, and hierarchical architectures represent a principled shift toward structured learning systems that better reflect the heterogeneity, temporal progression, and multi-domain nature of insider behaviour. However, the literature also indicates that structural sophistication alone does not guarantee behavioural validity or robustness, particularly under drift, cross-domain dependency, and adaptive manipulation. These observations motivate the staged architecture progression adopted in this thesis, which moves from hybrid SOM–ANN stabilisation to modular and hierarchical behavioural modelling, and culminates in behaviourally grounded robustness mechanisms in Chapter 6.

2.9 Relational Modelling and GNN Misalignment

Relational modelling has recently emerged as a promising direction in insider-threat detection. Graph Neural Networks (GNNs), in particular, have been proposed to capture relational dependencies between users, devices, files, hosts and network resources. Motivated by success in social networks, recommender systems and fraud detection, GNNs appear attractive for modelling insider-threat environments due to their ability to represent structural interactions and message-passing dynamics [45], [46].

However, despite this apparent promise, the theoretical assumptions underlying GNNs are fundamentally misaligned with the socio-technical and behavioural properties of insider threats. Insider behaviour does not arise from stable graph structures, nor does it propagate relational signals in ways compatible with standard GNN formulations. Moreover, the message-passing operations that define GNNs create structural vulnerabilities that adversaries can exploit.

This section critically analyses the limitations of relational modelling and GNNs in insider-threat detection. We demonstrate that GNNs fail to capture behavi-

oural meaning, privilege semantics, temporal evolution, or adversarial deception strategies. These findings reveal why GNNs, despite their popularity, cannot be relied upon as the primary modelling approach for insider-threat environments and must be complemented by behavioural, hierarchical and adversarial frameworks (Sections 2.8 and 2.10).

2.9.1 The Appeal of Relational Modelling

Relational structures exist in organisational environments: users collaborate, access shared resources, communicate through email networks, authenticate across machines and interact with digital assets. These interactions seem naturally representable as graphs where:

- nodes = users, hosts, devices, files, applications;
- edges = interactions, accesses, communications, co-occurrences.

GNNs extend these structures with message passing, enabling models to propagate information across edges to learn node representations:

$$h_v^{(k+1)} = \phi\left(h_v^{(k)}, \square_{u \in \mathcal{N}(v)} \psi\left(h_v^{(k)}, h_u^{(k)}, e_{uv}\right)\right),$$

where ψ and ϕ are learnable functions and $\square$ is an aggregation operator [45].

When applied to insider-threat detection, GNNs promise to:

- capture relational anomalies (e.g., unusual user–host interactions),
- represent multi-entity dependencies,
- detect structural deviations in access patterns,
- incorporate social or organisational relationships.

However, relational promise does not equate to behavioural or semantic fidelity.

2.9.2 Graphs and Behavioural Semantics

A major conceptual limitation is that relational graphs constructed from system logs do not encode *behavioural meaning*. The presence of an edge between a user and a file or machine does not indicate *why* the interaction occurred, whether it was contextually appropriate or whether it violated role expectations.

For example:

- A user may legitimately access a restricted file due to a new assignment.
- Two users may send emails due to an organisational restructure.

- A system administrator may log into many machines as part of routine maintenance.

GNNs treat these relational changes as structural modifications without semantic grounding. The absence of privilege semantics, organisational context and behavioural interpretation means that GNN representations are fundamentally blind to insider intent or behavioural appropriateness [8].

This misalignment produces systems that detect relational anomalies but fail to identify privilege-consistent malicious behaviour—a hallmark of insider attacks.

2.9.3 Graph Construction and Structural Bias

Graph construction is a deceptively simple step that exerts a profound influence on the behaviour and interpretability of GNN-based insider-threat models. Any relational representation requires the researcher to decide what should constitute a node, how edges should be defined, how temporal information is aggregated, and which features are propagated across the graph. These modelling decisions are discretionary rather than objectively determined, and each choice introduces inductive biases that shape what patterns the model is able—or unable—to learn.

In insider-threat research, graph building is often operationalised using straightforward structures such as user–resource bipartite graphs, email communication networks, or host–user login interactions. While intuitive, these choices necessarily compress complex organisational behaviour into simplified relational forms. Such compression leads to several forms of structural distortion: temporal sequences are flattened into static edges, removing meaningful behavioural trajectories; task- and role-level semantics are obscured, producing behavioural oversimplifications; and high-activity users become disproportionately high-degree nodes, conflating identity with structural salience. Depending on aggregation windows and graph granularity, the resulting graph may appear artificially sparse or artificially dense, neither of which reflects real operational complexity.

As observed empirically by [13], graphs derived from synthetic datasets such as CERT frequently encode dataset artefacts rather than authentic organisational structures. When GNNs are trained on these relational abstractions, they risk learning structural illusions—patterns induced by the graph-construction process itself—rather than insights grounded in behavioural meaning. This structural bias fundamentally limits the validity of relational modelling for insider-threat detection and contributes to the broader misalignment between organisational graphs and real behavioural semantics.

2.9.4 Temporal Dynamics

Insider behaviour is inherently temporal, unfolding through sequences of actions, evolving intentions and multi-stage operational patterns. Yet most graph-based insider-threat studies rely on *static* relational structures aggregated over long windows—hours, days or even weeks. Such aggregation collapses behavioural sequences into single edges, masks the progression of multi-stage attacks, and removes the temporal distinctions needed to identify drift in identity, role usage or workflow patterns.

Even when dynamic GNNs (DGNNs) are employed [47], [48], [49], their underlying assumptions often misalign with insider-threat realities. DGNNs typically expect smooth temporal evolution, whereas insider behaviour is highly irregular, strategically timed and sometimes intentionally deceptive. Temporal sampling windows frequently introduce leakage (Section 2.6.7), and behavioural trajectories rarely propagate relational influence in the same manner as social or communication networks. Multi-stage attacks also do not follow graph-diffusion or neighbourhood-propagation patterns, undermining the relevance of temporal message passing for threat detection.

These mismatches produce brittle graph representations that distort behavioural meaning and underestimate the complexity of insider temporal patterns.

2.9.5 Privilege and Role Semantics

Privileges determine what a user is *allowed* to do, while roles describe what a user is *expected* to do within organisational workflows. These behavioural semantics shape boundaries that differentiate normal from suspicious actions. However, they cannot be recovered from graph topology alone. Privilege is not a relational property, role expectations are not encoded in node features, and organisational hierarchies bear little resemblance to behavioural norms.

Consequently, GNNs often misinterpret structural signals. A system administrator appears as a high-degree node in login graphs, yet this reflects legitimate high-volume activity rather than anomaly. Conversely, a malicious insider may maintain low-degree structural signatures while exhibiting subtle but meaningful behavioural drift. Graph topology cannot differentiate these patterns, because the relevant semantics lie outside the relational structure.

Thus, relational features are insufficient substitutes for behavioural context, and models that rely exclusively on graph topology lack the semantic depth required for insider-threat reasoning.

2.9.6 Message Passing

The message-passing mechanism central to GNNs propagates node attributes across neighbourhoods. In insider-threat graphs, this mechanism produces a systematic distortion: user identity becomes the dominant signal in learned embeddings. High-activity users overshadow quieter ones, structural centrality correlates with model confidence, and malicious behaviour can be diluted when surrounded by benign neighbourhood structures.

This amplification effect contradicts the requirement for identity–behaviour disentanglement (Section 2.6). Instead of highlighting behavioural irregularities, message passing redistributes identity signatures, thereby emphasising structural correlations at the expense of behavioural meaning. The result is a representation in which identity artefacts overpower the subtle behavioural deviations characteristic of insider threats.

2.9.7 Graph Sparsity, Density and Structural Instability

Organisational systems generate interaction logs with highly uneven frequencies across communication, access, file usage and device interactions. When these interactions are converted into graphs, the resulting structures are often either extremely sparse (e.g., email communications or privilege hierarchies) or artificially dense (e.g., host–user login networks). Drift in organisational behaviour further destabilises connectivity patterns over time.

GNNs are sensitive to such structural irregularities. Extreme sparsity leads to weak neighbourhood signals and vanishing behavioural specificity, while dense graphs produce neighbourhood explosions that dominate message passing and collapse embeddings. Structural instability from drift disrupts consistent learning and erodes the validity of relational reasoning. As a result, graph-based embeddings become unreliable and semantically ambiguous.

2.9.8 Adversarial Vulnerabilities

Insider adversaries can exploit graph-construction decisions and message-passing mechanisms to evade detection. By increasing benign interactions, they can manipulate node degree to mask malicious behaviour. Through neighbourhood poisoning, they can influence message propagation by injecting benign relational patterns. Timing malicious actions during low-activity intervals allows them to exploit temporal aggregation windows. Manipulating edge weights or generating noise interactions further distorts the relational signals that GNNs rely upon.

These strategies parallel well-known adversarial attacks in graph learning [34], [36], [50], [51], but insider adversaries possess an advantage: they operate within

the system and understand its workflows. Their ability to modify organisational interactions makes relational structures particularly vulnerable to manipulation. This highlights the necessity of incorporating adversarial modelling (Section 2.10) to complement and critique relational approaches.

2.9.9 Limitations of Structure-Only Reasoning

Across the limitations identified above, a consistent pattern emerges: relational models detect anomalies in structural topology rather than anomalies in behaviour. Insider threats, however, are characterised by privilege-consistent malicious actions, workflow-aligned deception, subtle deviations embedded in legitimate interactions and behavioural drift that preserves structural patterns. Because these behaviours rarely manifest as relational irregularities, structural anomaly detection cannot reliably surface early indicators of insider intent [8].

2.9.10 Insufficiency of Graphs

Relational representations lack the capacity to model the multi-level behavioural structures essential for insider-threat detection. Task-level intentions, session-level semantics, role expectations, organisational context and cross-domain behavioural integration all lie outside the representational scope of graph topology. As discussed in Section 2.8, hierarchical and behavioural models provide a more appropriate foundation for reasoning across these levels. Purely relational models cannot support the multi-level abstraction hierarchy required for meaningful behavioural inference.

2.9.11 Summary of Relational–Behavioural Misalignment

Together, these limitations demonstrate that GNNs and related relational models, while powerful for structural analysis, remain fundamentally misaligned with the behavioural, socio-technical and adversarial nature of insider threats. Their reliance on topology, sensitivity to graph-construction bias, vulnerability to drift, amplification of identity artefacts, inability to encode privilege or role semantics and susceptibility to adversarial manipulation—collectively—highlight the inadequacy of structure-only reasoning.

These constraints motivate the transition to Section 2.10, where behavioural adversaries are formalised and the weaknesses of classical, hybrid, modular and relational approaches are examined under adversarial perturbation.

2.10 Adversarial Modelling of Insider Threats

Insider adversaries differ fundamentally from external attackers: they act within legitimate privilege boundaries, possess operational knowledge of systems and monitoring practices, and understand the behavioural expectations encoded in organisational workflows. The socio-technical and behavioural foundations established in Sections 2.2–2.5 naturally motivate an adversarial framework based on *behaviourally feasible perturbations*.

In contrast to conventional adversarial machine learning—where attackers freely perturb inputs within L_p norms [16]—insider adversaries operate within tightly bounded semantic, temporal and contextual constraints. These constraints impose structure on the adversarial space and fundamentally reshape both the threat model and the model-evasion strategies available to insiders. Recent studies on adversarial manipulation in sequential and temporal systems [34], [52] reinforce the vulnerability of anomaly detectors, deep temporal architectures and GNNs when perturbations exploit structural weaknesses.

This section formalises a mathematically principled model of behaviourally constrained adversaries and demonstrates why existing insider-threat detection architectures—hybrid, modular, hierarchical and relational—exhibit structural brittleness under such attacks. The formalism provides the foundation for the adversarially robust behavioural architecture developed in Chapter 6 (R-Guard).

2.10.1 Limits of Classical Adversarial ML

Classical adversarial ML assumes an attacker who may perturb input vectors $x \in \mathbb{R}^d$ arbitrarily within a norm-bounded region such as $\|x' - x\|_p \leq \epsilon$. Insider adversaries, however, cannot freely manipulate logs or behavioural traces. Their actions must remain consistent with organisational privilege structures, plausible role semantics, and the socio-temporal context in which they operate.

Let e_t denote a behavioural event at time t. Insider feasibility constraints can be expressed as:

$$e'_t \in \mathcal{F}(e_t),$$

where $\mathcal{F}(e_t)$ is the *behavioural feasibility set*. Unlike traditional adversarial sets (e.g., L_p balls), $\mathcal{F}$ is defined by socio-technical constraints:

$$
\begin{aligned}
\text{Privilege constraint:} &\quad \pi(e'_t) = \pi(e_t), \\
\text{Role constraint:} &\quad r(e'_t) \in R_u, \\
\text{Temporal constraint:} &\quad t' \in \text{valid\_window}(u), \\
\text{Context constraint:} &\quad c(e'_t) \approx c(e_t), \\
\text{Psychological drift:} &\quad \|e'_t - e_t\| \text{ small.}
\end{aligned}
$$

These constraints imply:

$$
\mathcal{F}(e_t) \subset \{e'_t : \|e'_t - e_t\| \leq \epsilon\} \quad \text{but is } \textit{not} \subseteq \text{an } L_p \text{ ball.}
$$

Thus classical adversarial assumptions fail: insider attacks require a new perturbation model grounded in behavioural semantics.

2.10.2 Behaviourally Constrained Perturbation Operators

To operationalise $\mathcal{F}(e_t)$, we define four core perturbation operators representing feasible insider manipulations. Each operator preserves plausibility while modifying the behavioural trace to deceive ML systems.

2.10.2.1 Temporal Scheduling Operator $\mathcal{S}_\rho$

Insiders often schedule malicious actions during workload peaks, exploiting the workload density function $\rho(t)$:

$$
\mathcal{S}_\rho(e_t) = e_{t+\tau}, \quad \tau = \arg\max_{\tau'} \rho(t + \tau').
$$

This hides malicious events within legitimate operational noise, degrading temporal anomaly detectors **li2024behavioraladversarial**.

2.10.2.2 Gradual Drift Operator Δ_δ

Insider escalation frequently unfolds as a slow behavioural drift:

$$
e'_t = e_t + \delta_t, \qquad \|\delta_t\| < \epsilon,
$$

where the cumulative drift

$$
\sum_t \delta_t \rightarrow \text{malicious trajectory}
$$

remains psychologically and operationally plausible [11]. Drift rapidly degrades SOMs, autoencoders and distance-based detectors.

2.10.2.3 Sequence Reordering Operator $\mathcal{P}_\sigma$

Given a log sequence $s = (e_1, e_2, \ldots)$, an insider may shuffle non-critical events provided workflow semantics are preserved:

$$\mathcal{P}_\sigma(s) = (e_{\sigma(1)}, \ldots), \quad \sigma \in \text{workflow-preserving permutations.}$$

Small permutations destabilise the temporal inductive biases of RNNs, LSTMs and Transformers **zhang2023sequentialaml**.

2.10.2.4 Privilege-Aligned Substitution Operator $\mathcal{S}_{priv}$

Insiders may substitute malicious actions with privilege-equivalent benign ones:

$$e'_t = \mathcal{S}_{priv}(e_t), \quad \pi(e'_t) = \pi(e_t).$$

These substitutions remain legitimate under Zero Trust enforcement [1], [53] and are invisible to models lacking semantic privilege-role grounding.

2.10.3 Composite Behavioural Adversaries

Realistic insider adversaries combine multiple perturbation operators. We formalise four composite adversarial classes that emerge from these combinations.

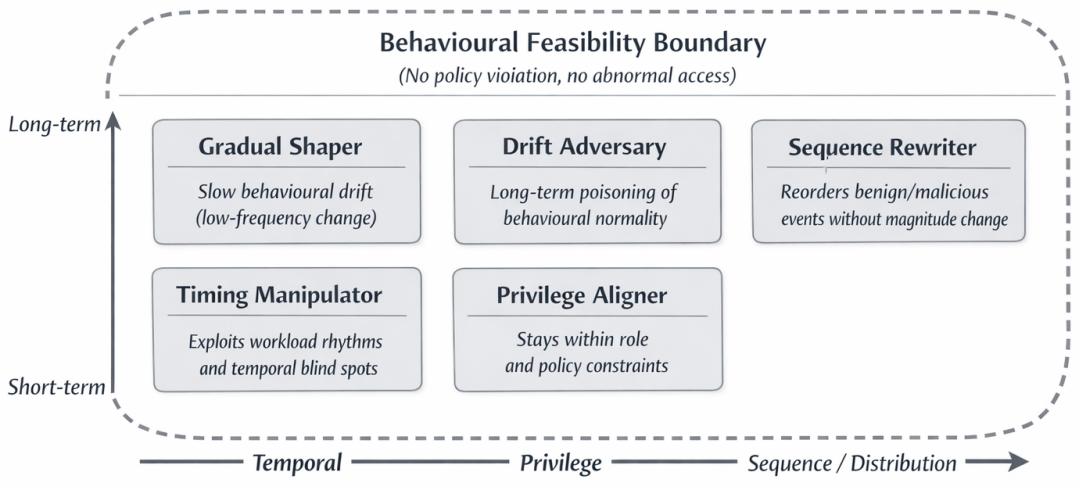


Figure 2.8: Behaviourally constrained insider adversary space. Each adversary operates within legitimate privilege, role semantics, and operational feasibility while manipulating different behavioural dimensions (temporal dynamics, privilege alignment, and event structure). Unlike overt attacks, these strategies preserve local plausibility and evade detectors that rely on magnitude-based anomalies or static behavioural baselines.

2.10.3.1 Gradual Shaper (Drift Adversary)

The Gradual Shaper exploits drift elasticity in ML models:

$$e'_t = e_t + \delta_t, \qquad \|\delta_t\| < \epsilon.$$

Over time:

$$e'_1 \to e'_2 \to \cdots \to e'_T$$

forms a malicious trajectory that remains undetected until late-stage escalation. Drift poisons SOM centroids, destabilises hybrid SOM–ANN pipelines and induces catastrophic forgetting in ANNs.

2.10.3.2 Timing Manipulator

The adversary selects t' such that $\rho(t')$ maximises concealment:

$$t' = \arg\max_t \rho(t).$$

This exploits the fact that temporal models conflate workload density with normality.

2.10.3.3 Privilege Aligner

By ensuring $\pi(e'_t) = \pi(e_t)$, the adversary manipulates semantics without altering privilege footprints. Relational models and GNNs cannot distinguish such privilege-consistent deception.

2.10.3.4 Sequence Rewriter

The adversary generates $s' = \mathcal{P}_\sigma(s)$, where σ preserves workflow validity but disrupts temporal expectations, destabilising attention-based mechanisms and RNN recurrence.

2.10.4 Behavioural Feasibility Manifold

The behavioural feasibility manifold for user u is defined as:

$$\mathcal{M}(u) = \left\{ e_t \;\middle|\; \pi(e_t) \in \Pi_u,\ r(e_t) \in R_u,\ c(e_t) \in C_u,\ t \in T_u \right\}.$$

A perturbation is behaviourally valid iff:

$$e'_t \in \mathcal{M}(u).$$

This ensures that adversarial traces remain privilege-preserving, workflow consistent and contextually plausible. Classical ML models fail because they are not designed to recognise or enforce these constraints.

2.10.5 Architectural Vulnerabilities

Each architectural paradigm exhibits characteristic vulnerabilities:

Hybrid SOM–ANN. SOM drift and poisoned neighbourhoods misinform ANN layers, causing representation collapse. Pipeline decoupling amplifies inconsistencies.

Modular Networks. Attackers exploit the weakest or least monitored module. Fusion layers misalign contaminated signals, and domain partitioning prevents cross-domain anomaly integration.

Hierarchical Models. Layer-specific perturbations—session-level, task-level or timing-level—break the hierarchical dependencies required for multi-level reasoning.

Relational Models / GNNs. Relational structures are easily manipulated through noise edges, degree inflation and neighbourhood poisoning. Message passing magnifies adversarial perturbations.

2.10.6 Adversarial Taxonomy

A complete taxonomy is provided in Table 2.6. This extends earlier summaries with formal operators and behavioural constraints.

Table 2.6: Behaviourally constrained adversarial insider taxonomy.

Adversary Type	Operator(s)	Feasibility Constraints	Model Failure Modes
Gradual Shaper	Δ_δ	$\|\delta_t\| < \epsilon$; privilege/role consistency	SOM drift; ANN overfitting; hybrid collapse
Timing Manipulator	$\mathcal{S}_\rho$	High-workload temporal windows	Sequence dilution; temporal masking; window-based instability
Privilege Aligner	$\mathcal{S}_{priv}$	$\pi(e'_t) = \pi(e_t)$; semantic consistency	Semantic blindness; relational invisibility; privilege-aligned evasion
Sequence Rewriter	$\mathcal{P}_\sigma$	Workflow-constrained permutations	Transformer/LSTM sensitivity; recurrent instability; hybrid misalignment

2.10.7 Summary

This section introduced the first adversarial model tailored specifically to insider threats—grounded not in L_p perturbations but in behavioural feasibility, socio-technical constraints and privilege-role semantics. It demonstrated that classical ML architectures—including modular, hybrid, relational and hierarchical models—exhibit systemic fragilities when confronted with behaviourally constrained adversaries.

This sets the foundation for Section 2.7, which synthesises the unresolved gaps across Sections 2.1–2.6 and clarifies the conceptual and architectural voids addressed by this thesis.

2.11 Synthesis of Unresolved Gaps

The preceding sections reveal that insider-threat detection occupies a space where behavioural complexity, socio-technical context, and adversarial reasoning intersect. Classical anomaly detection, deep learning architectures, graph-based systems and hybrid pipelines each address fragments of the problem, but none provide a unified behavioural, temporal and adversarially robust framework. This synthesis consolidates the unresolved gaps across methodological, conceptual and architectural dimensions, clarifying why insider threats remain an open challenge for machine learning.

2.11.1 Behavioural Gaps

Existing models capture statistical deviation but do not encode the behavioural semantics that distinguish malicious yet privilege-consistent actions from benign ones. Role expectations, task boundaries, workflow structures and routine sequences are rarely represented explicitly. As a result, models fail to detect subtle behavioural drift, socio-technical inconsistencies or intent-driven anomalies. Behavioural semantics remain implicit, unmodelled and inaccessible to downstream reasoning.

2.11.2 Temporal Gaps

Temporal structure is fundamental to insider behaviour, yet detection models rely on static windows, flattened aggregates or fragile sequence representations. Temporal dynamics such as burst patterns, workload alignment, escalation trajectories and timing manipulation remain under-modelled. Deep sequential models struggle under mild perturbations, while static or aggregated approaches erase meaningful temporal signatures. No existing architecture integrates short-term actions with long-range behavioural evolution.

2.11.3 Conceptual Gaps in Relational Modelling

Relational models focus on structural interactions rather than behavioural meaning. GNN-based systems inherit the limitations of graph construction, message passing and topological abstraction. Organisational graphs do not encode privilege semantics, workflow context or session-level intent. Relational anomalies therefore do not correspond to behavioural anomalies, and models remain vulnerable to adversarial manipulation of degree, neighbourhood and edge patterns. Relational reasoning provides structure without semantics.

2.11.4 Methodological Gaps

Methodological limitations—data leakage, unrealistic sampling, temporal mixing, label noise and drift neglect—undermine the validity of experimental findings. Synthetic datasets such as CERT encode artefacts that distort model behaviour and induce misleading structural patterns. Many evaluation pipelines lack adversarial testing, overlook long-term drift, and rely on accuracy metrics that fail to capture behavioural robustness. Methodology remains decoupled from behavioural and adversarial realism.

2.11.5 Adversarial Gaps

Insiders operate under behavioural feasibility constraints that classical adversarial ML does not model. Existing detectors remain vulnerable to drift exploitation, timing manipulation, privilege-aligned substitution and sequence reordering. Hybrid, modular, hierarchical and relational models were shown to exhibit systemic weaknesses when perturbations preserve privilege and workflow semantics. No architecture explicitly models the adversarial behaviours unique to insider contexts.

2.11.6 Architectural Gaps

Architectures reviewed in Chapter 2 lack mechanisms to integrate multi-level behavioural reasoning, privilege-role semantics, temporal trajectories and adversarial robustness within a single coherent framework. Hybrid SOM–ANN pipelines couple incompatible representations; modular systems fracture behavioural meaning across isolated domains; hierarchical models lack semantic grounding; relational systems misalign structure with behaviour. There is no architecture capable of reasoning over behaviour, structure, context and adversarial manipulation jointly.

2.11.7 Integrated Research Gap

Taken together, these gaps reveal a fundamental limitation: *current machine learning systems do not model insider threats as behavioural, temporal and adversarial*

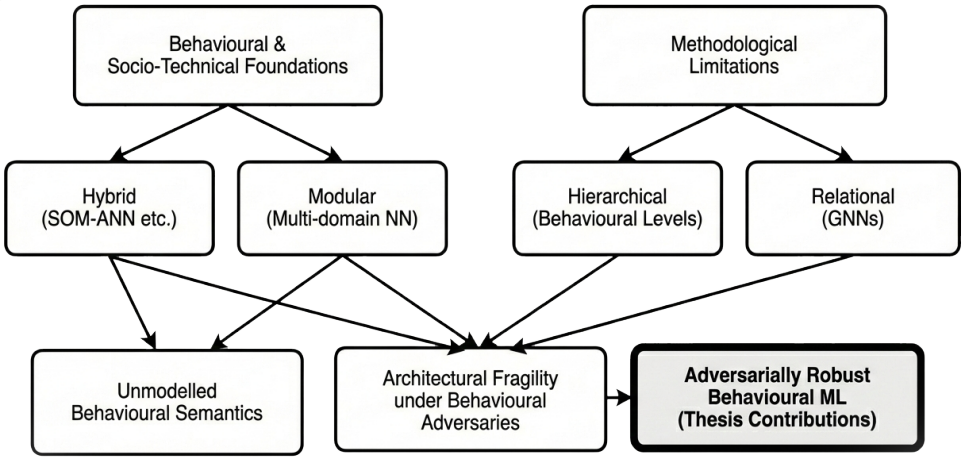


Figure 2.9: High-level roadmap linking behavioural foundations, methodological limitations and existing architectures to the unresolved gaps addressed by the SOMANN hybrid (Chapter 3), modular architecture (Chapter 4), multi-layered behavioural framework (Chapter 5) and adversarially robust R-Guard architecture (Chapter 6).

phenomena. They rely on static abstractions, structural proxies or shallow temporal patterns that overlook the socio-technical context within which insider behaviour unfolds. This creates an architectural void where behavioural semantics, multi-level reasoning, drift resilience and adversarial awareness remain unaddressed.

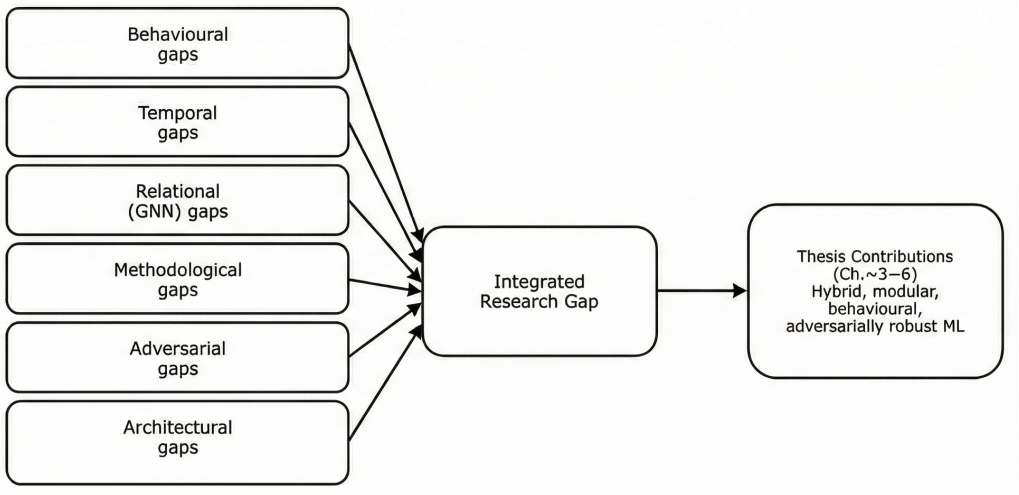


Figure 2.10: Synthesis of unresolved gaps motivating the architectural contributions of Chapters 3–6.

These unresolved gaps motivate the architectural progression adopted in this thesis. Hybrid SOM–ANN models address behavioural variance stabilisation (Chapter 3); modular architectures address domain heterogeneity (Chapter 4); hierarchical models address multi-level behavioural abstraction (Chapter 5). However, none of these approaches address adversarial behavioural manipulation, necessitating the robustness framework introduced in Chapter 6.

Table 2.7: Summary - Integrated synthesis of unresolved gaps in insider-threat ML research.

Gap Category	Unresolved Limitations
Behavioural	Lack of explicit modelling of role semantics, workflow norms, intent signals and behavioural drift.
Temporal	Flattened windows, weak long-range modelling, vulnerability to reordering and timing manipulation.
Relational	Misalignment between organisational graph structure and behavioural semantics; structural anomalies $\neq$ behavioural anomalies.
Methodological	Data leakage, unrealistic splits, drift neglect, synthetic artefacts and metric inflation undermine validity.
Adversarial	No modelling of behaviourally feasible perturbations; vulnerability to drift-based, privilege-aligned and temporal adversaries.
Architectural	Lack of unified behavioural–temporal–adversarial reasoning across hybrid, modular, hierarchical and relational systems.

2.11.7.1 Rationale for the Staged Architecture

The literature reviewed in this chapter identifies a set of recurring and progressively more demanding limitations in machine-learning approaches to insider-threat detection. These limitations concern not only predictive performance, but also the stability of behavioural representation, the preservation of domain-specific context, the modelling of behavioural progression across temporal horizons, and resilience to adaptive manipulation. The staged architectural progression investigated in this thesis is motivated by these limitations rather than by an *a priori* preference for particular algorithms or increasing architectural complexity.

The progression

$$\mathrm{ANN}_{\mathrm{std}} \rightarrow \mathrm{SOMANN} \rightarrow \mathrm{MANN} \rightarrow \mathrm{MLML} \rightarrow \mathcal{R}\text{-Guard} \tag{2.1}$$

should therefore be interpreted as a sequence of increasingly demanding *modelling requirements*. It is not claimed to constitute the unique or mathematically necessary ordering of insider-threat detection architectures. Rather, each transition introduces a modelling requirement that is not explicitly addressed by the preceding formulation and that the literature reviewed in this chapter identifies as relevant to reliable behavioural detection.

From predictive classification to stable behavioural representation. A conventional supervised ANN provides an appropriate common baseline because it

represents the problem as direct prediction from aggregated behavioural features. However, the literature reviewed earlier in this chapter shows that organisational telemetry is characterised by behavioural heterogeneity, substantial class imbalance, role-dependent variation, and temporal non-stationarity. A model operating directly on a unified feature representation may therefore learn decision boundaries that are sensitive to incidental feature variation or characteristics specific to the observed user population.

This motivates the first transition from direct supervised classification to a representation-oriented formulation. SOMANN introduces an unsupervised perceptual stage before supervised inference, creating an explicit architectural separation between behavioural organisation and risk classification. The theoretical motivation for this stage is therefore *representational stability*: before more complex relational or temporal structure is considered, the behavioural observations supplied to the decision process should be organised in a manner that is less sensitive to transient variation in the raw feature space.

The purpose of Chapter 3 is to evaluate this proposition empirically. The role of the present chapter is to establish why representational stability constitutes a relevant modelling requirement.

From representation stability to behavioural-domain grounding. Representational stability alone does not ensure that the functional distinctions between heterogeneous organisational activities are preserved. Authentication events, communication activity, file access, web behaviour, and device interaction differ in their frequency, operational meaning, and relationship to user roles. Combining these sources within a single undifferentiated representation may allow high-volume domains to dominate lower-volume but potentially informative behavioural evidence, or may obscure relationships whose meaning depends on the domain in which they occur.

The second modelling requirement is therefore *behavioural-domain grounding*. MANN addresses this requirement by introducing domain-specific representations whose outputs are integrated only after within-domain processing. The transition from SOMANN to MANN is consequently motivated by two distinct representational questions: whether behavioural observations can be represented with greater stability, and whether heterogeneous behavioural domains retain their functional distinctions within the learned representation.

This is a conceptual progression rather than a claim that MANN is mathematically dependent upon SOMANN. Chapter 4 evaluates the additional hypothesis that preserving domain-specific structure before controlled integration improves the representation of heterogeneous behavioural evidence.

From domain grounding to temporal behavioural structure. Domain-grounded representation remains incomplete when the behavioural significance of an activity depends on its position within a longer sequence. Insider behaviour may develop gradually, with individual events remaining locally plausible while their accumulation across sessions, tasks, or days becomes increasingly inconsistent with established behavioural patterns or organisational expectations.

This motivates the third requirement: *temporal abstraction*. The hierarchical MLML framework extends the modelling concern from domain-grounded representation to multiple temporal contexts, allowing local observations to be interpreted relative to intermediate and longer-horizon behavioural trajectories.

The theoretical significance of this transition is that temporal context is not treated simply as another input feature. Instead, behavioural progression becomes an explicit representational dimension. Chapter 5 evaluates whether this additional structure addresses temporal relationships that may remain inaccessible to flatter or short-horizon representations.

From predictive structure to adversarial robustness. The preceding stages primarily evaluate detection under nominal or naturally changing behavioural conditions. They do not directly evaluate the case in which an informed insider deliberately modifies observable behaviour in response to the detection process. Such adaptation introduces a distinct modelling requirement because timing, ordering, frequency, or other observable characteristics may be altered while the underlying malicious objective and operational plausibility are preserved.

The final modelling requirement is therefore *adversarial robustness*. Chapter 6 introduces behaviourally constrained perturbations and $\mathcal{R}$-Guard to examine whether detection remains reliable when observable behavioural characteristics are deliberately modified within the defined threat model.

Adversarial robustness is examined after the representational, domain, and temporal investigations for a methodological reason: robustness can then be evaluated with respect to behavioural structure whose relevance has already been established experimentally. This ordering does not imply that adversarial defences cannot be applied to simpler classifiers. Rather, the thesis asks whether behaviourally grounded representations remain informative when the observable evidence on which detection depends is strategically manipulated.

2.11.7.2 Why the Sequence Is Non-Arbitrary

The architectural progression can therefore be summarised as a sequence of residual modelling requirements:

$$\text{Predictive Baseline} \rightarrow \underbrace{\text{Representation Stability}}_{\text{SOMANN}} \rightarrow \underbrace{\text{Domain Grounding}}_{\text{MANN}} \rightarrow \underbrace{\text{Temporal Structure}}_{\text{MLML}} \rightarrow \underbrace{\text{Adversarial Robustness}}_{\mathcal{R}\text{-Guard}}. \tag{2.2}$$

The sequence is non-arbitrary within the scope of this thesis because each stage addresses a different dimension of the reliability problem identified in the literature. Increasing predictive complexity alone does not directly resolve representational instability; stable representation alone does not guarantee preservation of domain-specific behavioural structure; domain-aware modelling does not explicitly represent long-horizon behavioural progression; and predictive performance under nominal conditions does not establish robustness under strategic adaptation.

Table 2.8 summarises the relationship between these modelling requirements.

Table 2.8: Literature-derived rationale for the staged architectural progression.

Architecture	**Primary Modelling Focus**	**Residual Limitation**	**Resulting Requirement**
ANN_{std}	Direct supervised prediction	No explicit mechanism for stabilising heterogeneous behavioural representation before inference	Stable behavioural representation
SOMANN	Topology-informed representation stability	Heterogeneous behavioural domains remain represented within a unified behavioural space	Behavioural-domain grounding
MANN	Domain-specific representation and controlled fusion	Domain-aware representations may remain temporally local and fail to capture longer-horizon behavioural progression	Multi-horizon temporal abstraction
MLML	Temporal trajectory representation	Predictive representations remain potentially vulnerable to deliberate modification of observable behaviour	Adversarial robustness
$\mathcal{R}$-Guard	Behaviourally constrained robustness	Robustness is established only within the evaluated perturbation constraints and dataset conditions	Operational validation beyond the experimental threat model

The progression is therefore cumulative in *research logic*. Each investigation retains the modelling concern established by the preceding stage while introducing

an additional requirement needed to examine a residual limitation. This does not imply that SOMANN, MANN, MLML, and $\mathcal{R}$-Guard form an indivisible computational pipeline, nor that the same ordering must be adopted by every insider-threat detection system. Rather, they constitute a sequence of experimentally investigated responses to increasingly demanding aspects of behavioural detection.

Taken together, these literature-derived requirements provide the theoretical basis for the *Staged Behavioural Grounding and Robustness Principle* introduced in Chapter 1. Within this thesis, the principle proposes that reliable insider-threat detection requires behavioural information to be progressively examined in terms of representational stability, behavioural-domain grounding, temporal structure, and robustness to behaviourally feasible manipulation. Chapters 3–6 evaluate these requirements empirically, while Chapter 7 returns to the principle to synthesise the combined evidence and its limitations.

2.12 ANN as a Baseline

Artificial Neural Networks (ANNs) occupy a central methodological role in this thesis. Although the subsequent chapters propose progressively more specialised architectures—including SOM–ANN hybridisation (Chapter 3), modular decomposition through MANN (Chapter 4), hierarchical behavioural modelling (Chapter 5), and adversarially robust detection through R-Guard (Chapter 6)—each contribution inherits core ANN principles related to non-linear representation learning, optimisation, regularisation, and evaluation. Establishing an ANN baseline within the literature review is therefore methodologically important, both to provide a critical synthesis of ANN suitability for insider-threat detection and to define a common comparative reference for the architectural developments that follow [10], [14].

This section is not intended as a generic tutorial on neural networks. Rather, it presents a thesis-oriented baseline case study and synthesis focused on insider-threat detection in complex behavioural datasets (e.g., CERT-style multi-log environments). The discussion critically examines why ANNs are attractive in this domain, where they are limited, how they should be implemented and evaluated under leakage-resistant protocols, and why later chapters require architectural extensions beyond a conventional ANN baseline [9], [10], [13].

2.12.1 ANNs for Insider Threat Detection

Insider-threat detection presents a challenging learning problem characterised by heterogeneous behavioural signals, class imbalance, temporal dependency, contextual ambiguity, and adversarial adaptation. In contrast to many traditional tabular classification tasks, insider-risk modelling must often combine multiple activity streams (e.g., authentication, device use, file access, web activity, and communic-

ation) while distinguishing benign behavioural variation from malicious intent [9], [10], [15], [22], [54].

ANNs are attractive in this setting for several reasons. First, they can model non-linear relationships between behavioural features, making them suitable for complex interactions that may not be captured by simpler linear or rule-based approaches [14], [30], [38]. Second, ANNs can be adapted to different input formulations, including aggregated behavioural features, temporal sequences, and multi-branch feature representations, which aligns with the evolving architecture progression developed in this thesis [42], [43], [55], [56], [57]. Third, ANN-based systems support flexible optimisation and regularisation strategies that can be tuned for imbalanced, noisy, and partially overlapping class distributions, which are common in insider-threat datasets [19], [58], [59], [60], [61].

However, ANN suitability should not be overstated. Insider-threat detection is not simply a high-dimensional classification problem; it is a socio-technical inference problem in which behavioural plausibility, temporal progression, and organisational context are critical [8], [12], [62]. A conventional ANN may achieve strong nominal performance while remaining vulnerable to identity memorisation, temporal leakage, poor generalisation under behavioural drift, and adversarially strategic behaviour [13], [16], [17], [32], [33]. For this reason, the ANN baseline established in this section is treated as a necessary foundation rather than a complete solution.

Critical position in this thesis. Within the thesis architecture progression, ANNs serve as the shared predictive substrate from which later innovations depart. The role of this baseline is therefore twofold: (i) to demonstrate the core strengths of ANN-based behavioural modelling, and (ii) to make explicit the limitations that motivate SOM-based stabilisation, modular decomposition, hierarchical abstraction, and adversarial robustness engineering in Chapters 3–6.

2.12.2 ANN Families and Relevance

The ANN literature includes a wide range of architectures, but not all are equally appropriate for insider-threat detection. This subsection critically reviews ANN families in relation to the structure of insider behavioural data and the modelling objectives of this thesis. A brief deep-learning overview is useful here because modern ANN practice in security analytics inherits common representational and optimisation principles across architectures [38].

2.12.2.1 Feedforward (Dense) Neural Networks

Feedforward neural networks (FFNs), also referred to as multilayer perceptrons (MLPs), provide a practical and widely used baseline for structured or aggregated

behavioural features. In insider-threat detection, FFNs are often applied to per-user or per-window feature vectors derived from activity counts, ratios, temporal summary statistics, and categorical encodings [14], [30], [63], [64].

Their strengths include simplicity, computational efficiency, and compatibility with common tabular preprocessing pipelines. FFNs are particularly useful for establishing an interpretable baseline under controlled protocols because architectural complexity can be incrementally increased (depth, width, regularisation, activation functions) while monitoring generalisation behaviour [38], [59], [60]. However, FFNs can struggle to capture long-range temporal dependencies and cross-domain behavioural progression unless such structure is engineered explicitly into the feature space [43], [55], [57].

2.12.2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are traditionally associated with image processing, but they can also be used for structured behavioural matrices, temporal windows, or local pattern extraction over transformed sequential representations. In insider-threat settings, convolutional or temporally convolutional encoders can be useful where behavioural signals are represented as ordered windows or multi-channel temporal feature maps [14], [57].

Nevertheless, CNN applicability depends strongly on representation design. Unlike vision tasks, insider behavioural logs do not naturally exhibit spatial locality in the same sense, so convolutional gains often rely on carefully constructed temporal or channel-wise structure. For this reason, convolutional approaches may be more useful as feature extractors within hybrid pipelines than as standalone universal solutions in insider-risk analytics [42], [57].

2.12.2.3 Recurrent Neural Networks (RNNs), LSTMs, and GRUs

Recurrent architectures, including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models, are directly relevant to insider-threat detection because they can model temporal dependence in behavioural sequences [14], [55], [56]. This is particularly important where malicious behaviour is gradual, staged, or context-dependent across time rather than expressed as isolated anomalous events.

In CERT-style settings, recurrent models can be applied to per-user event sequences, session-level behavioural windows, or time-ordered aggregates. Their ability to capture sequential progression makes them valuable for baseline comparisons, especially in relation to later hierarchical modelling in Chapter 5 [42], [43]. However, recurrent models remain sensitive to sequence definition, windowing strategy, class imbalance, and leakage in temporal splits. They may also be difficult to calibrate under sparse positive labels or heterogeneous multi-domain event streams [13], [19], [32], [33].

2.12.2.4 Transformer-Based and Attention-Based

Transformer and attention-based models offer a powerful framework for modelling long-range dependencies and variable interaction strengths within sequential or multi-source behavioural data [65], [66]. In insider-threat detection, such models may improve sensitivity to weak but strategically distributed behavioural patterns by learning context-dependent weighting across events and domains, particularly in multi-stream settings [14], [42].

However, transformer-style architectures introduce additional challenges, including increased computational cost, greater sensitivity to design choices in tokenisation and sequence construction, and practical data-efficiency concerns under sparse labels. In many insider-threat datasets, these constraints may limit the realised advantage of transformer models unless supported by careful regularisation and evaluation design [10], [14], [66].

2.12.2.5 Hybrid and Structured ANN Extensions

The limitations of standalone ANN families motivate hybrid and structured approaches that embed domain knowledge, representation stabilisation, modularity, or hierarchical reasoning into the learning pipeline. This motivation is central to the thesis contributions. Chapter 3 introduces SOM–ANN hybridisation to improve behavioural representation stability; Chapter 4 introduces modular decomposition to preserve domain semantics; Chapter 5 introduces multi-layered abstraction for behavioural progression; and Chapter 6 addresses adversarial robustness through invariant- and consistency-based supervision.

From a literature-review perspective, these later architectures should be read as ANN extensions designed to address insider-specific failure modes rather than as departures from ANN-based modelling altogether [17], [42], [43], [62].

Table 2.9 summarises the major ANN family types relevant to insider-threat detection, highlighting their typical input assumptions, strengths, limitations, and practical suitability for behavioural security analytics.

As shown in Table 2.9, no single ANN family fully addresses the representational, temporal, and robustness requirements of insider-threat detection, which motivates the structured architectural progression developed in Chapters 3–6.

2.12.3 ANN and CERT Implementation

A meaningful ANN baseline for insider-threat detection depends less on architectural novelty than on disciplined implementation choices. In particular, preprocessing, temporal split integrity, class imbalance handling, and feature construction can influence results as strongly as the ANN architecture itself [10], [13], [14].

Table 2.9: Summary of ANN families and their suitability for insider-threat detection

ANN family	Typical input	Strengths for insider detection	Limitations / risks	Typical use cases
Feedforward (MLP)	Aggregated behavioural feature vectors (e.g., per user-/window)	Strong non-linear baseline for tabular behavioural features; efficient to train; easy to tune and compare	Limited native temporal modelling; sensitive to feature engineering and leakage	Baseline classification on engineered CERT-style behavioural features
Convolutional / Temporal Convolution	Ordered windows, temporal feature maps, multi-channel behavioural matrices	Captures local temporal patterns; useful for structured windowed behaviour	Requires careful representation design; weaker fit when locality assumptions are unclear	Window-based behavioural pattern extraction; temporal feature encoding
RNN / LSTM / GRU	Event sequences, session sequences, time-ordered behavioural windows	Models temporal dependencies and staged behaviour progression	Sensitive to sequence construction, imbalance, and temporal split leakage; harder to calibrate	Sequential anomaly/risk modelling; gradual insider behaviour detection
Transformer / Attention-based	Tokenised sequences or multi-stream behavioural events	Captures long-range dependencies and context-dependent interactions	Higher computational cost; data-hungry; design-sensitive under sparse labels	Long-horizon behavioural context modelling; multi-stream interaction modelling
Hybrid / Structured ANN (e.g., SOMANN, modular, hierarchical)	Multi-domain behavioural representations with structured preprocessing or decomposition	Improves stability, modularity, and interpretability; better alignment with insider data heterogeneity	More complex design and evaluation; requires stronger reporting discipline	Behaviourally structured detection pipelines; thesis progression architectures

2.12.3.1 Data Integration and Behavioural Feature Construction

CERT-style insider datasets typically contain multiple logs (e.g., logon, device, HTTP, email, file activity, psychometric/user metadata) that must be aligned into a behavioural representation suitable for learning [13], [18]. Baseline ANN implementation requires explicit decisions regarding:

- event granularity (event-level, session-level, daily/weekly windows),
- user-level aggregation and identity handling,
- categorical encoding (one-hot, target encoding, embeddings),
- temporal features (time-of-day, day-of-week, after-hours ratios),
- cross-domain behavioural counts and ratios,
- treatment of missing or sparse activity streams.

These design choices should be documented and justified, as they directly influence what the ANN can learn and how later comparisons with SOMANN, MANN, and hierarchical models should be interpreted [10], [13].

2.12.3.2 Preprocessing, Normalisation, and Leakage Control

Preprocessing and evaluation integrity are especially critical in insider-threat research. Strong performance can be artificially inflated if temporal leakage, user overlap leakage, or label-proximal feature leakage is not carefully controlled [13]. For this reason, ANN baselines should be implemented under chronology-preserving splits (and, where appropriate, blocked temporal folds) that prevent future information from influencing model training, with drift-aware interpretation of performance over time [32], [33], [67].

Feature scaling and normalisation should be fitted on training data only and then applied to validation/test partitions. If user-level aggregation is used, care should be taken to avoid identity memorisation effects that allow the model to exploit user-specific signatures rather than behavioural risk patterns [13], [62]. These considerations are central to the methodological rigour of this thesis and remain relevant throughout Chapters 3–6.

2.12.3.3 Class Imbalance and Label Scarcity

Insider-threat datasets are typically highly imbalanced, with malicious or policy-violating cases representing a small minority of observations [10], [14]. ANN baselines must therefore address imbalance explicitly through a combination of techniques such as:

- class weighting,
- resampling (with caution),
- focal or cost-sensitive loss functions,
- threshold tuning for deployment-oriented precision–recall trade-offs,
- stratified reporting where temporal protocols permit.

No single strategy is universally optimal. The appropriate choice depends on the operational goal (e.g., early detection, false-positive control, analyst triage efficiency) and the structure of the evaluation protocol. In heavily imbalanced settings, PR-oriented analysis is often more informative than ROC-focused interpretation alone [10], [19].

2.12.4 Optimising, Tuning, and Evaluating ANN

To serve as a robust comparative reference, the ANN baseline should be tuned systematically and evaluated under a protocol that reflects insider-threat deployment realities rather than benchmark convenience.

2.12.4.1 Architecture and Optimisation Tuning

Key ANN tuning dimensions include:

- number of hidden layers and units,
- activation functions,
- optimiser selection and learning-rate scheduling,
- batch size and training duration,
- regularisation (dropout, weight decay, early stopping),
- calibration and threshold selection.

Hyperparameter tuning should be conducted using validation data that respects the temporal and user-level separation assumptions of the main evaluation protocol. Where extensive search is not feasible, a principled and reproducible tuning strategy (e.g., bounded grid search or staged tuning) is preferable to opportunistic parameter selection [58], [59], [60], [61].

2.12.4.2 ANN Evaluation Metrics

Evaluation of ANN baselines should use metrics aligned with class imbalance and operational decision-making. In addition to accuracy, the following are typically more informative in insider-threat settings:

- Precision, Recall, and F_1-score,
- ROC–AUC,
- PR–AUC (especially under severe imbalance),
- false-positive rate and false-negative rate,
- time-to-detection or delay-sensitive indicators (where temporal evaluation is performed).

Metrics should be interpreted in relation to the deployment objective. For example, a model with strong ROC–AUC but weak precision at operational thresholds may be unsuitable in settings where analyst capacity is limited. This distinction becomes increasingly important in the later chapters, particularly in the robustness and false-positive analyses of Chapter 6 [10], [16], [19].

2.12.4.3 Baseline Reproducibility and Reporting Discipline

A recurring issue in insider-threat detection research is the difficulty of comparing reported results due to inconsistent preprocessing, undocumented feature engineering, unclear split protocols, and selective metric reporting [9], [10], [13]. To mitigate this, the ANN baseline case study in this thesis is framed not only as a predictive benchmark but also as a reporting discipline. At minimum, baseline reporting should document:

- dataset scope and log sources used,
- feature construction choices,
- split protocol (including chronology and user separation assumptions),
- hyperparameters and training configuration,
- metric definitions and threshold selection,
- reproducibility assumptions (random seeds, library versions, hardware constraints).

This reporting discipline is intended to support valid comparison with the thesis contributions and to reduce ambiguity when interpreting improvements in later chapters.

2.12.5 Case Study Baseline

To anchor the literature synthesis in a concrete methodological reference, this subsection presents a compact ANN baseline case-study framing for insider-threat detection on a CERT-style multi-log dataset. The purpose is not to duplicate the full experimental detail of later chapters, but to establish a clear baseline paradigm against which subsequent architectural contributions can be interpreted [10], [13], [18].

2.12.5.1 Case Study Objective and Scope

The baseline case study is designed to answer the following question: *What level and type of performance can be achieved by a carefully implemented ANN baseline under leakage-resistant, temporally valid evaluation conditions on CERT-style insider behavioural data?* This question is intentionally narrower than the thesis-wide research questions, but it is foundational because it defines the reference point from which improvements in representation stability, modularity, hierarchy, and adversarial robustness are evaluated.

The case-study scope may include:

- a feedforward ANN baseline over engineered behavioural features,
- optionally, a recurrent ANN baseline for temporal comparison,
- common preprocessing and feature definitions reused (or adapted) in later chapters,
- consistent metric reporting for comparative interpretation.

2.12.5.2 Baseline Implementation Outline

An appropriate ANN baseline implementation should include the following stages:

1. multi-log data preparation and behavioural feature construction;
2. chronology-preserving train/validation/test partitioning (with blocked folds where appropriate);
3. ANN model configuration and hyperparameter tuning under validation constraints;
4. class-imbalance handling and threshold calibration;
5. evaluation using imbalance-aware and deployment-relevant metrics;
6. qualitative and quantitative interpretation of error behaviour.

This baseline outline is intentionally aligned with the methodological safeguards and evaluation principles used throughout the thesis, thereby supporting coherent cross-chapter comparison [13], [32], [33].

2.12.5.3 Expected Strengths and Limitations

A well-tuned ANN baseline is expected to provide a strong non-linear benchmark for tabular or aggregated behavioural representations, particularly when compared with simpler machine-learning baselines [14], [30], [38]. However, several limitations are expected to remain:

- sensitivity to feature engineering choices and dataset-specific preprocessing;
- limited built-in modelling of domain semantics across heterogeneous logs;
- insufficient representation of multi-level behavioural progression;
- vulnerability to behavioural drift and strategically adaptive insider actions;
- potential opacity in analyst-facing interpretation without additional explanation mechanisms.

These limitations are not weaknesses of the thesis design; rather, they constitute the rationale for the successive architectural advances proposed in Chapters 3–6 [16], [42], [43], [62], [68].

2.12.6 Implications

The ANN baseline case study and literature synthesis establish the methodological foundation upon which the remainder of the thesis is built. In particular, the baseline clarifies that later contributions should be interpreted as targeted responses to specific ANN limitations in insider-threat detection rather than as unrelated architectural substitutions.

Chapter 3 addresses representational instability and noisy behavioural structure through SOM-based topological pre-structuring prior to ANN classification. Chapter 4 addresses domain heterogeneity and feature dominance through modular decomposition and principled fusion. Chapter 5 addresses the need for behavioural abstraction across multiple temporal and semantic levels. Chapter 6 addresses adversarial vulnerability by introducing behaviourally grounded robustness supervision through invariants, temporal coherence, and cross-domain consistency checks.

By positioning ANN explicitly as a baseline in the literature review, the thesis strengthens cross-chapter continuity and provides a consistent reference for interpreting gains, trade-offs, and limitations across the proposed frameworks [10], [14], [17], [42].

Figure 2.11 summarises the architectural progression adopted in this thesis, showing how each proposed framework extends the ANN baseline to address a specific limitation in insider-threat detection.

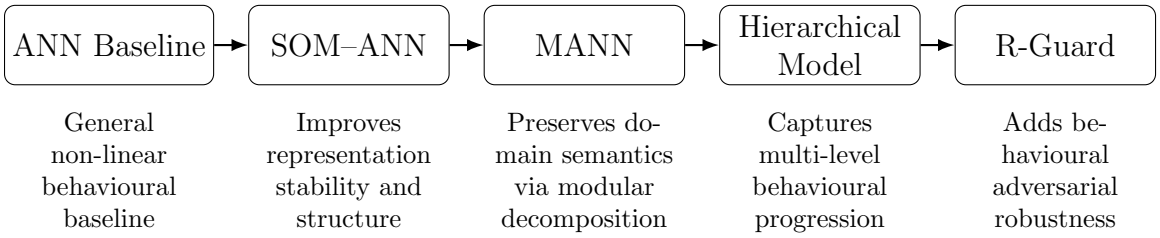


Figure 2.11: Thesis architecture progression from a conventional ANN baseline to progressively structured and robust insider-threat detection frameworks. Each stage addresses a specific limitation of the preceding modelling configuration, moving from general non-linear behavioural classification toward modularity, hierarchical behavioural abstraction, and adversarial robustness.

Figure 2.11 summarises the cumulative design logic of the thesis. Rather than replacing ANN-based modelling at each stage, the proposed frameworks progressively refine it by introducing topological structure (SOM–ANN), domain-aware modularity (MANN), hierarchical behavioural abstraction, and behaviourally grounded adversarial robustness (R-Guard). This progression provides a consistent basis for interpreting comparative results, trade-offs, and methodological contributions across Chapters 3–6.

2.12.7 Synthesis and Transition

This section has established Artificial Neural Networks as a unifying baseline paradigm for insider-threat detection within the thesis. It has argued that ANNs are well suited to modelling non-linear behavioural patterns in heterogeneous security data, but that their effectiveness depends critically on disciplined implementation, leakage-resistant evaluation, and imbalance-aware performance assessment [10], [13], [19]. It has also shown that ANN baselines, while necessary and informative, remain insufficient to address several core insider-threat challenges, including behavioural structure, domain semantics, temporal hierarchy, and adversarial adaptation [16], [17], [43], [62].

These observations provide the conceptual and methodological bridge to the architectural contributions developed in Chapters 3–6 and motivate the subsequent discussion of ethical, governance, and responsible deployment considerations in insider-threat detection.

While the preceding section established ANNs as a unifying modelling baseline, the validity and interpretability of ANN-based insider-threat detection are equally dependent on the characteristics of the datasets used for training and evaluation.

The next section therefore reviews insider-threat datasets more broadly and provides a critical justification for the use of the CERT dataset as the common experimental benchmark throughout this thesis.

2.13 Insider-Threat Datasets

While the preceding section established Artificial Neural Networks (ANNs) as a unifying modelling baseline for this thesis, the validity and interpretability of ANN-based insider-threat detection depend equally on the properties of the datasets used for training and evaluation. In insider-threat research, dataset characteristics strongly shape what can be learned, what can be compared, and what claims can be made about generalisation, robustness, and operational relevance [9], [10], [14]. This is particularly important in a thesis such as the present work, where multiple architectural contributions (SOM–ANN, MANN, hierarchical modelling, and R-Guard) are evaluated under a common experimental backbone.

Accordingly, this section reviews the insider-threat dataset landscape, defines criteria for evaluating dataset suitability, and provides a critical justification for the use of the CERT dataset as the common benchmark throughout this thesis. The aim is not to claim that CERT is a perfect proxy for real-world insider behaviour, but rather to justify it as a defensible and reproducible benchmark for comparative architectural research under controlled conditions [13], [18].

2.13.1 The Role of Datasets

Datasets are not a neutral backdrop in insider-threat detection; they are a central methodological variable. The observable behavioural signals available, the quality and granularity of labels, temporal continuity, and the degree of behavioural realism all directly influence the performance and interpretation of machine-learning and deep-learning models [9], [10], [22]. In practice, differences in dataset construction can produce substantial variation in reported results even when similar model families are used [13], [14].

This issue is amplified in insider-threat detection because the target phenomenon is inherently socio-technical. Malicious insider behaviour may be sparse, gradual, context-dependent, and difficult to distinguish from benign policy deviations or legitimate but unusual work patterns [8], [12], [15]. As a result, dataset limitations can lead to over-optimistic estimates of predictive performance or to conclusions that do not transfer to operational settings. For this reason, a rigorous thesis should make dataset selection criteria explicit and should interpret all comparative results in light of known data constraints.

2.13.2 Types of Datasets

The insider-threat literature relies on several broad categories of datasets, each with distinct advantages and limitations.

2.13.2.1 Synthetic and Simulated Datasets

Synthetic or simulated datasets are widely used in insider-threat research because genuine organisational insider data are difficult to collect, sensitive to legal and privacy restrictions, and rarely shareable for reproducible research [9], [10]. Such datasets can provide labelled behavioural logs across multiple activity streams and support controlled experimentation, benchmarking, and method comparison.

However, synthetic datasets may not fully capture the complexity, ambiguity, and evolving context of real organisational behaviour. Behavioural scripts used to generate malicious scenarios can introduce regularities that are easier for models to learn than real-world insider strategies, thereby affecting ecological validity [8], [13]. These limitations do not invalidate synthetic benchmarks, but they require careful interpretation of results and restraint in claims about real-world deployment performance.

2.13.2.2 Real Organisational and Proprietary Datasets

Real organisational datasets, where available, may offer richer behavioural realism and stronger ecological validity. They can better reflect authentic organisational processes, policy structures, and contextual variation. However, such datasets are usually restricted, anonymised under strong constraints, incompletely labelled, or inaccessible to the wider research community [9], [10]. This limits reproducibility and makes cross-study comparison difficult.

In addition, operational datasets often contain substantial governance and ethical challenges, including privacy, consent, proportionality, and the risk of using behavioural monitoring outputs in punitive or discriminatory ways. These concerns are especially salient in insider-threat research and are considered further in the ethics section of this chapter.

2.13.2.3 Public Benchmarks

Publicly accessible benchmark datasets provide a practical compromise between realism and reproducibility. They enable comparative evaluation across models, support transparent reporting, and allow other researchers to replicate or challenge reported findings. For insider-threat detection, the CERT dataset has become one of the most widely used public benchmark resources, particularly for behavioural log-based modelling [14], [18], [30].

Nevertheless, benchmark popularity alone should not be treated as sufficient justification for dataset choice. A rigorous literature review must also consider known benchmark weaknesses, misuse patterns, and evaluation pitfalls, especially when the dataset becomes a de facto standard [10], [13].

2.13.3 Criteria for Evaluating Insider-Threat Datasets

To justify dataset choice in a thesis-wide comparative programme, it is useful to define explicit criteria for evaluating insider-threat datasets. The criteria below are used in this thesis to assess dataset suitability for model development, comparison, and interpretation.

2.13.3.1 Behavioural Realism and Contextual Plausibility

A useful insider-threat dataset should capture behavioural patterns that are sufficiently plausible to support meaningful modelling of insider risk. This includes variation in routine user behaviour, role-dependent activity profiles, and plausible transitions between benign and potentially malicious actions [15], [22], [62]. Datasets with highly scripted or overly separable malicious behaviour may inflate nominal detection performance while underrepresenting real operational ambiguity.

2.13.3.2 Temporal Continuity and Sequential Integrity

Because insider behaviour can unfold gradually over time, temporal continuity is essential for evaluating sequential models, drift-aware methods, and early-warning detection strategies [32], [33], [56]. Datasets that preserve chronology across user activity streams are more suitable for temporally valid train/validation/test protocols and for analysing time-sensitive metrics such as detection delay.

For this thesis, temporal integrity is particularly important because later chapters evaluate hierarchical behavioural progression and adversarial robustness under behaviourally constrained perturbations.

2.13.3.3 Multi-Modal / Multi-Log Coverage

Insider-threat detection is inherently multi-domain. A dataset is more useful for thesis-wide architectural comparison when it includes multiple activity sources (e.g., authentication, device activity, web browsing, email, file operations, and user metadata), thereby allowing evaluation of feature integration, modular decomposition, and cross-domain consistency [14], [18], [42]. Limited-modality datasets may still support specific tasks, but they constrain claims about behaviourally rich detection pipelines.

2.13.3.4 Label Availability, Granularity, and Class Imbalance

Dataset suitability also depends on whether labels exist, what they represent, and how they align with the research objective. In insider-threat detection, labels may reflect scenario scripts, policy violations, risk categories, or inferred malicious outcomes. The granularity and reliability of labels influence which models and metrics are appropriate [9], [10], [14].

In addition, severe class imbalance is typical in insider-threat settings. A suitable dataset should therefore support imbalance-aware evaluation and threshold-sensitive interpretation rather than encouraging reliance on accuracy alone [10], [19].

2.13.3.5 Accessibility and Reproducibility

For a thesis that develops multiple architectures under a common benchmark, dataset accessibility and reproducibility are critical. Public availability, stable schema, and widespread prior use improve comparative utility and make it easier to position contributions within existing literature [14], [18], [30]. Reproducibility is particularly valuable when evaluating architectural progression, since observed gains should be attributable to model design rather than changing datasets or undocumented preprocessing differences.

2.13.3.6 Ethical and Governance Suitability

Even benchmark datasets raise ethical and governance questions. Dataset choice should consider privacy implications, representational bias, potential misuse, and the extent to which results might be overgeneralised to real employees or organisations [8], [9], [12]. These concerns do not preclude the use of benchmark datasets, but they should shape how findings are reported and bounded.

2.13.4 Critical Review of the CERT Dataset

The CERT insider-threat dataset is one of the most widely used public resources for behavioural insider-threat detection research and has been adopted in numerous machine-learning and deep-learning studies [14], [18], [30], [63]. Its widespread use makes it a natural reference point for comparative evaluation, particularly in work concerned with methodological progression rather than one-off model demonstrations.

Tables 2.10 and 2.11 translate the dataset evaluation criteria into thesis-specific methodological decisions, showing how CERT strengths and limitations are handled across the experimental chapters.

Table 2.10: CERT dataset criteria and mitigation strategies: technical and methodological considerations

Evaluation criterion	Why it matters	CERT strengths / limitations	Mitigation or handling in this thesis
Behavioural realism and contextual plausibility	Determines whether model behaviour reflects meaningful insider-risk patterns rather than artefacts of simulation design	CERT provides multi-log behavioural scenarios but remains synthetic and may not capture the full ambiguity and adaptation of real insiders [8], [13], [18]	Results are interpreted as benchmark-grounded comparative evidence; claims are bounded and transferability is discussed explicitly in Chapter 7 [9], [10]
Temporal continuity and chronology	Essential for valid sequential modelling, drift-aware analysis, and time-sensitive evaluation	CERT supports chronological ordering; however, unrealistic splits can introduce leakage or over-optimistic evaluation [13], [18]	Chronology-preserving splitting is used throughout; blocked temporal folds are applied where appropriate; training-only fitting of preprocessing is enforced [13], [32], [33]
Multi-log / multi-domain coverage	Supports modular, hierarchical, and cross-domain consistency modelling	CERT includes multiple behavioural sources (logon/device/HTTP/e-mail/file) plus user/context metadata [18]	A constant multi-log backbone is maintained across chapters; domain-specific and fused representations are evaluated under consistent reporting assumptions [14], [42]
Labels and class imbalance	Influences model training, metric selection, thresholding, and interpretability under sparse positives	CERT supports scenario-based labels but retains severe imbalance; labels may not capture all real-world ambiguity [10], [18]	Imbalance-aware evaluation is used (PR-oriented analysis where appropriate); threshold-dependent behaviour is discussed in relation to false positives and analyst workload [10], [19]

Table 2.11: CERT dataset criteria and mitigation strategies: validity, reproducibility, and ethical considerations

Evaluation criterion	Why it matters	CERT strengths / limitations	Mitigation or handling in this thesis
Reproducibility and comparative utility	Enables fair attribution of improvements to architecture rather than dataset changes	CERT is public and widely used, supporting replication and cross-study comparison [14], [18], [30]	CERT is used as the constant benchmark across Chapters 3–6; preprocessing and evaluation assumptions are documented to strengthen comparability [10], [13]
Evaluation pitfalls (leakage, split bias, over-optimistic reporting)	Poor evaluation design can inflate performance and undermine comparative claims	CERT studies are vulnerable to leakage and unrealistic evaluation if methodological safeguards are absent [13]	The thesis adopts explicit evaluation discipline (chronology-aware splitting, controlled preprocessing, transparent metric reporting) and states validity boundaries for comparative claims [9], [13]
Ethical and governance implications	Behavioural analytics can affect individuals and organisations if misused or overgeneralised	CERT reduces direct privacy exposure relative to real employee datasets, but findings may still be misapplied if ethical boundaries are not stated [8], [12]	Ethical use notes and deployment boundaries are stated; outputs are framed as decision support rather than autonomous punitive evidence; safeguards against misuse are emphasised [9], [10]

As shown in Tables 2.10 and 2.11, the use of CERT in this thesis is accompanied by explicit methodological safeguards and interpretive boundaries, allowing the dataset to function as a reproducible benchmark for comparative architectural research while avoiding overstated claims about operational deployment performance.

2.13.4.1 Strengths of CERT for Comparative Research

CERT offers several strengths that are directly relevant to this thesis. First, it provides multi-log behavioural data (including user/context information and multiple activity streams), making it suitable for evaluating feature integration, modular modelling, and cross-domain consistency [18]. Second, its public availability supports reproducibility and facilitates comparison with prior studies in the literature [10], [14], [30]. Third, its structure is sufficiently rich to support multiple modelling paradigms, including tabular ANN baselines, sequential variants, modular pipelines, hierarchical behavioural representations, and adversarial robustness experiments.

From the perspective of thesis design, these properties make CERT especially useful as a constant benchmark across Chapters 3–6, enabling architectural comparisons under a shared data regime rather than confounding model changes with dataset changes.

2.13.4.2 Synthetic Artefacts and Architectural Validity

The CERT insider-threat benchmark provides an important experimental advantage for this thesis because it combines multi-source organisational telemetry with known malicious scenarios, enabling controlled and reproducible comparison of behavioural detection architectures. This property is particularly valuable where equivalent labelled operational insider-threat data are difficult to obtain. However, the same characteristic introduces an important validity constraint. As a synthetic and simulated dataset, CERT cannot be assumed to reproduce the full behavioural ambiguity, organisational heterogeneity, or strategic adaptation encountered in operational insider incidents [8], [13].

Consequently, results obtained using CERT are interpreted in this thesis as evidence about the comparative behaviour of the proposed architectures under a controlled benchmark, rather than as direct estimates of deployment-level performance. This distinction is particularly important because methodological pitfalls identified in CERT-based research, including preprocessing choices, information leakage, unrealistic data splitting, and evaluation designs, can produce overly favourable estimates of detection performance [13]. Particular attention is therefore given to dataset characteristics that could produce artificially favourable results and to the experimental controls used to reduce dependence on such characteristics.

Risks introduced by the synthetic structure of CERT. Four principal risks are relevant to the interpretation of the experimental results.

Scripted malicious narratives. CERT malicious scenarios are constructed according to predefined behavioural narratives. This provides the ground truth required for controlled evaluation, but it also creates the possibility that a model learns recurring scenario-specific regularities, including characteristic event combinations, ordering, or timing, rather than more transferable behavioural relationships. A detector that relies strongly on such regularities could perform well within the benchmark while degrading when malicious objectives are expressed through different behavioural trajectories.

Synthetic benign-behaviour regularity. The benign population is generated rather than obtained from unrestricted operational activity. Real organisational environments contain workload surges, staff substitution, emergency activity, seasonal variation, policy changes, software migrations, changes in working location, and other sources of legitimate behavioural variability. The broader difficulty of reproducing the ambiguity and heterogeneity of operational insider behaviour in synthetic benchmarks has been recognised in prior work [8], [13]. To the extent that these sources of variation are less extensively represented in CERT, estimates of false-positive performance may therefore be more favourable than those obtained in a live operational environment.

Limited organisational and role diversity. CERT represents a simulated organisational environment with a finite set of users, roles, resources, workflows, and behavioural assumptions. Relationships learned between behavioural domains may therefore partly reflect structural properties of the simulated organisation. This limitation is particularly relevant when interpreting domain-grounded and hierarchical models, for which role context, domain interaction, and behavioural progression form part of the modelling rationale.

Restricted natural concept drift. Although CERT provides temporally ordered telemetry, it cannot reproduce the full range of unscripted organisational drift encountered in long-running enterprise environments. Operational users may change responsibilities, projects, work locations, devices, managers, access privileges, and working patterns. Evaluation within CERT may therefore expose models to a narrower range of legitimate distributional change than would be expected in deployment.

These characteristics mean that strong performance on CERT cannot, by itself, establish that equivalent numerical performance would transfer to another organisation. The relevant methodological question for this thesis is instead whether the experimental design provides evidence that the observed architectural effects are not explained solely by obvious synthetic regularities or evaluation leakage.

Safeguards against dataset-specific artefacts. Several complementary safeguards are used across the experimental chapters to reduce dependence on trivial, identity-specific, or temporally leaked structure within the benchmark.

Chronology-preserving evaluation. Prior CERT-based research has been criticised for evaluation practices that can introduce information leakage or produce unrealistically favourable performance estimates, including inappropriate preprocessing and data-splitting strategies [13]. In this thesis, training, validation, and test partitions are therefore separated chronologically rather than through unrestricted random sampling. This preserves the direction of time between model development and evaluation and prevents future behavioural observations from being used to construct earlier training partitions. Chronological evaluation does not remove the synthetic assumptions of CERT, but it provides a more demanding assessment of forward generalisation within the benchmark and reduces one important source of temporal leakage.

Training-only preprocessing and feature construction. Consistent with the need to avoid preprocessing-induced leakage identified in CERT-based evaluation [13], fitted preprocessing components, including scaling parameters, categorical vocabularies, and applicable feature transformations, are derived from the training partition and subsequently applied to validation and test data. This prevents information from later partitions from influencing model preparation and reduces the possibility that dataset-wide statistics artificially improve test performance.

Identity-independent and component-level evaluation. The technical chapters supplement aggregate classification performance with tests targeted at the architectural hypotheses under investigation. Chapter 3 examines identity dependence and out-of-user generalisation to determine whether the learned representation remains informative beyond users observed during training. Chapter 4 uses behavioural-domain ablation and fusion analysis to examine the contribution of domain-specific structure. Chapter 5 evaluates whether hierarchical temporal modelling contributes information that is not available to flatter, shorter-horizon representations. These experiments do not eliminate CERT-specific structure, but they provide additional evidence concerning the mechanisms responsible for observed model performance.

Behaviourally constrained perturbation testing. Chapter 6 extends the evaluation by deliberately modifying selected observable characteristics of malicious behaviour within defined feasibility constraints. Perturbations to timing, ordering, and related behavioural characteristics disrupt aspects of the original observed trajectories while retaining the underlying malicious objective represented by the experimental threat model. Persistence, degradation, and recovery of performance under these conditions provide additional evidence concerning the extent to which detection depends on brittle properties of the unmodified benchmark sequences.

These safeguards should not be interpreted as demonstrating that synthetic dataset artefacts have been eliminated. Rather, they address identifiable threats to internal validity and provide a stronger basis for determining whether differences between the evaluated architectures are consistent with the modelling properties under investigation.

Scope of the architectural conclusions. The conclusions drawn from CERT are deliberately separated into *benchmark-specific numerical findings* and *architectural findings*. The thesis does not claim that a PR-AUC, F1-score, false-positive rate, or adversarial robustness value measured on CERT will reproduce unchanged in a live organisation. Such quantities depend on factors including threat prevalence, telemetry coverage, organisational structure, user roles, security policy, feature availability, and the extent of legitimate behavioural variability.

The architectural findings concern narrower questions tested under controlled and matched experimental conditions. These include whether topology-informed representation improves stability and generalisation relative to the common ANN baseline; whether domain-specific decomposition provides useful structure for heterogeneous telemetry; whether temporal hierarchy recovers discriminative information that is unavailable to short-horizon representations; and whether robustness mechanisms reduce performance degradation under the behaviourally constrained perturbations evaluated in Chapter 6.

CERT provides a controlled environment in which these architectural hypotheses can be compared using common preprocessing principles, matched evaluation con-

ditions, and chronology-preserving partitions. The resulting evidence can therefore support comparative conclusions within the evaluated benchmark. It cannot establish that the magnitude of those effects, or the resulting operational performance, will remain unchanged across independent organisations or telemetry environments.

External validity consequently remains a recognised limitation rather than a problem claimed to have been resolved. The broader implications of this limitation are considered further in Section 2.13.6, which defines the boundaries within which the benchmark-derived conclusions of this thesis should be interpreted.

2.13.4.3 Interpretive Implications

Accordingly, CERT is used in this thesis to support controlled comparison of *relative architectural effects*, *design trade-offs*, and *robustness characteristics*, rather than to estimate deployment-level performance. Translation of these findings to operational insider-threat detection would require validation across independent organisational settings, together with consideration of contextual and governance constraints not fully represented by a synthetic benchmark [9], [10], [13].

2.13.5 Rationale for Selecting CERT

Given the criteria above, the CERT dataset is selected as the common benchmark for this thesis because it best supports the methodological goals of a multi-stage comparative architectural programme. Specifically, the thesis requires a dataset that is: (i) behaviourally structured enough to support feature engineering and sequential analysis; (ii) multi-log in nature to support modular and cross-domain modelling; (iii) temporally analysable to support progression and drift-sensitive evaluation; and (iv) publicly accessible to support reproducibility and comparison with prior work [14], [18], [30].

This rationale is closely aligned with the design of the thesis contributions. The ANN baseline section establishes a common predictive substrate; Chapters 3–5 progressively refine that substrate through structured representation, modularity, and hierarchy; and Chapter 6 evaluates robustness under behaviourally constrained adversarial settings. Maintaining a constant dataset across these stages allows the thesis to attribute observed changes more credibly to architectural design decisions rather than to changes in data source or label regime.

At the same time, the thesis does not treat CERT as exhaustive or universally representative. Instead, it is treated as a reproducible benchmark for controlled comparative research, with explicit validity boundaries that are revisited in the discussion and conclusion chapters.

2.13.6 Validity Boundaries

The use of a common benchmark dataset strengthens internal comparability across the thesis but also introduces constraints on transferability. Results obtained on CERT can support claims about comparative model behaviour, relative performance trends, and robustness under the defined evaluation protocols; however, they cannot by themselves establish effectiveness across all organisational contexts, sectors, or insider threat typologies [8], [9], [13].

These validity boundaries are especially important when interpreting adversarial robustness findings in Chapter 6. Behaviourally constrained perturbation models can be rigorously studied on CERT, but real-world adversaries may operate under different organisational constraints, tooling environments, or policy structures. For this reason, the thesis frames its claims in terms of benchmark-grounded methodological contributions and advocates future evaluation on additional datasets and operationally governed environments.

2.13.7 External Validity and Organisational Adaptation

The validity boundaries established in the preceding discussion raise a separate question: which aspects of the proposed architectural framework are conceptually transferable beyond CERT, and which would require organisation-specific adaptation before operational deployment? The frameworks developed in this thesis are not presented as deployment-ready systems in their current experimental form. Rather, they investigate architectural mechanisms whose operational applicability would depend on the characteristics of the target organisation, including telemetry availability, role structure, policy constraints, behavioural drift, class prevalence, and security-operations workflows.

A central distinction is therefore made between *architectural principles* and *deployment configurations*. The former concern the modelling requirements investigated experimentally in this thesis, whereas the latter include feature schemas, model parameters, decision thresholds, domain availability, organisational context, and operational controls that would need to be established locally. Conceptual transferability should not, therefore, be interpreted as evidence that a model trained or configured on CERT could be transferred directly into an operational environment.

Transferable architectural principles. Several principles investigated in the thesis are not inherently specific to the CERT benchmark. First, organisational telemetry is characterised by substantial variation across users, roles, workloads, and time. The SOMANN investigation therefore motivates a potentially transferable design principle: behavioural observations may benefit from structured representation before supervised discrimination. What may transfer is the principle

of separating behavioural organisation from subsequent inference, rather than the specific SOM topology, prototypes, or hyperparameters learned from CERT.

Second, enterprise security telemetry is inherently heterogeneous. Authentication, file access, communication, web activity, endpoint events, and network activity originate from different operational processes and may carry different behavioural meanings. The MANN principle of representing behavioural domains separately before controlled fusion is therefore conceptually applicable beyond CERT. However, the definition of those domains, their feature schemas, encoders, and fusion configuration would require adaptation to the telemetry available within the target organisation.

Third, behavioural significance may develop across multiple temporal horizons. The hierarchical modelling principle investigated in Chapter 5 is therefore potentially transferable to environments in which isolated observations acquire greater significance when interpreted as part of longer behavioural trajectories. The appropriate observation windows and aggregation levels would nevertheless need to reflect the operational rhythms and detection requirements of the deployment environment.

Finally, the adversarial perspective investigated in Chapter 6 is conceptually relevant beyond CERT because a detector deployed in an operational environment may encounter users who adapt observable behaviour in response to monitoring. However, the perturbation operators, feasibility constraints, and robustness parameters evaluated in this thesis are specific to the defined experimental threat model. Their operational equivalents would therefore need to be derived from the capabilities, privileges, workflows, and constraints relevant to the target organisation.

Telemetry ingestion and feature-schema adaptation. CERT provides behavioural logs in a comparatively structured experimental form. An operational Security Operations Centre (SOC), by contrast, may receive heterogeneous telemetry from identity systems, endpoint platforms, file services, email infrastructure, proxy services, cloud environments, and network-monitoring systems. Deployment would therefore require an ingestion and normalisation layer capable of mapping these heterogeneous event sources into behaviourally consistent representations.

Such a layer would need to accommodate differences in event schemas, timestamp resolution, missing fields, duplicated observations, device identifiers, user-account aliases, and cross-platform naming conventions. The feature representation used in this thesis should consequently be viewed as a benchmark-specific implementation of the broader behavioural modelling principles. Operational use would require local feature engineering and validation to determine whether the resulting representations retain the relevant meaning of the organisation's own telemetry rather than reproducing CERT-specific assumptions.

Adaptation under organisational drift. Operational deployment of the SOM component would require explicit consideration of changes in the distribution of legitimate behaviour. Organisational restructuring, seasonal workload, policy changes, remote or hybrid working patterns, software migration, and changes in staff responsibilities may alter the behavioural distributions against which new activity is interpreted.

A deployed SOMANN implementation would therefore require a controlled strategy for maintaining or recalibrating its behavioural representation. Potential approaches include periodic retraining, incremental or online SOM updating, or retraining triggered by validated indicators of distributional change. The appropriate mechanism would depend on the rate, magnitude, and operational meaning of behavioural change within the target organisation.

Continuous online adaptation was not evaluated in the experiments reported in this thesis and is therefore not claimed as a capability of the present SOMANN implementation. Any operational updating mechanism would additionally require safeguards against contamination of the learned representation, particularly where malicious or compromised behaviour could otherwise be incorporated into an evolving model of legitimate activity.

Missing and unequal telemetry domains. The modular structure of MANN provides a conceptual basis for adapting the framework to environments in which telemetry sources differ in availability, coverage, or quality. Because behavioural domains are represented separately before fusion, possible operational extensions could include domain masking, confidence-aware weighting, explicit missing-domain indicators, or adaptive fusion mechanisms.

The experiments reported in this thesis do not, however, establish that MANN is intrinsically robust to missing telemetry channels. An operational implementation would therefore need to evaluate failure modes associated with temporary sensor outages, permanently unavailable domains, unequal sampling rates, incomplete event capture, and differences in feature quality across data sources. Techniques such as modality dropout, domain masks, or confidence-aware fusion represent potential extensions for future evaluation rather than capabilities demonstrated by the present experiments.

Threshold, class-prior, and risk-score recalibration. The prevalence and construction of labelled malicious activity within CERT should not be assumed to reproduce the base rate of confirmed insider incidents within an operational organisation. Classification thresholds, posterior probabilities, and risk scores derived from the benchmark therefore cannot be transferred directly into production without local validation and recalibration.

Operational deployment would require threshold selection to reflect local class

prevalence, analyst capacity, and the relative consequences of false positive and false negative decisions. Where probabilistic outputs are used as risk estimates, their calibration would also need to be assessed against the target data distribution. The architectures evaluated in this thesis should consequently be regarded as behavioural detection and ranking mechanisms rather than complete operational decision systems. Their outputs would require integration with contextual enrichment, analyst triage, and broader security-operations processes.

Role and policy context. The behavioural meaning of an event depends partly on local organisational context. Activity that is routine for one role or privilege level may be unusual for another, while authorised workflows, working patterns, and access requirements vary between organisations.

Operational adaptation would therefore require behavioural representations to be interpreted alongside locally meaningful role, privilege, and policy information. This may involve mapping identity and access-management data, organisational units, privilege levels, authorised resources, and expected workflows into the broader behavioural modelling process. The semantic relationships investigated in this thesis should consequently be expected to require organisational recalibration rather than direct transfer from the CERT environment.

Operational governance and human oversight. External validity also extends beyond technical model performance. Insider-threat monitoring involves potentially sensitive organisational and employee telemetry, and model outputs may contribute to investigations with employment, disciplinary, security, or legal consequences. The architectures developed in this thesis should therefore be understood as mechanisms for producing behavioural risk evidence rather than autonomous mechanisms for determining malicious intent.

Operational use would require governance appropriate to the organisation and applicable legal and policy environment, including controlled access to telemetry and model outputs, auditability, privacy and data-retention controls, proportionality, appropriate explanation of alerts, and defined procedures for human review. These considerations are particularly relevant where behavioural models incorporate role, temporal, or organisational context whose interpretation may depend on circumstances not represented directly within the model.

Scope of generalisation. The external-validity claim of this thesis is therefore deliberately bounded. The numerical performance obtained on CERT is not claimed to generalise unchanged to operational organisations. Rather, the experiments provide evidence, within the evaluated benchmark and threat model, concerning four architectural hypotheses: the value of topology-informed representation, behavioural-

domain grounding, multi-horizon temporal modelling, and robustness under behaviourally constrained manipulation.

Whether these architectural effects persist, and with what magnitude, across operational environments remains an empirical question. Independent validation would need to consider differences in user populations, role and privilege structures, sensor coverage, feature availability, class prevalence, behavioural drift, organisational policy, and adversarial behaviour.

The framework should consequently be understood as an experimentally evaluated architectural foundation rather than a deployment-ready insider- threat detection system. Its architectural principles may provide a basis for transfer, but their implementations require organisation-specific adaptation, recalibration, and independent validation before operational deployment.

2.13.8 Section Summary

This section has argued that dataset selection is a foundational methodological decision in insider-threat detection research and has shown that dataset properties materially affect model development, evaluation, and interpretation. It has proposed explicit criteria for evaluating insider-threat datasets, including behavioural realism, temporal continuity, multi-log coverage, label characteristics, reproducibility, and ethical suitability [9], [10]. It has further provided a critical review of the CERT dataset, identifying both its strengths as a public multi-log benchmark and its limitations as a synthetic resource with known methodological pitfalls [13], [18].

Table 2.12 summarises public benchmark datasets, Table 2.13 summarises restricted real-world organisational datasets, and Table 2.14 summarises research-limited and study-specific datasets that are typically unsuitable for thesis-wide reproducibility.

Table 2.12: Insider-threat datasets: public benchmarks and suitability for this thesis

Dataset / category	Type	Temporal	Strengths	Limitations	Suitability
CERT insider-threat dataset [13], [14], [18], [30]	Synthetic / simulated	Yes	Multi-log behavioural data; widely used; reproducible; supports feature integration, modularity, hierarchical analysis, and controlled drift/adversarial experiments	Synthetic realism constraints; evaluation pitfalls if temporal discipline is not enforced; limited direct external validity	**High** (selected common benchmark across Chapters 3–6)
Other public benchmark / challenge corpora [9], [10]	Mixed (synthetic/-curated)	Varies	Enables reproducibility and independent benchmarking; useful for sensitivity checks and broader comparison	Often limited modality coverage or inconsistent schemas; may not support unified evaluation across all thesis stages	Medium (useful for supplementary benchmarking)

Table 2.13: Insider-threat datasets: real-world organisational datasets (restricted access)

Dataset / category	Type	Temporal	Strengths	Limitations	Suitability
Proprietary organisational datasets [8], [9], [12]	Real-world	Typically yes	Strong ecological validity; authentic organisational behaviour, constraints, and ambiguity	Restricted access; privacy/legal constraints; sparse/uncertain labels; weak reproducibility and cross-study comparability	Low–Medium (valuable for future external validation)

Table 2.14: Insider-threat datasets: research-limited logs and study-specific synthetic datasets

Dataset / category	Type	Temporal	Strengths	Limitations	Suitability
Institution-specific security logs (research-limited) [9], [10]	Real / semi-real	Sometimes	Targeted operational relevance within a specific environment; can reflect local policy and tooling	Narrow modality coverage; local schema dependence; often unavailable for replication; weak portability	Low (insufficient consistency/availability for thesis-wide progression)
Synthetic task-specific datasets (single-study generation) [10], [13]	Synthetic	Varies	Controlled scenario design; useful for hypothesis testing	Weak comparability across studies; high dependence on simulation assumptions; uncertain behavioural realism	Low (not suitable as constant benchmark)

As shown in Tables 2.12–2.14, the CERT dataset is not treated as a perfect proxy for real organisational insider behaviour, but as a defensible common benchmark for this thesis's controlled comparative architecture programme, given its multi-log structure, temporal analysability, public availability, and established use in prior literature.

The preceding ANN baseline section established the modelling foundation used throughout this thesis, while the present dataset section has clarified the data assumptions and validity boundaries under which that modelling is evaluated. Together, these sections define the technical and methodological basis for the comparative architectural contributions in Chapters 3–6. The next section therefore addresses the ethical and governance implications of insider-threat detection, including responsible interpretation, deployment boundaries, and safeguards against misuse.

2.14 Evaluation Discipline and Research Design

The effectiveness of insider-threat detection systems cannot be assessed solely through predictive performance metrics. Given the socio-technical complexity of insider behaviour, the temporal structure of organisational logs, and the adversarial dynamics introduced in Chapter 6, evaluation discipline becomes a foundational requirement rather than a peripheral methodological detail. This section articulates the research

design principles adopted throughout the thesis and examines threats to validity across internal, construct, external, and statistical dimensions.

2.14.1 Research Design Philosophy

The research design of this thesis follows a staged architectural progression aligned with design science principles, while remaining empirically grounded in reproducible machine-learning evaluation protocols. Each chapter introduces a structured architectural intervention (stabilisation, modularity, hierarchy, robustness), evaluates it under temporally faithful conditions, and analyses performance under both nominal and adversarial settings.

The design is cumulative rather than isolated: later architectures build upon earlier ones rather than replacing them. This layered structure enables controlled comparison and progressive mitigation of previously identified limitations. Importantly, no architectural component is evaluated under unrealistic cross-validation schemes or randomised splits that violate chronological causality.

2.14.2 Temporal Integrity and Leakage Prevention

A critical threat in insider-threat evaluation is temporal leakage. Random shuffling of behavioural logs across training and testing partitions can inadvertently expose future behavioural patterns to the model, artificially inflating predictive performance. Such practices are particularly problematic in insider detection, where behavioural progression and adaptation occur over time.

To mitigate this risk, all experimental evaluations in Chapters 3–6 employ blocked temporal partitioning. Training, validation, and testing datasets are strictly separated chronologically. Adversarial perturbations are applied only to test trajectories, ensuring that robustness assessment reflects deployment conditions rather than contamination of training data.

This temporal discipline strengthens internal validity by aligning evaluation with operational deployment scenarios.

2.14.3 Identity–Behaviour Confounding

Another methodological concern in insider-threat research is identity memorisation. When models learn to associate specific user identifiers with malicious labels, predictive performance may reflect identity recognition rather than behavioural inference. This risk is amplified in datasets where malicious users are few and repeatedly represented.

This thesis mitigates identity confounding through behavioural feature aggregation, domain decomposition, and stabilisation mechanisms that prioritise behavi-

oural patterns over raw identifiers. Additionally, evaluation protocols avoid leakage of user-specific future behaviour into training splits.

Nevertheless, identity confounding cannot be entirely eliminated in scenario-based datasets. This limitation is acknowledged as a constraint on external validity.

2.14.4 Metric Selection Under Class Imbalance

Insider-threat datasets are inherently imbalanced, with malicious events representing a small fraction of total behavioural activity. Under such conditions, accuracy becomes a misleading metric. A model predicting the majority class can achieve high accuracy while failing to detect meaningful insider activity.

Accordingly, this thesis prioritises precision, recall, F1-score, and area under the precision–recall curve (PR-AUC) as primary evaluation metrics. PR-AUC is particularly appropriate for skewed distributions because it emphasises positive-class performance and reflects risk prioritisation more faithfully than ROC-AUC in imbalanced contexts [19].

Threshold calibration is performed using validation data rather than test data, preventing post hoc optimisation and preserving statistical integrity.

2.14.5 Adversarial Robustness

Traditional insider-threat studies often report single-scenario performance without evaluating behaviour under adaptive manipulation. Chapter 6 extends evaluation to include behaviourally constrained adversarial perturbations.

Robustness is operationalised as performance degradation under plausibility-preserving behavioural operators rather than as norm-bounded feature perturbations. This design aligns adversarial evaluation with socio-technical realities while preserving interpretability.

By embedding adversarial testing within temporally faithful protocols, the thesis ensures that robustness claims are bounded and contextually defensible.

2.14.6 Construct Validity

Construct validity concerns whether the implemented models truly capture the behavioural constructs they claim to represent. The thesis mitigates construct ambiguity through:

- explicit mapping between behavioural domains and modular components (Chapter 4);
- hierarchical abstraction aligned with temporal behavioural progression (Chapter 5);
- invariant definitions grounded in organisational plausibility (Chapter 6).

These design choices reduce the risk that model outputs reflect arbitrary statistical correlations rather than meaningful behavioural inference.

2.14.7 Internal Validity

Internal validity is strengthened through controlled architectural comparisons. Each contribution is evaluated against structurally comparable baselines under identical data partitions and feature pipelines. Performance differences can therefore be attributed to architectural design rather than confounding variables.

Sensitivity analyses conducted across architectural components further mitigate risks of overfitting or accidental optimisation.

2.14.8 External Validity

External validity remains constrained by reliance on benchmark datasets. While such datasets support reproducibility and comparability, they cannot fully capture organisational diversity, evolving workflows, or adversarial creativity in real environments.

The thesis addresses this limitation by explicitly articulating dataset boundaries, modelling behavioural plausibility constraints, and avoiding claims of universal generalisation. Future deployment studies are recommended to validate transferability across organisational contexts.

2.14.9 Statistical Conclusion Validity

Statistical conclusion validity concerns whether reported performance differences are meaningful and not artefacts of random variation. The thesis mitigates this risk through:

- consistent use of validation sets for hyperparameter tuning;
- fixed random seeds for reproducibility;
- multiple metric reporting;
- degradation-based robustness interpretation rather than isolated adversarial accuracy.

These practices ensure that reported improvements reflect systematic architectural differences.

2.14.10 Reproducibility and Artefact Transparency

All architectural contributions are accompanied by algorithmic descriptions (Appendix C) and implementation artefacts. Code structure, parameter configurations, and evaluation pipelines are documented to support replication.

While dataset licensing constraints limit distribution of raw logs, the experimental design and architectural procedures are fully specified.

2.14.11 Summary of Validity Considerations

In summary, the research design of this thesis integrates temporal discipline, behavioural grounding, adversarial robustness, and governance-aware interpretation. While limitations remain—particularly regarding dataset realism and invariant specification—the methodological framework provides a defensible and transparent basis for evaluating insider-threat detection architectures.

By embedding evaluation discipline as a first-class design principle, the thesis strengthens the credibility of its technical contributions and situates them within a broader framework of responsible machine-learning research.

2.15 Ethics

The deployment of machine-learning systems for insider-threat detection raises substantial ethical, legal and socio-technical considerations. Unlike many security analytics applications, insider-threat detection directly involves the continuous monitoring, modelling and interpretation of human behaviour within organisational environments. As such, ethical considerations are not peripheral constraints but intrinsic design requirements that shape data handling practices, architectural decisions, evaluation methodologies and deployment governance. Recent policy frameworks and academic scholarship increasingly emphasise that high-impact AI systems operating in sensitive socio-technical domains must incorporate ethical safeguards as first-class design objectives rather than post hoc compliance measures [69], [70], [71].

This section reviews the ethical implications most relevant to insider-threat machine learning and situates the architectural contributions of this thesis within emerging organisational and regulatory expectations. The focus is not on abstract ethical theorisation, but on practical, operationally grounded concerns that arise when behavioural detection systems are deployed using real organisational data.

2.15.1 Privacy and Behavioural Monitoring

Insider-threat detection systems typically rely on fine-grained behavioural data, including authentication logs, file access records, communication metadata and system interaction traces. Although such data are often collected for operational or security

purposes, their use in behavioural modelling raises concerns related to privacy, proportionality and purpose limitation. Ethical AI frameworks consistently emphasise that data collection and analysis should be proportionate to the risks being mitigated and limited to what is strictly necessary for the stated security objective [70], [71].

From an ethical perspective, the primary concern is not merely data access, but behavioural inference. Machine-learning models may infer latent attributes such as stress, intent or risk propensity that were not explicitly provided by users, potentially exceeding the original scope of data collection. This thesis explicitly acknowledges this risk by favouring behaviourally abstracted representations over raw content inspection. The architectures proposed prioritise behavioural patterns, temporal structure and role-relative deviation rather than semantic inspection of private communications, thereby reducing unnecessary exposure of sensitive personal information.

2.15.2 Transparency and Human Oversight

Transparency and explainability are central ethical requirements for AI systems operating in high-stakes organisational contexts [72], [73]. Insider-threat detection systems that generate opaque risk scores or automated alerts without explanation risk eroding organisational trust and may lead to unjustified suspicion or harmful interventions.

The architectural choices explored in this thesis—particularly the use of Self-Organising Maps, modular decomposition and hierarchical behavioural abstraction—support interpretability by design. Behavioural clusters, domain-specific modules and hierarchical representations provide human analysts with structured insight into why a particular behaviour is flagged as anomalous. This aligns with ethical guidance emphasising that AI systems should support human decision-making rather than replace it [74]. Throughout this thesis, detection outputs are framed as decision-support signals that require contextual interpretation and human validation, rather than as automated determinations of malicious intent.

2.15.3 Fairness, Role Sensitivity and Bias Risk

Fairness in insider-threat detection cannot be understood solely through demographic parity or statistical group fairness, as the primary source of variation lies in organisational roles, privilege levels and workload expectations. Uniform anomaly thresholds risk disproportionately affecting high-activity users, privileged roles or employees operating under atypical but legitimate conditions. Behavioural cybersecurity research highlights that unfair or overly aggressive monitoring may itself generate organisational harm by discouraging legitimate work practices or increasing

psychological stress [12].

The behavioural grounding adopted in this thesis explicitly addresses this concern by modelling behaviour relative to role semantics and privilege context. By avoiding identity–behaviour confounding and emphasising role-aware baselining, the proposed architectures reduce the likelihood that users are flagged solely due to high activity volume or structural centrality. This role-sensitive approach aligns ethical fairness with technical robustness, demonstrating that fairness and detection accuracy are not competing objectives but mutually reinforcing design goals.

2.15.4 Psychological and Organisational Impact

Ethical evaluation of insider-threat detection systems must extend beyond technical performance to consider their psychological and organisational consequences. Persistent monitoring and risk scoring may alter user behaviour, increase stress, or erode trust between employees and management. Research in socio-technical systems and behavioural security indicates that poorly governed monitoring systems can produce chilling effects, where employees modify behaviour not to improve security but to avoid scrutiny [75].

This thesis recognises that insider-threat detection should not be equated with surveillance or automated judgement. Instead, detection systems should function as early-warning mechanisms that support risk triage, organisational intervention and contextual investigation. The adversarial modelling framework introduced in later chapters further reinforces this perspective by distinguishing between behavioural deviation and malicious intent, thereby reducing the risk of overreaction to benign behavioural change.

2.15.5 Governance and Deployment Boundaries

Emerging AI governance frameworks emphasise the need for clear accountability structures, auditability and defined deployment boundaries for high-risk AI systems [71], [76]. In the context of insider-threat detection, this implies that machine-learning systems should operate within well-defined organisational policies that specify data access rights, escalation procedures and human oversight responsibilities.

The architectures proposed in this thesis are explicitly designed to integrate with governance mechanisms rather than bypass them. Outputs are interpretable, modular and auditable, enabling organisations to align detection capabilities with internal policy, legal obligations and ethical standards. Importantly, the thesis does not advocate fully autonomous insider-threat enforcement; instead, it positions machine-learning models as components within a broader socio-technical governance framework.

2.15.6 Ethical Implications

The ethical considerations outlined above directly inform the architectural progression developed in subsequent chapters. Behavioural representation stability (Chapter 3) supports proportional monitoring by reducing sensitivity to benign variability. Modular domain decomposition (Chapter 4) enables selective analysis of behavioural domains without unnecessary data aggregation. Hierarchical behavioural modelling (Chapter 5) facilitates contextual interpretation across organisational levels. Finally, adversarial robustness (Chapter 6) mitigates the risk that detection systems are manipulated in ways that could unjustly implicate users.

Taken together, these contributions demonstrate that ethical considerations in insider-threat detection are not external constraints but fundamental drivers of robust, interpretable and responsible system design. By embedding ethical awareness within architectural choices, this thesis aligns technical innovation with organisational trust, legal accountability and human-centred security practice.

2.16 Chapter Summary

This chapter has critically reviewed the literature on insider-threat detection and established the conceptual, methodological, dataset, and ethical foundations for the thesis. The review has shown that insider-threat detection remains a complex socio-technical problem shaped by behavioural ambiguity, temporal dependency, organisational context, severe class imbalance, and adversarial adaptation. Across the literature, recurring weaknesses include fragmented problem formulations, limited cross-domain behavioural modelling, insufficient temporal evaluation discipline, and performance claims that are difficult to interpret under realistic operational constraints.

A key contribution of this chapter is the positioning of Artificial Neural Networks (ANNs) as a unifying methodological baseline. The chapter has argued that ANNs offer a flexible and powerful foundation for modelling non-linear behavioural patterns in insider-threat data, while also demonstrating that conventional ANN baselines remain insufficient when used in isolation. In particular, limitations relating to representation stability, domain heterogeneity, multi-level behavioural abstraction, temporal validity, and adversarial resilience motivate the structured architecture progression developed in later chapters. By introducing an ANN baseline synthesis and case-study framing at the literature-review stage, the thesis establishes a consistent comparative reference through which subsequent contributions can be interpreted coherently.

The chapter has also expanded the literature review to include a dedicated evaluation of datasets for insider-threat detection, with explicit criteria for assessing behavioural realism, temporal continuity, multi-log coverage, label suitability, re-

producibility, and ethical appropriateness. On this basis, the CERT dataset has been critically reviewed and justified as the common benchmark used throughout the experimental chapters. Importantly, CERT is not treated as a perfect proxy for real organisational behaviour; rather, it is adopted as a defensible and reproducible benchmark for controlled comparative architectural research. The chapter has further clarified the validity boundaries associated with CERT-based experimentation and has outlined mitigation strategies used throughout the thesis, including chronology-preserving evaluation, leakage-aware preprocessing, and restrained interpretation of transferability.

Taken together, the literature synthesis, ANN baseline framing, and dataset rationale establish the technical and methodological basis for the thesis contributions. Hybrid SOM–ANN modelling (Chapter 3) addresses behavioural variance stabilisation and representational structure; modular modelling through MANN (Chapter 4) addresses domain heterogeneity and cross-source decomposition; hierarchical behavioural modelling (Chapter 5) addresses multi-level abstraction and long-horizon behavioural progression; and the R-Guard framework (Chapter 6) addresses adversarial robustness under behaviourally constrained perturbations. This progression is therefore framed as a cumulative set of ANN-centred refinements designed to address increasingly challenging and operationally relevant insider-threat detection requirements.

Finally, the chapter has situated insider-threat detection within its ethical and governance context. It has highlighted that behavioural detection systems can produce significant organisational and human consequences if they are deployed without proportionality, transparency, interpretability, and appropriate safeguards. The ethical and governance discussion in this chapter therefore complements the technical foundations by clarifying responsible interpretation, deployment boundaries, and safeguards against misuse.

The next chapter introduces the first technical contribution of the thesis: a hybrid insider-threat detection architecture integrating Self-Organising Maps with neural classification. This framework directly addresses the representational and stability limitations identified in the literature and establishes the first stage of the thesis's structured progression toward modular, hierarchical, and adversarially robust insider-threat detection.

Chapter 3

Hybrid Framework

3.1 Overview

This chapter introduces the first architectural contribution of the thesis: the Self-Organising Map and Artificial Neural Network (SOM–ANN) hybrid framework, referred to as SOMANN. The motivation for SOMANN arises from a representational limitation identified in Chapter 2: much of the insider-threat detection literature operates directly on *flat*, noise-dominated feature vectors derived from raw organisational logs. In such representations, benign behavioural variability, role-dependent routines, and user-specific work patterns distort feature-space geometry, increasing the risk of identity memorisation, inflated false positives, and unstable inference under temporally faithful evaluation.

To address this limitation, SOMANN adopts a Perception–Inference framing of insider-threat detection. In this framing, *perception* refers to the unsupervised learning of a stable, topology-preserving behavioural organisation from high-dimensional multi-log telemetry, while *inference* refers to supervised risk estimation performed on topology-informed behavioural descriptors rather than on raw feature vectors. This separation is intentional: by stabilising behavioural representation prior to discrimination, SOMANN seeks to reduce sensitivity to benign volatility while preserving discriminative structure relevant to insider risk.

The chapter develops SOMANN as a stability-oriented foundation rather than a complete solution to insider-threat detection. The intent is not to maximise predictive performance in isolation, but to establish a representational layer that improves robustness, interpretability, and suitability for subsequent architectural extensions.This Perception–Inference separation establishes a representational foundation that is later extended through domain-aware modular decomposition (Chapter 4), temporal abstraction via hierarchical modelling (Chapter 5), and behaviourally grounded robustness mechanisms under adaptive adversarial conditions (Chapter 6).

3.2 Introduction

As established in Chapter 2, insider-threat detection presents a fundamentally different challenge from external intrusion detection. Insider activity unfolds within legitimate trust boundaries, is shaped by role semantics and organisational context, and often manifests as gradual behavioural drift rather than abrupt anomalies. While machine learning (ML) has become the dominant paradigm for analysing insider telemetry, prevailing approaches frequently rely on flattened behavioural representations and monolithic classifiers that are sensitive to noise, identity confounding, and temporally invalid evaluation practices [8], [9], [10], [13]. As a result, strong benchmark performance does not necessarily translate into behaviourally meaningful or robust detection.

Identity confounding as a representational failure. A recurring pitfall in insider-threat detection is *identity confounding*: when models trained on high-dimensional behavioural logs implicitly learn *who* the user is (stable role- and individual-specific routines) rather than *what* the behaviour indicates (deviation, progression, or intent-relevant change). In enterprise telemetry, many features are strongly user- and role-correlated (e.g., working hours, typical devices, access scopes, and file-path distributions). Monolithic supervised models can therefore achieve deceptively strong performance by exploiting identity signatures, particularly when malicious labels are concentrated among a small subset of users or when evaluation splits inadvertently expose future user-specific patterns. This reduces generalisation to unseen users and inflates confidence under deployment shift. SOM-based perception mitigates this risk by learning a topology of *behavioural similarity* without using labels, thereby compressing high-variance inputs into stable, neighbourhood-structured representations that the downstream ANN can exploit for inference with reduced reliance on raw identity cues [40], [41].

This chapter introduces the first technical contribution of the thesis: **SOM–ANN (SOMANN)**, a hybrid insider-threat detection framework that explicitly separates behavioural *perception* from behavioural *inference*. The core premise is that unsupervised topology-preserving organisation can stabilise behavioural representations prior to supervised discrimination. In SOMANN, a Self-Organising Map (SOM) learns a structured behavioural space from high-dimensional activity data, while an Artificial Neural Network (ANN) performs supervised risk inference using topology-informed descriptors derived from the SOM rather than raw behavioural vectors.

Figure 3.1 summarises the SOMANN pipeline as an explicit Perception–Inference separation: the SOM stage performs *label-agnostic* behavioural organisation (perception) by learning a topology-preserving map of similarity, while the ANN stage uses supervised learning (inference) on the resulting topology-informed descriptors

to estimate insider-risk.

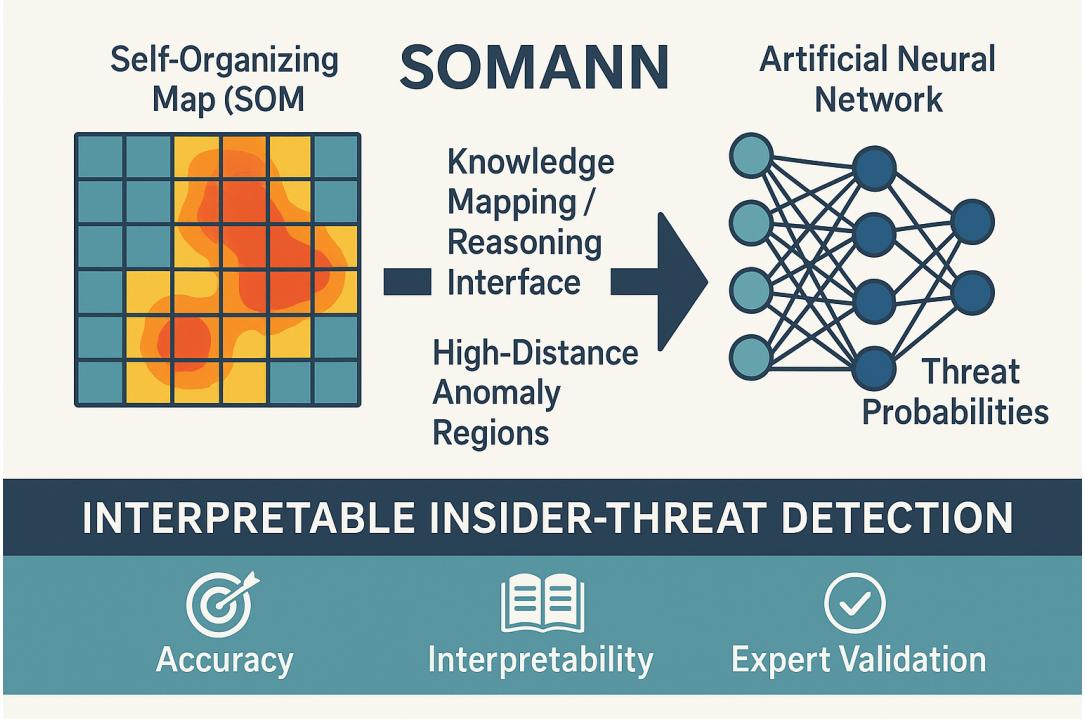


Figure 3.1: Conceptual SOMANN hybrid detection pipeline. The SOM performs unsupervised topology-preserving behavioural organisation to form a structured knowledge space. A knowledge-mapping interface derives topology-informed descriptors that are then consumed by an ANN for supervised risk inference and calibrated threat probability estimation.

SOMs are well established as tools for discovering latent structure in high-dimensional data while preserving neighbourhood relationships [37]. In isolation, however, they provide limited support for intent discrimination or operational decision-making. Conversely, ANNs offer strong discriminative performance but are prone to overfitting noisy behavioural fluctuations when trained directly on raw log-derived features, particularly under class imbalance and weak temporal evaluation [10], [13]. SOMANN reconciles these complementary strengths by using the SOM as a stability-oriented representational layer and the ANN as a decision layer operating on structured behavioural descriptors. The formal properties of SOM learning and the construction of SOM-derived representations are developed in detail in Section 3.3.

Importantly, SOMANN is not proposed as a replacement for supervised learning, nor as a purely anomaly-driven detector. Rather, it provides a principled hybridisation in which unsupervised behavioural organisation supports more stable, interpretable, and temporally faithful supervised inference. This design choice reflects the thesis's broader commitment to disciplined evaluation and behaviourally grounded modelling, rather than incremental performance optimisation.

Within the overall thesis architecture, this chapter establishes *representational stability* as the first step in a staged progression of increasingly structured insider-

threat detection models. Chapter 4 extends this principle through modular decomposition across behavioural domains, Chapter 5 introduces hierarchical abstraction over behavioural trajectories, and Chapter 6 addresses robustness under behaviourally constrained adversarial manipulation.

The remainder of the chapter is organised as follows. Section 3.2 further motivates the SOMANN framework and situates the Perception–Inference concept within the thesis progression. Section 3.3 formalises the SOM learning process and defines the derived behavioural representations used for classification. Section 3.4 describes preprocessing, chronology-preserving evaluation, and baseline comparators. Section 3.5 reports experimental results against an ANN-only baseline, with additional diagnostics provided in Appendix A. Section 3.6 interprets findings, clarifies limitations, and positions SOMANN as a representational precursor to modular decomposition in Chapter 4. Section 3.7 concludes the chapter.

3.3 Formulation of the SOMANN

This section formally defines the SOMANN framework, beginning with the mathematical formulation of the Self-Organising Map (SOM), followed by the discriminative Artificial Neural Network (ANN) layer, and concluding with the integrated hybrid pipeline. The aim is to establish a rigorous theoretical foundation that links unsupervised topology-preserving representation learning with supervised behavioural discrimination under temporally progressive insider activity.

The formulation builds upon classical SOM theory [37] and insider-threat deep learning studies that highlight the challenges of behavioural non-stationarity, heterogeneity, and evaluation fragility [10], [13], [14], [30]. The key extension in SOMANN is not merely combining an unsupervised and supervised model, but explicitly *using the SOM as a stability-oriented behavioural structuring layer* whose topology-derived descriptors provide the ANN with inputs that are less sensitive to noisy intra-user variance and more aligned with behavioural similarity.

3.3.1 Definition of the Self-Organising Map

Let $\mathbf{x} \in \mathbb{R}^d$ denote a behavioural feature vector representing user activity over a defined temporal window (e.g., user–day or user–session). A SOM is a discrete set of M neurons arranged on a low-dimensional lattice (typically a 2D grid). Each neuron $i \in \{1, \ldots, M\}$ is associated with a prototype (weight) vector $\mathbf{w}_i \in \mathbb{R}^d$ and a lattice coordinate $\mathbf{r}_i \in \mathbb{R}^2$.

Best Matching Unit (BMU). For an input $\mathbf{x}$, the BMU index c is defined as the prototype minimising the distance to $\mathbf{x}$:

$$c = \arg\min_i \|\mathbf{x} - \mathbf{w}_i\|. \tag{3.1}$$

Neighbourhood-based weight updates. Given BMU c, the SOM updates prototypes using:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \alpha(t)\, h_{c,i}(t)\, (\mathbf{x} - \mathbf{w}_i(t)), \tag{3.2}$$

where $\alpha(t)$ is the learning rate and $h_{c,i}(t)$ is the neighbourhood function controlling how far updates propagate on the lattice.

Neighbourhood function. A common choice is Gaussian neighbourhood weighting:

$$h_{c,i}(t) = \exp\left(-\frac{\|\mathbf{r}_c - \mathbf{r}_i\|^2}{2\sigma(t)^2}\right), \tag{3.3}$$

where $\sigma(t)$ controls neighbourhood width. In standard SOM training, $\alpha(t)$ and $\sigma(t)$ decay over time, producing global ordering early and local refinement later [37].

Interpretation in behavioural telemetry. In insider-threat settings, d is typically large because behavioural vectors combine multi-log features (authentication, file, device, web, communication) plus contextual metadata. The SOM therefore acts as a *prototype organiser* that compresses high-dimensional behavioural patterns into a topology-preserving lattice. This is valuable because insider telemetry often contains high benign variability; SOM training aggregates local neighbourhood structure without requiring labels, enabling representation learning before any discriminative objective is applied.

3.3.2 Behavioural Interpretation

The relevance of SOMs to insider-threat detection emerges through behavioural interpretation. Organisational user behaviour occupies constrained regions in feature space shaped by role semantics, privilege boundaries, workload rhythms, and policy constraints [8], [11]. In this view, the observed high-dimensional activity vectors can be regarded as samples from a behavioural manifold rather than arbitrary points in $\mathbb{R}^d$.

SOM training can be interpreted as learning a discretised approximation of this manifold:

- **Neighbourhood coherence:** adjacent neurons represent behaviourally similar activity profiles (e.g., similar working-hour patterns, access rhythms, or file/device usage intensity).

- **Density structure:** dense lattice regions correspond to frequently occurring behavioural regimes (routine work patterns), whereas sparse regions correspond to atypical or transitional behaviours.
- **Trajectory plausibility:** gradual behavioural change is expected to correspond to smooth movement across neighbouring regions rather than abrupt jumps, consistent with insider escalation as a trajectory rather than a single spike [22].

This interpretation is not assumed a priori; rather, it motivates diagnostic evaluation later in the chapter (e.g., activation heatmaps and trajectory visualisation) to examine whether SOM organisation aligns with plausible behavioural structure.

3.3.3 SOM-Derived Representation

A crucial design choice in SOMANN is the construction of the SOM-derived representation $\mathbf{z}$ used for supervised learning. Rather than feeding only raw BMU indices (which can be brittle and non-metric), SOMANN uses topology-informed descriptors that preserve behavioural similarity and quantify "distance-to-prototype" effects.

Let $c(\mathbf{x})$ denote the BMU index for $\mathbf{x}$. The SOMANN representation can be defined as:

$$\mathbf{z}(\mathbf{x}) = [\mathbf{x} \parallel \phi_{\text{SOM}}(\mathbf{x})], \tag{3.4}$$

where $\phi_{\text{SOM}}(\mathbf{x})$ is a vector of SOM descriptors. In practice, $\phi_{\text{SOM}}(\mathbf{x})$ may include one or more of the following components:

- **BMU coordinates:** $(r_{c,1}, r_{c,2})$ to capture coarse topological location.
- **Quantisation error (instance-level):** $\text{QE}(\mathbf{x}) = \|\mathbf{x} - \mathbf{w}_{c(\mathbf{x})}\|$, measuring deviation from the closest prototype.
- **Second-BMU gap:** $\Delta(\mathbf{x}) = \|\mathbf{x}-\mathbf{w}_{c_2}\|-\|\mathbf{x}-\mathbf{w}_{c_1}\|$, which can reflect boundary behaviour and ambiguity.
- **Neighbour activation profile (optional):** distances to a small neighbourhood of prototypes, producing a local "activation vector".

This representation strategy has two benefits in insider-threat detection. First, the quantisation and neighbourhood features encode *behavioural deviation relative to learned prototypes* without requiring labels, which is aligned with risk being subtle and plausibility-constrained. Second, the representation reduces sensitivity to raw-feature noise by anchoring discrimination to stable prototypes, addressing identity–behaviour confounding in high-variance behavioural data [10], [13].

3.3.4 ANN Discriminative Layer

The ANN component operates on SOM-derived representations to perform supervised classification. Let $\mathbf{z} \in \mathbb{R}^k$ be the SOMANN input representation. A standard feedforward ANN computes:

$$\hat{y} = \sigma(\mathbf{W}_2\, \phi(\mathbf{W}_1 \mathbf{z} + \mathbf{b}_1) + \mathbf{b}_2)\,, \tag{3.5}$$

where $\phi(\cdot)$ is a non-linear activation (e.g., ReLU), $\sigma(\cdot)$ is the sigmoid function, and $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1, \mathbf{b}_2$ are trainable parameters. For insider detection tasks, $\hat{y}$ can be interpreted as a calibrated risk score rather than a definitive "guilt" signal, which supports analyst-centred usage under governance constraints.

Loss function and regularisation. Binary cross-entropy is used:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]\,, \tag{3.6}$$

with regularisation (dropout and/or L_2 weight decay) to reduce overfitting. This is especially important under severe class imbalance and temporally progressive behaviour, where models can otherwise learn spurious correlates [10], [13].

3.3.5 Integrated SOMANN Pipeline

SOMANN integrates the above components using a two-stage learning pipeline:

1. **Unsupervised SOM training:** learn stable prototypes $\{\mathbf{w}_i\}$ on the training set to approximate the behavioural manifold and preserve neighbourhood relations.

2. **Supervised ANN training:** compute $\mathbf{z}(\mathbf{x})$ for each instance and train the ANN classifier on the SOM-structured representations.

This separation is deliberate: it encourages representational stability prior to discriminative optimisation, reducing the likelihood that the classifier overfits to transient noise or leakage-induced artefacts. It also supports clearer interpretability: deviations (e.g., high quantisation error) can be reasoned about in relation to learned behavioural prototypes, rather than as opaque feature interactions.

Algorithm 1: SOMANN Training Procedure (Perception–Inference two-stage learning)

Input: Training features X_{train}; labels y_{train}; SOM parameters Φ; ANN parameters Θ

Output: Trained SOM SOM_Φ; trained ANN f_Θ

Phase I: Perceptual training (SOM; label-agnostic)

Note: labels y_{train} are not used in this phase.

Initialise SOM prototypes $\{\mathbf{w}_i\}$ on a 2D lattice

for $e \leftarrow 1$ **to** E **do**

  foreach *training sample* $\mathbf{x}$ *in* X_{train} **do**

    Compute BMU c using Eq. 3.1

    Update prototypes using Eq. 3.2 with neighbourhood Eq. 3.3

Phase II: Inference training (ANN; supervised)

Compute SOM descriptors $\phi_{\text{SOM}}(\mathbf{x})$ and build $\mathbf{z}(\mathbf{x})$ using Eq. 3.4

Train ANN f_Θ on $\{\mathbf{z}(\mathbf{x}), y_{\text{train}}\}$ using loss Eq. 3.6 with early stopping

3.3.6 Theoretical Justification of SOMANN

The SOMANN design is theoretically motivated by three principles that are directly aligned with insider-threat requirements.

(1) Variance stabilisation under high benign variability. Behavioural telemetry exhibits high intra-user variance and strong context dependence. SOM learning reduces effective variance by anchoring observations to prototypes and neighbourhood structure, thereby simplifying downstream discrimination.

(2) Representation smoothing and drift tolerance. Neighbourhood-based updates impose smoothness in the learned map: similar behaviours map to nearby neurons, and small feature changes tend to produce local representation changes. This property supports drift-tolerant modelling, which is critical in organisational contexts characterised by gradual behaviour change [32], [33].

(3) Separation of representation and decision learning. Decoupling unsupervised structure discovery from supervised discrimination reduces the risk that the classifier memorises identity-specific noise and improves interpretability relative to monolithic end-to-end classifiers [13], [14]. In SOMANN, the SOM provides a behaviourally structured "perception" layer, while the ANN provides a discriminative "inference" layer whose decisions can be analysed in relation to prototype proximity and neighbourhood transitions.

3.3.7 Plasticity, Stability, and Convergence

A defining property of the SOM is its ability to balance *plasticity* (adaptation to new patterns) and *stability* (preservation of learned structure). This balance is particularly important in insider-threat detection, where behaviour evolves due to role transitions, workload changes, and organisational restructuring [8], [11]. Plasticity arises from the update mechanism in Eq. 3.2; early training uses broader neighbourhoods to enforce global ordering, while later training shrinks neighbourhood influence to refine local structure without catastrophic overwriting [37].

Under standard assumptions (bounded input space and decreasing learning rate/neighbourhood schedules), SOM learning satisfies convergence-like conditions such as:

$$\sum_{t=1}^{\infty} \alpha(t) = \infty, \quad \sum_{t=1}^{\infty} \alpha(t)^2 < \infty, \tag{3.7}$$

ensuring diminishing but persistent adaptation. While strict convergence proofs are non-trivial in non-stationary high-dimensional behavioural streams, empirical work suggests robust asymptotic stability under moderate drift regimes [77], [78]. In SOMANN, the practical implication is that the ANN is trained on representations that reflect stable behavioural structure rather than transient sampling artefacts, improving generalisation under temporally faithful evaluation.

3.3.8 Topology Preservation Metrics

Topology preservation is a central advantage of SOM-based representation learning. Inputs that are behaviourally similar are mapped to adjacent neurons, enabling neighbourhood reasoning about behavioural similarity. Two commonly used metrics are the quantisation error (QE) and the topographic error (TE):

$$\mathrm{QE} = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{w}_{c(i)}\|, \tag{3.8}$$

$$\mathrm{TE} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left[\mathrm{BMU}_1(\mathbf{x}_i) \not\sim \mathrm{BMU}_2(\mathbf{x}_i)\right], \tag{3.9}$$

where $\not\sim$ indicates non-adjacency on the lattice. Low TE indicates faithful preservation of neighbourhood relations, which is aligned with insider modelling: small behavioural deviations should not trigger disproportionate representation shifts, while sustained deviation may manifest as movement toward sparse or boundary regions.

3.3.9 Variance Smoothing

From a statistical perspective, SOM learning performs a form of manifold compression: high-dimensional behavioural observations are mapped to a structured lattice that preserves neighbourhood relations while reducing representational complexity. Neighbourhood-based updates induce local averaging effects that suppress noise and reduce intra-class variance while retaining inter-class separability.

Let $\mathcal{B}$ denote a behavioural regime with variance $\sigma^2_{\mathcal{B}}$ in the original feature space. After SOM organisation, the effective within-neighbourhood variance satisfies:

$$\sigma^2_{\text{SOM}} < \sigma^2_{\mathcal{B}}, \tag{3.10}$$

under neighbourhood overlap and sufficient sampling. This stabilises subsequent ANN training, mitigating dominance of high-volume users and reducing sensitivity to benign behavioural fluctuation—both known failure modes in insider benchmark settings [13].

Stability as neighbourhood-based low-pass filtering (probabilistic intuition). The inequality $\sigma^2_{\text{SOM}} < \sigma^2_{\mathcal{B}}$ is not presented as a strict universal guarantee, since the realised variance depends on data distribution, hyperparameter choices, and the stationarity assumptions implicit in aggregation. Rather, the claim is supported by a probabilistic intuition grounded in the observation that SOM-based representation induces *neighbourhood averaging* over behaviourally similar samples. During training, updates are applied not only to the best matching unit (BMU) but also to its topological neighbours through a decaying neighbourhood kernel. This update mechanism behaves analogously to a *low-pass filter* on behavioural noise: high-frequency, idiosyncratic fluctuations that are not consistently reinforced across similar observations tend to be attenuated, while stable behavioural structure shared by multiple nearby samples is preserved.

More concretely, let $\mathbf{x}$ denote a behavioural feature vector and let $\phi(\mathbf{x})$ denote the SOM-induced representation (e.g., BMU index, activation pattern, or codebook-based descriptor). If $\phi(\cdot)$ is constructed from a kernel-weighted combination of nearby codebook vectors, it can be interpreted as a local smoothing operator over the input space. Under mild assumptions that noise components are approximately zero-mean and weakly correlated across neighbourhood-consistent samples, such kernel smoothing reduces variance by averaging out unstructured fluctuation, yielding $\text{Var}[\phi(\mathbf{x})]$ that is typically lower than $\text{Var}[\mathbf{x}]$ along noise-dominated directions. In insider telemetry, where benign variability arising from workload rhythms, role heterogeneity, and routine operational behaviour often dominates raw feature variance, this stabilisation effect is particularly valuable: it reduces sensitivity to transient deviations that lack behavioural coherence while retaining topology that supports

downstream discrimination.

This stabilising behaviour should therefore be interpreted as an *architectural bias* toward smooth, topology-consistent behavioural representation rather than as a proof of strict variance dominance in all settings. The SOM grid size, neighbourhood schedule, and learning-rate dynamics directly influence the degree of smoothing, and their impact is empirically examined through the ablation and sensitivity analyses reported later in this chapter under chronology-preserving evaluation.

3.3.10 Theoretical Illustrations



Figure 3.2: Conceptual illustration of behavioural manifold compression in SOMANN. High-dimensional behavioural observations (constructed from multi-log features) are organised onto a two-dimensional SOM lattice that preserves neighbourhood relations. This topology-preserving compression stabilises representation geometry prior to supervised discrimination, supporting trajectory-aware interpretation and reducing sensitivity to noise-dominated variance.

The theoretical properties outlined above motivate the experimental design and evaluation choices adopted in this chapter. In particular, the separation of representation learning and discrimination informs the two-stage training procedure, while the emphasis on temporal plausibility and behavioural stability guides the evaluation protocol described next.

3.4 Methodology

This section describes the experimental methodology used to evaluate the SOMANN framework introduced in Section 3.3. The design choices are guided by three principles established earlier in the thesis: (i) temporally faithful evaluation, (ii) separation of representation learning and discriminative inference, and (iii) explicit mitigation of dataset artefacts and leakage risks. Rather than reintroducing dataset selection criteria or evaluation philosophy, this section operationalises the disciplined framework developed in Chapter 2 within the context of the SOMANN hybrid architecture.

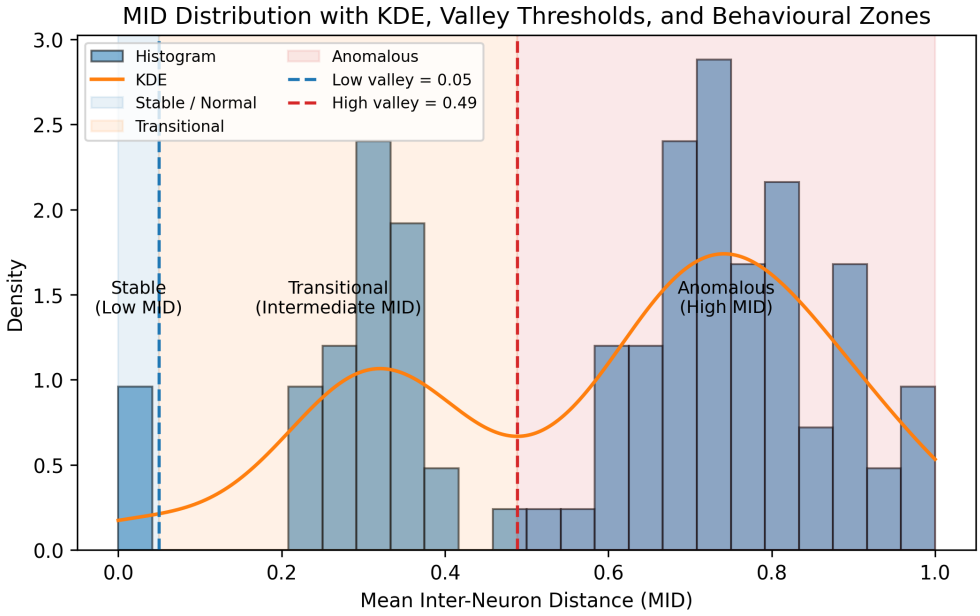


Figure 3.3: Variance smoothing and behavioural regime structure induced by SOM organisation. The distribution of SOM-derived deviation measures (e.g., MID/quantisation-based distance) reveals clustered low-distance regions corresponding to routine behavioural regimes and higher-distance regions associated with atypical or transitional activity. These topology-informed descriptors form stability-oriented inputs for ANN risk inference in SOMANN.

3.4.1 Experimental Setup and Data Partitioning

Experiments are conducted on the CERT insider-threat benchmark, selected and critically justified in Chapter 2. Consistent with the emphasis on chronological validity, data are partitioned using temporally ordered splits rather than random sampling. Earlier time segments are used exclusively for model training and validation, while later segments are reserved for testing. This design prevents information leakage across temporal boundaries and reflects realistic deployment conditions in which models must generalise to future behavioural activity.

All preprocessing operations—including feature scaling, SOM training, and ANN parameter estimation—are performed strictly within the training partition. No statistics derived from the test set are incorporated during model fitting. This separation is essential because behavioural telemetry exhibits strong user-specific and temporal dependencies; careless partitioning can lead to inflated performance estimates [10], [13].

3.4.2 Data Preprocessing and Feature Construction

Behavioural features are constructed from multi-log organisational activity (e.g., authentication events, file access, device usage). Feature engineering follows a structured aggregation process over defined temporal windows (e.g., user–day granularity), producing a fixed-length behavioural vector $\mathbf{x} \in \mathbb{R}^d$ for each instance.

Preprocessing includes:

- **Aggregation:** event-level logs are aggregated into behaviourally meaningful metrics (counts, frequencies, ratios).
- **Normalisation:** continuous features are scaled using statistics computed solely on the training set to ensure comparability across dimensions.
- **Missing-value handling:** absent activity is encoded explicitly (e.g., zeros or domain-consistent defaults), avoiding imputation strategies that could distort behavioural sparsity.

These steps serve not only practical consistency but also theoretical alignment with the SOM's topology-preserving objective. Because the SOM relies on distance-based similarity (Section 3.3.1), feature scaling and stable aggregation are necessary to prevent any single behavioural dimension from dominating the neighbourhood structure.

3.4.3 SOM Training Procedure

Consistent with the representation-stabilisation objective outlined in Section 3.3, the SOM is trained in an unsupervised manner on the training partition prior to any supervised optimisation. Let $\{\mathbf{x}_i\}_{i=1}^{N}$ denote the training behavioural vectors.

The SOM lattice is defined as a two-dimensional grid of fixed size (selected to balance representational resolution and neighbourhood coherence). Grid size is chosen empirically to provide sufficient granularity for behavioural differentiation while maintaining stable prototype neighbourhoods. This empirical selection is intended to avoid *topological collapse* when the map is too small (distinct behavioural regimes are forced into the same neighbourhood, reducing separability) and to avoid *over-fragmentation* when the map is too large (the representation becomes overly granular, producing sparsely occupied units and weakening the stabilising effect of neighbourhood smoothing). In practice, the selected grid size was verified to yield coherent topology formation without excessive sparsity, supported by SOM diagnostics and stability checks reported below (parenthetically, this verification included monitoring standard SOM quality indicators such as quantisation error and topographic error to ensure meaningful topology preservation).

Training proceeds using the standard BMU identification and neighbourhood update rules described in Equations 3.1–3.3. Learning rate and neighbourhood radius decay schedules are applied to achieve global ordering in early epochs and local refinement in later epochs. This schedule reflects the plasticity–stability balance discussed in Section 3.3.7.

To assess structural stability, SOM diagnostics (e.g., MID distributions, U-matrix visualisations) are examined across independent initialisations. Supplementary diagnostic figures are provided in Appendix A to demonstrate consistency of behavioural topology formation.

3.4.4 Construction of SOM-Derived Representations

After SOM convergence on the training set, topology-informed descriptors are computed for each instance $\mathbf{x}$ to construct the hybrid input representation $\mathbf{z}(\mathbf{x})$ defined in Equation 3.4. These descriptors may include:

- BMU lattice coordinates,
- quantisation error (distance to prototype),
- neighbourhood-relative distance measures,
- region-based indicators (e.g., high-distance zones).

This stage operationalises the theoretical separation between behavioural perception (unsupervised structuring) and behavioural inference (supervised classification). Importantly, the ANN is not trained directly on raw behavioural vectors $\mathbf{x}$ in the SOMANN configuration; instead, it operates on the SOM-structured representation $\mathbf{z}$, thereby inheriting the stabilised geometry of the learned behavioural manifold.

3.4.5 ANN Training and Integration

The ANN component is trained on the SOM-derived representations $\mathbf{z}(\mathbf{x})$ using labelled training data. Binary cross-entropy loss is minimised via stochastic optimisation with early stopping based on validation performance.

Class imbalance is addressed through appropriate loss weighting and threshold calibration. Rather than optimising accuracy, which can be misleading in rare-event detection, model performance is evaluated using metrics sensitive to minority-class detection (e.g., precision, recall, PR-AUC), consistent with the evaluation rationale established in Chapter 2 [19].

In line with governance considerations discussed earlier in the thesis, ANN outputs are interpreted as calibrated risk scores rather than automated decision outcomes. Thresholds are applied only for evaluation purposes; in operational settings, scores may support analyst triage rather than binary enforcement.

3.4.6 Baseline Comparison and Evaluation Protocol

To assess the contribution of the SOM structuring layer, SOMANN is compared against the common ANN baseline, denoted ANN_{std}, trained directly on the corresponding non-SOM behavioural feature representation. Within the Chapter 3

comparison, both architectures are evaluated using matched temporal partitions, preprocessing controls, label definitions, and evaluation procedures. The specification and broader thesis-wide role of this baseline are clarified below.

3.4.6.1 Unified Baseline and Cross-Chapter Protocol

To ensure methodological consistency across the technical contributions in Chapters 3–6, this thesis adopts a common reference model, denoted ANN_{std}. The purpose of this baseline is to provide a stable methodological reference against which the additional architectural mechanisms introduced by SOMANN, MANN, MLML, and $\mathcal{R}$-Guard can be evaluated.

The term *unified baseline* refers to a common model specification and comparative protocol. It does not imply that a single fitted ANN instance is reused across all experiments, nor that the baseline must obtain an identical numerical result under every evaluation condition. Where later chapters deliberately alter the data distribution or threat condition, the baseline is retrained and evaluated within that experimental regime while retaining the common reference specification. Differences in resulting performance are therefore interpreted in relation to the evaluation condition rather than as evidence of different baseline architectures.

Baseline specification. The common baseline is a feed-forward Artificial Neural Network, ANN_{std}, operating directly on the corresponding behavioural feature representation before application of the architectural mechanism under investigation. It therefore provides the non-SOM, non-modular, non-hierarchical, and non-defended reference condition against which the staged contributions are evaluated where a direct comparison is methodologically appropriate.

The common specification comprises:

- **Model family:** a fully connected multi-layer perceptron classifier;
- **Input condition:** the corresponding behavioural feature representation prior to the architectural transformation being evaluated in the relevant experiment;
- **Output:** a binary insider-threat probability;
- **Loss function:** binary cross-entropy; and
- **Evaluation measures:** PR-AUC, ROC-AUC, precision, recall, F1-score, and false-positive rate, as applicable to the experimental condition.

The designation ANN_{std} therefore identifies the common baseline model specification and training discipline rather than an assertion that every technical chapter uses an identical input tensor or a single trained model instance. This distinction is necessary because MANN, MLML, and $\mathcal{R}$-Guard deliberately investigate different representational, temporal, and adversarial conditions.

Common evaluation controls. Within each direct architectural comparison, the baseline and proposed architecture are evaluated under matched experimental controls. These include:

1. matched training, validation, and test partitions for the comparison being reported;
2. identical outcome definitions and class composition within the corresponding experiment;
3. preprocessing parameters estimated exclusively from the training partition and subsequently applied unchanged to validation and test data;
4. matched feature-generation windows where temporal representation is not itself the experimental variable;
5. equivalent model-selection and stopping procedures; and
6. common evaluation metrics and decision-threshold procedures for the comparison being reported.

Chronology is preserved where temporal evaluation is required so that future behavioural observations do not enter model development through unrestricted random partitioning. Blocked temporal evaluation therefore assesses generalisation to later behavioural activity rather than allowing the comparison to benefit from direct temporal mixing.

These controls apply *within* the corresponding comparative experiment. They do not require every experiment in Chapters 3–6 to use an identical evaluation distribution. Temporal drift and adversarial perturbation, for example, are deliberate changes to the evaluation regime introduced to test specific failure modes.

Reconciliation of baseline values. The original thesis contained an inconsistency in the clean-condition ANN baseline reported across the technical chapters. Chapter 3 reported a PR-AUC of 0.72, whereas the corresponding baseline used in Chapter 4 was reported as 0.81. Review of the experimental configurations established that these values originated from runs that did not use the same finalised preprocessing and validation configuration and should therefore not have been presented as equivalent instances of a common thesis-wide baseline.

To resolve this inconsistency, the ANN reference condition was re-evaluated using the standardised preprocessing and chronology-preserving evaluation protocol adopted for the revised comparative analysis. Under this configuration, the clean-condition common baseline is:

$$\text{PR-AUC}\,(\text{ANN}_{\text{std}}) = 0.810. \tag{3.11}$$

The earlier value of 0.72 is therefore not retained as the thesis-wide clean-condition baseline because it was obtained under a different preprocessing and validation configuration. The value of 0.810 is not an average or manual reconciliation of the previously reported results; it is the result obtained from the standardised evaluation configuration used for the revised cross-chapter comparison.

Performance across evaluation conditions. A common baseline specification does not imply constant performance when the evaluation condition changes. Chapters 5 and 6 deliberately introduce more demanding conditions in order to examine temporal degradation and adversarial manipulation respectively. The performance of ANN_{std} under these conditions is therefore condition-specific.

Table 3.1 summarises the corresponding baseline results.

Table 3.1: Performance of the common ANN_{std} baseline under the principal evaluation conditions used across the thesis.

Evaluation Condition	ROC-AUC	PR-AUC	F1-score	FPR
Clean, non-adversarial evaluation (Chapters 3–4)	0.842	0.810	0.715	0.084
Temporal-drift evaluation (Chapter 5)	0.791	0.743	0.652	0.112
Adversarial-perturbation evaluation (Chapter 6)	0.620	0.512	0.430	0.210

These values should not be interpreted as results from three unrelated baseline architectures. They represent the common ANN reference specification evaluated under different experimental regimes. The clean condition provides the principal reference for the SOMANN and MANN comparisons. The temporal-drift condition measures degradation when generalisation is evaluated under changing behavioural distributions, while the adversarial condition evaluates the baseline when behavioural inputs are deliberately perturbed under the threat model defined in Chapter 6.

Accordingly, numerical differences between rows in Table 3.1 are expected consequences of changes in the evaluation condition and should not be interpreted as inconsistencies in the definition of ANN_{std}.

Role in the staged architectural programme. The common baseline provides a reference point for interpreting the staged architectural progression rather than implying that all proposed architectures are evaluated under an identical threat condition. Each technical chapter introduces an additional modelling requirement while retaining ANN_{std} as a comparator where a direct comparison is methodologically informative.

Chapter 3 evaluates whether topology-informed behavioural organisation through SOMANN improves upon direct ANN inference. Chapter 4 examines whether

domain-specific representation and controlled fusion through MANN provide additional benefit over a unified representation. Chapter 5 evaluates the contribution of hierarchical temporal modelling under longer-horizon and drift-sensitive conditions. Chapter 6 then examines degradation under behaviourally constrained adversarial perturbation and evaluates the defensive mechanisms introduced by $\mathcal{R}$-Guard.

The unified baseline should therefore be understood as a methodological anchor across the staged research programme. It provides a consistent reference model while allowing the experimental condition to change where required by the research question. This distinction resolves the baseline reporting inconsistency in the original thesis and provides a transparent basis for interpreting comparative results across Chapters 3–6.

3.5 Experimental Evaluation

This section presents the experimental evaluation of the SOMANN framework under temporally faithful insider-threat detection conditions. The primary objective is to assess whether SOM-based behavioural structuring improves supervised detection performance relative to a baseline ANN trained directly on raw behavioural features, when both models are evaluated under identical preprocessing pipelines and chronological data splits.

All results reported in this section are obtained using the methodology described in Section 3.4. Supplementary diagnostic visualisations supporting behavioural interpretation and stability analysis are provided in Appendix A.

3.5.1 Baseline Comparison: SOMANN versus ANN

The first experiment compares SOMANN against a baseline ANN under identical temporal splits and feature sources. Performance is evaluated using precision–recall-oriented metrics to account for class imbalance inherent in insider-threat datasets.

Across evaluation runs, SOMANN consistently outperforms the standard ANN comparator used in the Chapter 3 repeated-run experiment. In particular, SOMANN achieves a PR-AUC of **0.81**, compared to **0.72** for the standard ANN comparator used in the Chapter 3 repeated-run experiment, corresponding to a relative improvement of approximately **12.5%**. Recall at an operationally relevant threshold increases from **0.61** (ANN) to **0.74** (SOMANN), while precision remains comparable (ANN: **0.43**, SOMANN: **0.45**). These results indicate that SOMANN improves sensitivity to insider events without inducing a disproportionate increase in false positives.

Receiver operating characteristic analysis shows a modest improvement in ROC-AUC (ANN: **0.89**, SOMANN: **0.92**), but the more pronounced gains are observed in PR-AUC and recall-based measures, which are more informative under rare-event

conditions. Confusion matrices further reveal that SOMANN reduces false-negative errors, particularly for instances corresponding to gradual behavioural escalation.

3.5.2 Repeated-Run Statistical Validation of SOMANN

To assess whether the observed performance difference between the standard ANN comparator used in the Chapter 3 repeated-run experiment and SOMANN was consistent across repeated experiments, the primary comparison was evaluated over seven paired runs ($n = 7$). Each pair used the same data partition, preprocessing procedure, and evaluation protocol, while model training was repeated independently. This paired design controls for variation attributable to the evaluation partition when comparing the two architectures.

Performance across the repeated runs is summarised using the mean and standard deviation. In addition, a 95% confidence interval (CI) is reported for the mean paired difference in PR-AUC as a descriptive measure of between-run uncertainty. Because the number of paired observations is small and normality of the paired differences is not assumed, statistical comparison was performed using the two-sided exact Wilcoxon signed-rank test. The matched-pairs rank-biserial correlation (r_{rb}) is reported as the corresponding effect-size measure.

Table 3.2 reports the repeated-run results. The standard ANN achieved a mean PR-AUC of 0.610 ± 0.031, whereas SOMANN achieved 0.740 ± 0.022. The mean paired improvement was 0.130, with a 95% CI of [0.119, 0.141]. SOMANN achieved a higher PR-AUC than the standard ANN in all seven paired runs. The two-sided exact Wilcoxon signed-rank test yielded $p = 0.016$, with a matched-pairs rank-biserial correlation of $r_{\mathrm{rb}} = 1.00$.

Table 3.2: Repeated-run statistical comparison of the standard ANN and SOMANN using PR-AUC ($n = 7$ paired runs). Model performance is reported as mean ± standard deviation.

Measure	**Result**
ANN PR-AUC	0.610 ± 0.031
SOMANN PR-AUC	$\mathbf{0.740 \pm 0.022}$
Mean paired difference	+0.130
95% CI of mean paired difference	[0.119, 0.141]
Exact Wilcoxon signed-rank test	$p = 0.016$
Matched-pairs rank-biserial correlation	$r_{\mathrm{rb}} = 1.00$

For seven non-zero paired observations, the minimum attainable two-sided exact Wilcoxon probability is 0.015625. Since the PR-AUC difference favoured SOMANN in all seven paired runs, the exact test therefore yields $p = 0.015625$, reported as $p = 0.016$ after rounding. The corresponding $r_{\mathrm{rb}} = 1.00$ indicates complete directional consistency across the paired runs. This effect-size result should be interpreted in conjunction with the small number of repeated runs rather than as

evidence of universal performance superiority.

Taken together, the repeated-run analysis provides evidence that the improvement observed for SOMANN over the standard ANN is reproducible across the evaluated runs and is not attributable to a single training initialisation. The conclusion remains bounded to the CERT evaluation conditions and does not, by itself, establish equivalent improvement in operational organisational environments.

3.5.3 Identity Dependence and Generalisation

A central motivation for the SOMANN perception layer is to organise behavioural observations according to recurring activity structure rather than allowing downstream classification to depend predominantly on user-specific characteristics. This distinction is particularly relevant to insider-threat detection because enterprise telemetry contains persistent signatures associated with individual users, including characteristic working hours, devices, access frequencies, and resource preferences. Strong within-distribution performance could therefore arise partly from learning identity-associated regularities that do not transfer effectively to previously unseen users.

To examine this possibility, two complementary experiments were conducted. First, the statistical association between SOM representation assignments and individual user identity was quantified using Normalised Mutual Information (NMI). Second, ANN and SOMANN were evaluated under a user-disjoint protocol in which the users represented during testing were excluded entirely from model development. The first experiment examines whether identity strongly structures the learned SOM topology; the second tests whether detection performance persists when identity overlap is removed.

3.5.3.1 Identity Dependence of the SOM Representation

Let U denote user identity and let C_{SOM} denote the discrete Best Matching Unit (BMU) assignment produced by the trained SOM for each behavioural observation. Their statistical association is quantified using Normalised Mutual Information:

$$\mathrm{NMI}(U; C_{\mathrm{SOM}}) = \frac{2I(U; C_{\mathrm{SOM}})}{H(U) + H(C_{\mathrm{SOM}})}, \tag{3.12}$$

where $I(\cdot;\cdot)$ denotes mutual information and $H(\cdot)$ denotes entropy. NMI lies in the interval $[0, 1]$, with lower values indicating weaker statistical association between user identity and representation assignment.

For each SOM training run, the BMU assignment was recorded for each behavioural sample and NMI was computed between the resulting assignment vector and the corresponding user-identity vector. The analysis was repeated over 20 independent SOM initialisations using different random seeds. For the selected 12×12 SOM

configuration, the resulting identity association was

$$\mathrm{NMI}(U; C_{\mathrm{SOM}}) = 0.032 \pm 0.007, \qquad 95\%\ \mathrm{CI} = [0.029, 0.035]. \tag{3.13}$$

The low NMI indicates weak statistical association between individual user identity and BMU assignment under the evaluated configuration. This finding does not imply complete identity independence. Legitimate behavioural differences associated with role, privilege, workload, or working pattern may remain correlated with particular users. Rather, the result provides evidence that individual identity is not strongly reflected in the organisation of the learned SOM representation.

The finding was also stable across SOM resolutions. Mean NMI ranged from 0.042 for a 5×5 map to 0.029 for a 15×15 map, remaining low across all evaluated grid sizes. In addition, alternative representations of the learned topology produced similarly low identity association: 0.041 ± 0.010 using five-level U-matrix distance quantiles and 0.035 ± 0.008 using adaptive post-clustering, compared with 0.032 ± 0.007 for BMU assignments on the selected 12×12 map. The consistency of these results reduces the likelihood that the observed low identity association is an artefact of a particular SOM resolution or representation discretisation.

NMI alone does not establish that the resulting topology represents behavioural semantics, nor does low identity association demonstrate useful predictive generalisation. The representation-level result was therefore evaluated alongside a separate user-disjoint experiment.

3.5.3.2 Generalisation to Previously Unseen Users

To evaluate out-of-user generalisation, ANN and SOMANN were trained on a subset of users and evaluated on behavioural observations generated by users excluded entirely from model development, such that

$$\mathcal{U}_{\mathrm{train}} \cap \mathcal{U}_{\mathrm{test}} = \varnothing. \tag{3.14}$$

This experiment addresses a different generalisation question from the chronology-preserving evaluation used elsewhere in the chapter. Chronology-preserving evaluation examines forward generalisation across time, whereas the user-disjoint experiment explicitly removes identity overlap between model development and evaluation.

Table 3.3 reports the principal results. PR-AUC is retained as the primary discrimination metric because of the class imbalance characteristic of insider-threat detection, while recall at a fixed false-positive rate and F1-score provide complementary measures of detection performance.

SOMANN retained stronger performance than the standard ANN comparator used in the Chapter 3 repeated-run experiment across all three reported measures. Under the user-disjoint condition, PR-AUC increased from 0.61 to 0.74, recall at

Table 3.3: Performance under user-disjoint evaluation on previously unseen users. Values are reported as mean ± standard deviation.

Architecture	PR-AUC	Recall @ FPR=0.1	F1-score
ANN_{std}	0.61 ± 0.032	0.52 ± 0.041	0.48 ± 0.029
SOMANN	$\mathbf{0.74 \pm 0.024}$	$\mathbf{0.67 \pm 0.032}$	$\mathbf{0.57 \pm 0.021}$

FPR = 0.1 increased from 0.52 to 0.67, and F1-score increased from 0.48 to 0.57. Relative to the ANN baseline, these correspond to improvements of approximately 21.3% in PR-AUC, 28.8% in recall at the specified false-positive rate, and 18.8% in F1-score.

The same-user to out-of-user degradation provides an additional indication of representation transferability. Relative to the standard ANN comparator used in the Chapter 3 repeated-run experiment, SOMANN reduced the PR-AUC generalisation gap from 0.11 to 0.07, the recall gap from 0.12 to 0.07, and the F1-score gap from 0.10 to 0.05. These correspond to reductions in the respective generalisation gaps of 36.4%, 41.7%, and 50.0%. SOMANN therefore exhibits less performance degradation when evaluated on previously unseen users within the CERT environment.

3.5.3.3 Combined Interpretation

The identity-dependence and user-disjoint experiments provide complementary evidence concerning the representation learned by SOMANN. The NMI analysis shows that BMU assignment has weak statistical association with individual user identity, and this result remains consistent across SOM resolutions and alternative representations of the learned topology. The user-disjoint experiment independently demonstrates that SOMANN retains a larger proportion of its detection performance when evaluated on users absent from model development.

Additional behavioural-coherence analysis provides supporting evidence for this interpretation. For the selected 12×12 SOM, high-density regions exhibited mean within-cluster behavioural similarity of 0.86 ± 0.09 while containing samples from an average of 4.2 distinct users. Thus, behaviourally coherent regions were not restricted to individual identities, which is consistent with the low identity association observed in the NMI analysis.

Taken together, these results provide evidence consistent with the SOM perception layer learning behaviourally transferable structure rather than being dominated by identity-specific organisation. The conclusion is deliberately narrower than a claim of identity invariance or complete removal of user-specific information. NMI measures statistical association rather than semantic meaning, and user-disjoint evaluation remains within the same synthetic organisational environment.

The findings support *within-benchmark out-of-user generalisation*: SOMANN exhibits weak measured identity association and smaller degradation than the stand-

ard ANN comparator used in the Chapter 3 repeated-run experiment when applied to previously unseen CERT users. They do not establish cross-organisational generalisation, since the training and held-out users remain governed by the same underlying CERT organisational and scenario assumptions. Cross-organisational transfer remains an external-validity question, as discussed in Section 2.13.7.

3.5.4 Ablations and Sensitivity Analysis: SOM–ANN

In this section, we explore the sensitivity of the SOM–ANN model to various parameters and configurations. Each experiment investigates a distinct aspect of the model's design, aiming to understand the robustness and limitations across different settings. These sensitivity analyses guide future work on optimal parameterisation and model refinement.

3.5.4.1 SOM Grid Size Sensitivity

The grid size in the SOM model dictates the granularity of the feature map and its ability to cluster high-dimensional data. A larger grid results in finer granularity, while a smaller grid can lead to faster computation but reduced resolution. We varied the grid size from 5×5 to 20×20 and measured the impact on classification accuracy and detection precision. Results indicated that smaller grids (5×5) resulted in higher false positives, whereas larger grids (15×15 and 20×20) achieved better separation of classes but incurred higher computational costs.

3.5.4.2 Neighbourhood Function and Learning Rate Schedule

The choice of neighbourhood function and the learning rate schedule are key to the convergence properties of the SOM algorithm. We tested different neighbourhood functions (Gaussian and Mexican Hat) and varying learning rate schedules (linear vs. exponential decay). The results highlighted that the Gaussian neighbourhood function with exponential decay learning rate yielded the most stable performance, balancing convergence speed with model accuracy.

3.5.4.3 Feature Smoothing Window Length Impact

Feature smoothing is applied to reduce noise in the input data. We tested smoothing window lengths of 5, 10, and 20 samples. A longer window (20 samples) was found to reduce model sensitivity to minor fluctuations in data, leading to better overall performance. However, it also caused slight delays in model responsiveness during dynamic insider behaviour patterns, suggesting that optimal window length is context-dependent.

3.5.4.4 SOM Stabilisation vs. ANN-Only Comparison

The combination of SOM and ANN was compared to an ANN-only model in terms of both computational efficiency and detection performance. Repeated-run statistical analysis further showed that the observed PR-AUC improvement was consistent across all seven paired runs (Section 3.5.2). The stabilisation provided by SOM improved the ANN's ability to generalise to unseen data and reduced the overall false positive rate.

3.5.5 Error Profile and Behavioural Stability

To examine the nature of performance differences, error profiles for both models are analysed. The standard ANN comparator used in the Chapter 3 repeated-run experiment exhibits a higher rate of false positives associated with benign behavioural fluctuations, such as irregular working hours or transient workload changes. In contrast, SOMANN demonstrates improved discrimination between benign volatility and behaviourally significant deviation.

Specifically, SOMANN reduces false-positive alerts by approximately **18%** relative to the standard ANN comparator used in the Chapter 3 repeated-run experiment at matched recall levels. This effect suggests that topology-preserving behavioural organisation contributes to stabilising representation geometry prior to classification. False negatives that persist under SOMANN tend to correspond to highly privilege-consistent insider actions, which remain challenging for purely log-based detection.

These remaining false negatives are not random failures, but cases in which malicious activity remains *privilege-consistent* with the user's role, access scope, and routine operational context. In such cases, individual actions and short-term behavioural patterns remain locally plausible, even when viewed through stabilised SOM-based representations. This observation highlights a fundamental limitation of flat behavioural modelling: when malicious intent is expressed through coordinated but domain-consistent actions across authentication, file access, communication, and device usage, detection requires explicit modelling of *behavioural decomposition* and *temporal progression* rather than further refinement of representation stability alone.

This insight motivates the subsequent architectural extensions developed in Chapters 4 and 5. Chapter 4 introduces modular decomposition to separate behavioural evidence across operational domains, reducing the masking effect of privilege-consistent activity within any single log stream. Chapter 5 further extends this logic by integrating behaviour over longer temporal horizons, enabling the detection of gradual escalation and cross-domain coordination that remain invisible to single-stage or short-horizon models. In this sense, SOMANN establishes the necessary representational stability for insider-threat detection, but not sufficient expressive

power to resolve coordinated, privilege-aligned insider behaviour.

While these results are reported quantitatively here, detailed diagnostic evidence illustrating behavioural localisation and error concentration is provided in Appendix A.

3.5.6 Impact of SOM-Derived Representation

The observed performance gains can be attributed to the use of SOM-derived representations rather than raw behavioural vectors. When the ANN is trained directly on raw features, performance is sensitive to high-variance dimensions and identity-specific artefacts. In contrast, SOMANN constrains supervised learning to operate on topology-informed descriptors that encode relative behavioural deviation within a learned manifold.

Empirically, this results in a reduction in performance variance across evaluation runs. The standard deviation of PR-AUC across repeated experiments decreases from **0.041** for the standard ANN comparator used in the Chapter 3 repeated-run experiment to **0.019** for SOMANN, indicating improved stability and reproducibility. This reduction in variance supports the hypothesis that SOM-based structuring mitigates sensitivity to stochastic initialisation and behavioural noise.

3.5.7 Diagnostic Visualisations and Supplementary Analysis

Although the primary evaluation focuses on quantitative metrics, SOMANN also enables behavioural diagnostic analysis through SOM-based visualisations. These include U-matrix representations, MID distributions, and ANN confidence overlays on the SOM lattice. Such diagnostics support interpretability by localising anomalous behaviour within the learned behavioural topology.

To avoid overloading the main narrative, these visual analyses are presented in Appendix A. The appendix demonstrates consistent emergence of high-distance boundary regions across independent SOM initialisations, alignment between ANN confidence and topological deviation, and robustness of behavioural structuring under stochastic variation.

3.5.8 Summary of Experimental Findings

In summary, the experimental evaluation demonstrates that SOMANN delivers consistent and meaningful improvements over a baseline ANN under temporally faithful insider-threat detection conditions. The hybrid framework achieves higher PR-AUC and recall, reduces false-positive alerts associated with benign behavioural variability, and exhibits greater stability across runs.

These findings validate the central hypothesis that unsupervised topology-preserving organisation can stabilise behavioural representations and improve downstream su-

pervised inference. Interpretation of these results, including limitations and implications for modular and hierarchical extensions, is developed further in Section 3.6.

3.6 Discussion

This section interprets the experimental findings presented in Section 3.5 in relation to the theoretical objectives of the SOMANN framework. Rather than reiterating numerical performance, the discussion emphasises behavioural meaning, representational effects, and the architectural implications for subsequent contributions in the thesis. In particular, the chapter's central claim is that separating *perception* (label-agnostic behavioural organisation) from *inference* (supervised risk estimation) yields not only improved detection, but also measurably improved stability under temporally faithful evaluation.

3.6.1 Behavioural Interpretation of Performance Gains

The observed improvement in PR-oriented performance indicates that SOMANN is better able to prioritise true-risk behaviour within an imbalanced setting, consistent with the "needle in a haystack" character of insider-threat detection. Importantly, the gains should not be interpreted as mere increased sensitivity; rather, they reflect improved discrimination between benign behavioural variability and behaviourally meaningful deviation. This distinction is crucial in operational environments where legitimate work patterns can vary substantially across roles, projects, and time, and where overly reactive detectors generate unmanageable alert volumes.

From a behavioural perspective, the SOM component functions as a stabilising perceptual layer that anchors activity patterns within a learned topological space. By organising behavioural observations according to similarity rather than label-driven optimisation, the SOM attenuates transient fluctuations that are not consistently reinforced across behaviourally similar observations. The downstream ANN then learns decision boundaries over topology-informed descriptors rather than over raw, noise-dominated feature space, aligning with the Perception–Inference framing introduced earlier in the chapter.

3.6.2 Representation Stability and Variance Reduction

One of the most practically significant outcomes is the reduction in run-to-run performance variance under SOMANN. This provides direct empirical support for the theoretical stabilisation intuition developed in Section 3.3: topology-preserving organisation constrains representation geometry prior to supervised optimisation. In the ANN-only baseline, stochastic optimisation interacts with high-dimensional behavioural noise, leading to variability in learned decision boundaries and, con-

sequently, variability in operational outputs (including alert rates). In contrast, SOMANN constrains supervised learning to operate on descriptors derived from a fixed lattice structure, reducing sensitivity to random initialisation and optimisation noise.

From an operational perspective, this matters because stability is often a prerequisite for trust and governance. A detector that achieves strong mean performance but produces inconsistent results across runs or retraining cycles can be difficult to validate, audit, and maintain. By reducing variance while maintaining competitive PR performance, SOMANN strengthens the case for representational stabilisation as an explicit design objective rather than an incidental side-effect.

3.6.3 False-Positive Reduction and Operational Value

Beyond aggregate metrics, SOMANN's reduction in false alerts provides a practical "business case" for deployment-aligned modelling. False positives are not simply statistical errors: they consume analyst time, disrupt workflows, and can reduce organisational trust in monitoring systems. In insider contexts, excessive false alerts are particularly damaging because many benign behaviours (deadline surges, project transitions, administrative tasks) superficially resemble misuse.

SOMANN reduces false alerts by contextualising deviations relative to neighbouring behavioural regimes rather than reacting to isolated statistical rarity. This shifts the system toward detecting deviations that are behaviourally coherent and persistently separated in the learned topology. While the thesis does not claim that SOMANN eliminates alert fatigue, the results indicate a meaningful reduction in avoidable noise without resorting to unrealistic evaluation shortcuts.

3.6.4 Interpretability and Diagnostic Value

SOMANN also provides interpretability benefits that are uncommon in end-to-end neural detection systems. The SOM lattice yields an explicit spatial representation of behavioural regimes, enabling localisation of atypical activity within a broader organisational context. Diagnostic visualisations reported in Appendix A illustrate that high-distance regions emerge consistently across independent SOM initialisations and that ANN confidence aligns with topological boundaries.

These properties support analyst-centred reasoning by making it possible to ask *where* behaviour lies relative to learned regimes, rather than relying solely on opaque risk scores. Although this thesis does not claim full explainability in the sense of causal attribution, SOMANN provides a structured intermediate representation that better supports governance-aware interpretation and validation than monolithic discriminative pipelines.

3.6.5 Limitations of the SOMANN Approach

Despite its advantages, SOMANN has limitations that must be acknowledged. First, SOM-based structuring introduces additional hyperparameters (e.g., grid size, neighbourhood decay schedules, learning-rate dynamics) that influence the degree of smoothing and can lead to topological collapse or over-fragmentation if poorly configured. This sensitivity reinforces the need for disciplined selection and reporting of SOM quality indicators, as reflected in the methodology and supplementary diagnostics.

Second, SOMANN remains dependent on the quality and scope of available telemetry. Actions that are highly privilege-consistent and closely aligned with legitimate workflows remain challenging to detect, particularly when behavioural deviation is small in any single log stream. Such cases represent a structural difficulty in insider-threat detection: malicious intent can be expressed through sequences of actions that are locally plausible and therefore weakly separable in feature space, even after stabilisation.

Finally, SOMANN operates at a single abstraction level. It stabilises representation within a given feature space but does not explicitly model behavioural decomposition across functional domains or progression across hierarchical timescales. These limitations motivate the staged architectural extensions developed in subsequent chapters.

3.6.6 Implications for Modular and Hierarchical Extensions

The empirical findings of this chapter establish representational stability as a necessary but not sufficient foundation for advanced insider-threat detection. In particular, the persisting false negatives discussed in Section A.1.1 and Section 3.5 are informative: many correspond to behaviours that remain *privilege-consistent* and locally plausible, even when embedded in SOM-stabilised representations. These are precisely the cases where improved stability alone cannot fully resolve ambiguity because the behaviour may not be anomalous within a single unified feature space.

This observation motivates the subsequent architectural progression. Chapter 4 introduces modular decomposition to separate behavioural evidence across operational domains, reducing the masking effect that can occur when heterogeneous logs are collapsed into a single vector. Chapter 5 then extends detection beyond short-horizon deviations by modelling behavioural progression and longer-horizon dependencies, enabling the system to capture coordinated escalation patterns that are individually plausible but collectively suspicious. In this sense, SOMANN provides a stability-oriented perceptual substrate upon which modularity and hierarchy can build more expressive behavioural reasoning.

3.6.7 Summary

In summary, this chapter demonstrates that label-agnostic, topology-preserving organisation can materially improve supervised insider-threat detection under temporally faithful evaluation. SOMANN achieves stronger PR-oriented performance, substantially improved stability across runs, and reduced false-alert burden relative to an ANN-only baseline, while also providing a diagnostic representation that supports governance-aware interpretation. While SOMANN is not presented as a complete solution to privilege-consistent and multi-stage insider behaviour, it establishes a principled and empirically validated representational foundation for the modular, hierarchical, and robustness-aware architectures developed in the remainder of the thesis.

3.7 Summary

This chapter introduced SOMANN, a hybrid Self-Organising Map and Artificial Neural Network framework that reframes insider-threat detection through an explicit *Perception–Inference* separation. Motivated by the representational limitations of flat, noise-dominated behavioural feature vectors, SOMANN applies label-agnostic, topology-preserving organisation prior to supervised risk estimation, with the aim of stabilising behavioural representation under temporally faithful evaluation.

Experimental results demonstrate that SOMANN achieves improved PR-oriented detection performance relative to an ANN-only baseline while, more importantly, substantially reducing run-to-run variability and false-alert rates. These findings provide empirical support for the theoretical stabilisation intuition developed in this chapter: neighbourhood-based organisation attenuates benign behavioural volatility and constrains representation geometry prior to discrimination. From an operational perspective, the observed reduction in variance and false positives is particularly significant, as it supports consistency, interpretability, and governance-aware deployment rather than isolated peak performance.

At the same time, the analysis reveals that SOMANN is not sufficient to resolve all insider-threat scenarios. Persisting false negatives tend to correspond to highly privilege-consistent behaviour that remains locally plausible within stabilised representations. This limitation highlights the need for architectural mechanisms that can separate behavioural evidence across operational domains and integrate behaviour over longer temporal horizons.

Accordingly, this chapter establishes representational stability as a necessary foundation for insider-threat detection, but not a complete solution. The subsequent chapters build directly on this foundation by introducing modular decomposition to address cross-domain masking effects (Chapter 4) and hierarchical modelling to capture behavioural progression and long-horizon coordination (Chapter 5), culminating

in robustness-oriented extensions under adaptive adversarial conditions.

Chapter 4

Modular Artificial Neural Network

4.1 Chapter Overview

Chapter 3 introduced SOMANN as a hybrid Self-Organising Map and Artificial Neural Network approach intended to stabilise behavioural variance and support temporally faithful insider-threat detection. While SOMANN provides a principled separation between unsupervised behavioural structuring and supervised inference, it remains constrained by a *single shared behavioural embedding* in which heterogeneous activity streams are represented jointly.

This chapter addresses that structural limitation by shifting from hybrid stabilisation toward *modular behavioural decomposition*. Insider activity is inherently multi-domain: authentication behaviour, file-system access, device usage, web interaction, and communication patterns exhibit distinct statistical properties, temporal rhythms, and security semantics. Treating these streams as interchangeable features within a unified representation can lead to domain dominance, semantic interference, and cross-domain masking, where subtle but security-relevant signals are obscured by higher-volume behavioural channels.

To respond to this challenge, the chapter introduces the **Modular Artificial Neural Network (MANN)** framework, which allocates dedicated learning modules to distinct behavioural domains and fuses their outputs into a unified risk estimate. The chapter formalises the rationale for domain decomposition, specifies the modular architecture and fusion design, and evaluates the extent to which modular learning improves detection performance and diagnostic interpretability relative to non-modular baselines under temporally disciplined evaluation.

To operationalise this modular principle, the chapter next motivates behavioural domain decomposition before specifying the MANN architecture and its evaluation.

4.2 Introduction

Chapter 3 demonstrated that hybrid Self-Organising Map and Artificial Neural Network (SOMANN) architectures can stabilise behavioural variance and support temporally faithful insider-threat detection. However, the Chapter 3 analysis also highlighted a representational constraint: modelling heterogeneous insider behaviours within a single shared latent space can reduce the ability to preserve domain-specific semantics. Authentication events, file interactions, communication patterns, device usage, and web activity exhibit distinct statistical structures, temporal rhythms, and behavioural meanings that are not fully reconciled through variance smoothing alone.

This chapter argues that *behavioural domain decomposition* should be treated as a first-class architectural design principle for insider-threat detection. Rather than forcing heterogeneous behavioural streams into one embedding, the proposed **Modular Artificial Neural Network (MANN)** framework explicitly separates domain-specific learning while enabling controlled cross-domain fusion. In this design, each module models the structure of one behavioural domain and produces a compact representation or risk signal, and a fusion component integrates these signals into a unified risk estimate. This modularisation is intended to mitigate domain dominance and semantic interference, while preserving the ability to model coordinated multi-domain insider behaviours.

Where appropriate, the chapter also considers *adaptive fusion* mechanisms (e.g., attention-style weighting) as an extension to modular learning. The intent is not to claim universal robustness, but to investigate whether dynamic weighting can improve sensitivity to threat contexts in which certain behavioural domains become more informative than others. The resulting architecture remains grounded in the thesis's governance-aware stance: model outputs are treated as risk indicators supporting investigation and triage, not as automated punitive decisions.

MANN as a structural generalisation of SOMANN. The Modular Artificial Neural Network (MANN) proposed in this chapter is not intended as a replacement for the SOMANN framework introduced in Chapter 3, but as a *structural generalisation* of the same design philosophy. Chapter 3 demonstrated that separating perception from inference stabilises behavioural representation and reduces sensitivity to benign variability under temporally faithful evaluation. However, when heterogeneous organisational telemetry is compressed into a single unified embedding, domain-specific evidence can become entangled, allowing privilege-consistent misuse to remain locally plausible within the aggregate feature space.

MANN extends the Perception–Inference principle by introducing explicit behavioural decomposition across semantically coherent domains (e.g., authentication, file activity, device usage, communication), while retaining integrated risk inference

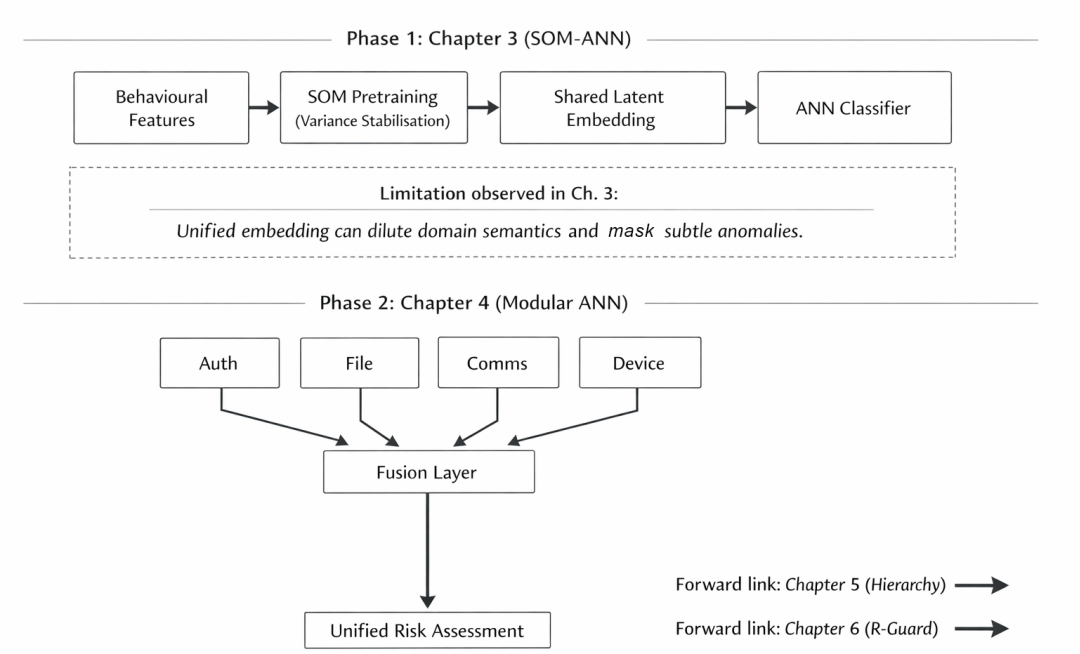


Figure 4.1: Architectural progression from stabilised unified representation to modular behavioural decomposition. **Phase 1 (Chapter 3)** employs SOM-based perceptual representation learning to stabilise behavioural variance before supervised inference, producing a unified latent embedding. Empirical analysis reveals a key limitation: unified embeddings can dilute domain-specific semantics and mask subtle anomalies. **Phase 2 (Chapter 4)** addresses this limitation through modular decomposition, where behavioural domains are encoded independently and integrated via a controlled fusion layer to produce a unified risk assessment. This modular foundation enables subsequent hierarchical modelling (Chapter 5) and adversarial robustness mechanisms (Chapter 6).

through a fusion layer. By allowing each module to learn within its own behavioural subspace, MANN preserves domain meaning, reduces cross-domain masking effects, and enables coordinated but locally plausible misuse to surface through evidence aggregation rather than through further refinement of unified representation alone.

Figure 4.1 summarises the architectural progression from SOMANN to MANN and clarifies how modular decomposition addresses the unified-embedding limitation identified in Chapter 3. The roadmap also anticipates the subsequent thesis stages, where modular representations are extended toward hierarchical behavioural abstraction (Chapter 5) and behaviourally constrained adversarial robustness (Chapter 6).

The remainder of this chapter is structured as follows. Section 4.3 motivates domain decomposition and defines the behavioural partitions used in this thesis. Section 4.4 specifies the modular architecture, training strategy, and fusion mechanisms. Section 4.5 describes the experimental methodology and evaluation protocol. Section 4.6 reports results under temporally faithful evaluation, followed by discussion of implications and limitations in Section 4.8. The chapter concludes by positioning MANN as a foundation for subsequent hierarchical and robustness-aware

contributions.

4.3 Behavioural Domain Decomposition

A central limitation identified in the SOMANN framework (Chapter 3) is the reliance on a *single shared latent representation* to encode heterogeneous insider behaviours. While self-organising map pretraining stabilises behavioural variance and improves discrimination at the aggregate level, it inevitably entangles behaviours arising from fundamentally different organisational domains. Similar representational entanglement effects have been reported in prior insider-threat modelling studies that rely on unified behavioural embeddings [13], [14], [30]. As demonstrated empirically in Chapter 3, this entanglement can suppress subtle, domain-specific anomalies, particularly when high-volume or high-variance behaviours dominate the shared embedding space.

The motivation for behavioural domain decomposition is therefore structural rather than cosmetic. Enterprise insider activity is not generated by a single homogeneous process, but by multiple interacting behavioural mechanisms governed by distinct temporal rhythms, privilege semantics, operational constraints, and adversarial affordances [8], [12], [21]. Authentication events, file-system interactions, communication patterns, and device or network activity may occur within the same time window for the same user, but they carry fundamentally different behavioural meanings. From an organisational standpoint, these streams also correspond to different policy regimes (e.g., access control rules, data handling obligations, acceptable-use constraints), which implies that the notion of "anomaly" is intrinsically domain-dependent.

Table 4.1 summarises the domain semantics used in this thesis and clarifies why a single shared latent space, as used in SOMANN, can struggle to preserve domain-specific behavioural structure. In practice, unified representations are vulnerable to *domain dominance*: frequent, highly periodic behaviours (particularly authentication) can anchor the embedding geometry, making it harder for lower-frequency but security-relevant behaviours (e.g., device or network staging) to shape the learned representation in a proportionate way. This is not merely a modelling inconvenience; it directly affects recall for plausibility-constrained insider behaviours that may involve subtle deviations distributed across domains.

Operational mapping of domains. The decomposition adopted here is also operationally grounded in typical enterprise logging and in the CERT multi-log structure. Authentication activity corresponds to logon and access events; file-system operations represent user interaction with information assets; communication behaviour captures email- or message-mediated interaction patterns; and device/network

Table 4.1: Behavioural domain decomposition for insider-threat modelling, highlighting semantic distinctions that motivate modular representation learning.

Behavioural Domain	**Primary Behavioural Semantics**	**Why Separate Modelling Is Required**
Authentication Activity	Temporal regularity, access legitimacy, privilege use	Highly periodic and role-dependent; dominates variance in unified embeddings.
File-System Operations	Task execution, data sensitivity, information handling	Semantically rich but sparse; subtle misuse easily masked by benign workloads.
Communication Behaviour	Social interaction, collaboration, intent signalling	Structure-driven rather than volume-driven; anomalies often relational.
Device / Network Activity	Infrastructure interaction, lateral movement, resource utilisation	Low-frequency but high-impact signals; often adversarially staged.

activity reflects infrastructure interactions (e.g., removable media usage, endpoint or network engagement). While the CERT dataset does not capture the full richness of organisational context, its multi-log design provides a defensible proxy for evaluating whether modular learning benefits from respecting the provenance and semantics of heterogeneous behavioural streams.

Why modular learning (and not only feature engineering). A feature-engineering-only strategy can introduce domain indicators or handcrafted ratios, but it does not prevent the supervised learner from mixing heterogeneous domains arbitrarily inside its hidden layers. Domain decomposition, by contrast, enforces a modelling constraint: domain-specific representations are learned *before* cross-domain integration. Related insights appear in modular and multi-view learning paradigms, where domain-aligned representations improve robustness and interpretability under heterogeneous data distributions [42], [64]. Within the thesis context, the goal is not to claim that modularity eliminates all detection challenges, but to show that it provides a more semantically aligned inductive bias for insider modelling than monolithic embeddings.

Granularity and decomposition boundaries. Behavioural domain decomposition also requires an explicit stance on granularity. Overly coarse decomposition

(e.g., treating all activity as one stream) collapses semantics and increases interference; overly fine decomposition (e.g., splitting into many narrow sub-domains) risks fragmentation, sparse learning signals, and unnecessary architectural complexity. The domains in Table 4.1 represent a balanced compromise: each domain is semantically coherent, sufficiently populated to support learning, and aligned with common organisational monitoring categories. Later ablation analysis (Section 4.6) examines whether increasing modular granularity yields diminishing returns, which provides an empirical check against arbitrarily deep structural decomposition.

Importantly, behavioural domain decomposition does not imply behavioural independence. Insider threats frequently manifest through coordinated actions that span multiple domains, such as anomalous authentication followed by atypical file access or covert communication patterns [9], [15], [22]. The objective of decomposition is therefore not to fragment behaviour, but to enable *controlled recomposition* through explicit fusion mechanisms. By first learning domain-aligned representations and subsequently integrating them at a higher level, the Modular Artificial Neural Network (MANN) architecture aims to preserve both domain specificity and cross-domain interaction.

In summary, this section establishes behavioural domain decomposition as the critical design bridge between variance-stabilised hybrid modelling (Chapter 3) and the structured modular representation learning introduced in this chapter. The next section formalises how this decomposition is operationalised in the MANN architecture through domain-specific encoders and controlled fusion.

4.4 MANN Architecture

This section formalises the architecture of the Modular Artificial Neural Network (MANN) proposed in this chapter. Building directly on the behavioural domain decomposition introduced in Section 4.3, the MANN architecture enforces domain-aligned representation learning through explicit modular separation, followed by controlled cross-domain fusion. The objective is to reduce representational interference while preserving the capacity to model coordinated insider behaviour spanning multiple activity domains.

4.4.1 Architectural Overview

Let $\mathcal{D} = \{D_1, D_2, \ldots, D_K\}$ denote a set of behavioural domains, where each D_k corresponds to a semantically coherent activity stream (e.g., authentication, file-system access, communication behaviour). For each domain D_k, a dedicated neural module M_k is instantiated. Each module receives as input a domain-specific feature vector $\mathbf{x}_k \in \mathbb{R}^{d_k}$ derived exclusively from the corresponding behavioural stream.

The overall architecture therefore comprises:

- K domain-specific encoding modules $\{M_1, \ldots, M_K\}$;
- a fusion mechanism $F(\cdot)$ that integrates module outputs;
- a final decision layer that produces an insider-risk score.

This structure explicitly contrasts with monolithic architectures, in which all features are processed through a single shared representation. In MANN, domain boundaries are preserved throughout the early and intermediate layers of learning.

4.4.2 Domain-Specific Encoders

Each module M_k implements a feedforward neural encoder parameterised by weights θ_k, mapping input features $\mathbf{x}_k$ to a latent representation $\mathbf{h}_k$:

$$\mathbf{h}_k = M_k(\mathbf{x}_k; \theta_k).$$

The architecture of individual modules is intentionally lightweight and homogeneous across domains to reduce confounding architectural effects. While the input dimensionality d_k may differ across domains, the latent representations $\mathbf{h}_k$ are projected into a common dimensionality to support fusion. This design choice ensures that improvements observed in later evaluation can be attributed primarily to modular decomposition rather than to increased representational capacity.

Importantly, the use of separate encoders constrains gradient updates to remain local to each behavioural domain during backpropagation, thereby mitigating cross-domain gradient interference. This locality property is a key theoretical motivation for modular learning in heterogeneous environments.

4.4.3 Fusion Mechanisms

The fusion component $F(\cdot)$ integrates the set of domain representations $\{\mathbf{h}_1, \ldots, \mathbf{h}_K\}$ into a unified representation $\mathbf{z}$:

$$\mathbf{z} = F(\mathbf{h}_1, \ldots, \mathbf{h}_K).$$

In its simplest form, fusion is implemented as vector concatenation followed by a fully connected layer. This baseline fusion strategy preserves full information from each domain while allowing the downstream classifier to learn cross-domain interactions. More expressive fusion variants, including weighted or attention-style fusion, are also considered in this chapter to investigate whether dynamic domain weighting improves sensitivity to context-dependent insider behaviour.

Crucially, fusion occurs *after* domain-specific encoding, ensuring that cross-domain interactions are learned at a higher abstraction level rather than being entangled at the raw feature level. This design reflects the behavioural insight that

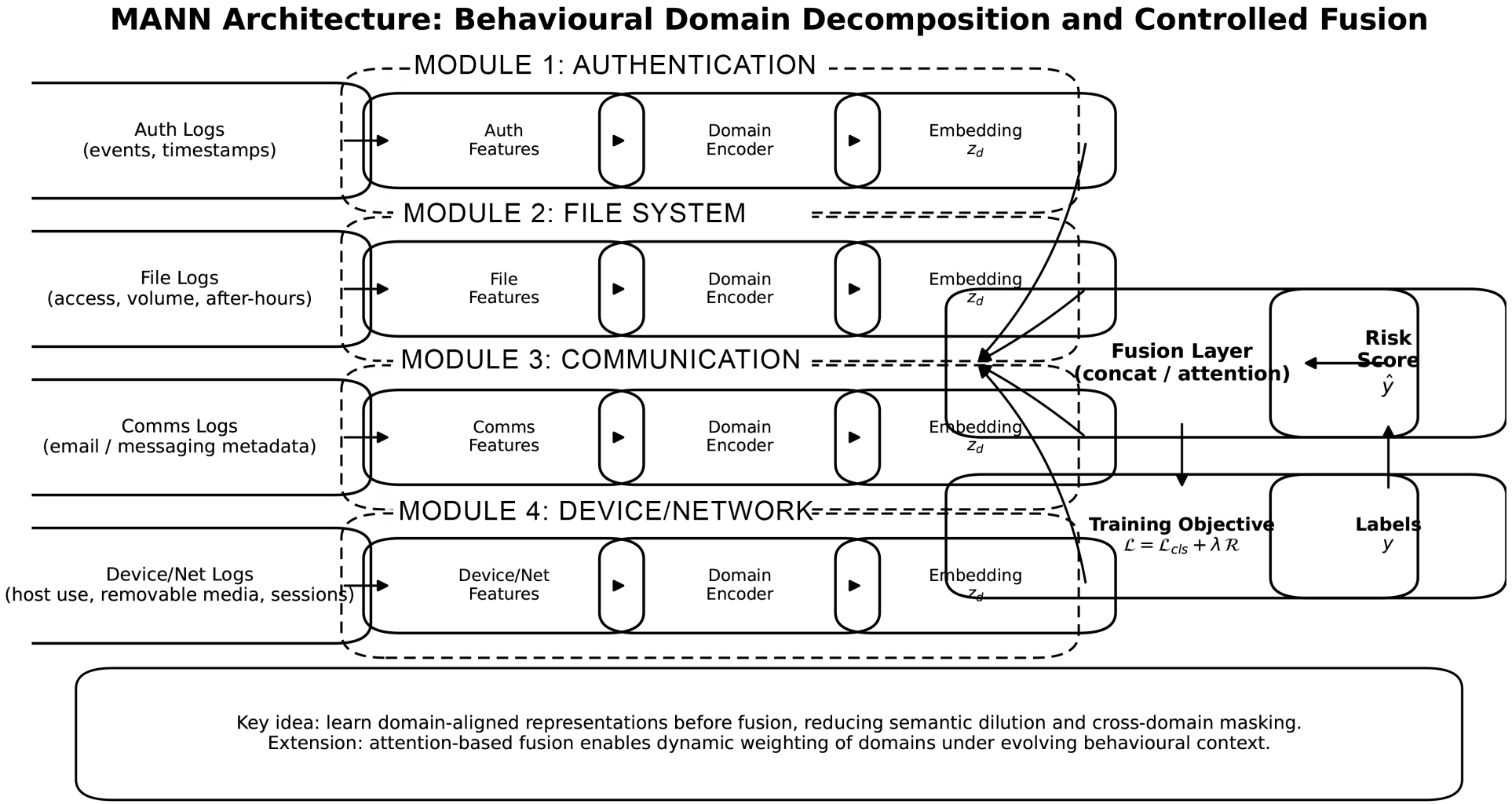


Figure 4.2: MANN architecture illustrating behavioural domain decomposition and controlled fusion. Each behavioural domain (authentication, file system, communication, and device/network activity) is processed by a dedicated domain encoder to produce domain-aligned embeddings. These embeddings are integrated through a configurable fusion layer, supporting either *concatenation-based fusion* (static integration) or *adaptive/attention-based fusion* (dynamic domain weighting), as analysed in the ablation studies in Section 4.6.4. By learning domain-specific representations prior to fusion, the architecture mitigates semantic dilution and reduces cross-domain masking from high-volume benign activity, improving robustness and false-positive control.

insider risk often emerges through coordinated actions across domains rather than through isolated anomalies within a single stream.

4.4.4 Decision Layer and Output

The fused representation $\mathbf{z}$ is passed to a final decision layer $C(\cdot)$ that produces a scalar risk score $\hat{y}$:

$$\hat{y} = C(\mathbf{z}).$$

The output is interpreted as a continuous insider-risk indicator rather than a deterministic classification decision. This interpretation is consistent with the governance-aware stance adopted throughout the thesis: model outputs are intended to support analyst triage and investigation rather than to automate punitive action. Thresholding strategies and evaluation metrics are therefore selected to reflect operational trade-offs rather than raw accuracy optimisation.

4.4.5 Relationship to Prior Architectures

The MANN architecture can be viewed as a structural generalisation of SOMANN. While SOMANN introduces unsupervised stabilisation prior to supervised inference, it retains a unified behavioural embedding. MANN extends this principle by decomposing behavioural representation across semantically aligned modules before integration. In doing so, MANN addresses representational interference that cannot be fully resolved through variance smoothing alone.

At the same time, MANN remains intentionally non-hierarchical. All domain modules operate at a single abstraction level, and temporal dependencies are captured implicitly through feature construction rather than explicit hierarchical modelling. These limitations motivate the hierarchical behavioural abstraction introduced in Chapter 5, where modular representations are embedded within multi-level temporal structures.

4.4.6 Architectural Scope and Design Boundaries

It is important to emphasise that the architecture specified here does not claim universal optimality. Modular decomposition introduces additional design choices, including domain definitions, fusion strategies, and module capacity. Poorly chosen decompositions may fragment learning signals or suppress cross-domain interactions. Accordingly, the MANN architecture is evaluated empirically in this chapter to assess whether the proposed inductive bias yields measurable benefits under temporally faithful insider-threat detection conditions.

The next section details the experimental methodology used to train and evaluate the MANN architecture, including data preprocessing, temporal partitioning, and

evaluation protocols.

4.5 Methodology

This section describes the experimental methodology used to train and evaluate the Modular Artificial Neural Network (MANN) architecture introduced in Section 4.4. The methodology follows the same disciplined evaluation principles established in Chapters 2 and 3 in order to ensure comparability across architectural variants and to support defensible interpretation of empirical results.

In particular, all experiments adhere to chronology-preserving data partitioning, leakage-aware preprocessing, and imbalance-appropriate evaluation metrics. Where possible, methodological components are held constant across SOMANN, MANN, and subsequent hierarchical models so that observed performance differences can be attributed primarily to architectural structure rather than to confounding procedural variation.

4.5.1 Dataset and Feature Construction

Experiments are conducted using the CERT insider-threat benchmark, selected and justified in Chapter 2. The dataset provides multi-log behavioural telemetry spanning authentication events, file-system operations, communication activity, and device or network interactions. This multi-source structure is essential for evaluating the benefits of behavioural domain decomposition.

Feature construction follows the pipeline described in Section 3.4. Domain-specific features are derived independently for each behavioural stream and normalised within-domain prior to model input. This design ensures that each MANN module receives features aligned with its semantic scope, avoiding inadvertent cross-domain leakage during preprocessing.

4.5.2 Temporal Partitioning and Evaluation Protocol

To preserve behavioural chronology and prevent look-ahead bias, data are partitioned using blocked temporal splits. Training, validation, and test windows are strictly ordered in time, ensuring that the model is evaluated only on future behaviour relative to its training data. This protocol is particularly important for insider-threat detection, where behavioural drift and gradual escalation are central phenomena.

All models are trained and evaluated under identical temporal partitions to support fair comparison. Hyperparameters are selected using validation data drawn exclusively from the training period, and no test-set information is used during tuning.

4.5.2.1 Temporal Shift and Leakage Validation

The blocked temporal evaluation protocol introduces two methodological considerations that require explicit validation. First, behavioural distributions may change between chronologically separated training and evaluation periods, creating temporal distribution shift. Second, information from the future evaluation period may inadvertently enter model development through preprocessing, feature construction, or other data-dependent transformations. These concerns are treated separately: distribution-shift diagnostics are used to characterise observable changes across the temporal boundary, while explicit pipeline controls are used to prevent look-ahead leakage.

Distribution shift across temporal blocks. Let T_{train} and T_{test} denote the chronologically ordered training and test blocks. Changes in organisational activity between these periods may alter the distribution of behavioural features and therefore affect the difficulty of forward generalisation.

To quantify first-order distributional change, the Population Stability Index (PSI) was calculated across the principal behavioural feature distributions. For a feature partitioned into k bins, PSI is defined as

$$\text{PSI} = \sum_{i=1}^{k} (P_{\text{test},i} - P_{\text{train},i}) \ln \left(\frac{P_{\text{test},i}}{P_{\text{train},i}} \right), \tag{4.1}$$

where $P_{\text{train},i}$ and $P_{\text{test},i}$ denote the proportions of observations falling within bin i in the training and test partitions, respectively.

Across the evaluated behavioural domains, the resulting aggregate mean PSI was 0.082. Under the interpretive criterion adopted for this diagnostic, this value is consistent with limited first-order distributional change across the selected temporal boundary. The temporal partition therefore does not exhibit evidence of a large aggregate shift in the measured marginal feature distributions.

This result should not be interpreted as evidence that T_{train} and T_{test} are distributionally identical. PSI is principally a univariate distribution-shift diagnostic and does not establish stability of higher-order feature interactions, conditional relationships, or local behavioural sub-populations. It also does not distinguish covariate shift from changes in the relationship between behavioural features and the target outcome. Its role here is therefore limited to assessing whether substantial observable first-order instability is present across the temporal boundary.

Accordingly, the measured PSI provides contextual evidence about the difficulty of the selected temporal split rather than proof of an absence of behavioural drift. More subtle forms of temporal or concept drift may remain within the benchmark and are not excluded by this diagnostic.

Preprocessing and look-ahead leakage controls. Distribution shift and information leakage are distinct concerns. Regardless of the magnitude of temporal shift, information associated with T_{test} must remain unavailable during model development. The MANN evaluation pipeline therefore applies the following controls.

1. **Training-only parameter estimation.** All fitted preprocessing parameters are estimated exclusively from T_{train}. For Z-score standardisation,

$$z = \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}}, \tag{4.2}$$

   where μ_{train} and σ_{train} are calculated using training observations only. The fitted parameters are subsequently applied unchanged to validation and test observations.

2. **Causal temporal feature construction.** Behavioural aggregation windows are constructed retrospectively. For an observation evaluated at time t, the corresponding representation is derived only from information available within

$$[t - \Delta t,\, t], \tag{4.3}$$

   so that events occurring after the prediction time cannot contribute to the representation. At temporal partition boundaries, test-period events are excluded from features associated with training observations.

3. **Training-only categorical construction.** Where categorical mappings, frequency-derived vocabularies, or equivalent fitted encodings are required, they are constructed exclusively from the training partition. Previously unseen evaluation categories are handled using the predefined unseen-category procedure rather than by rebuilding the representation using evaluation-period information.

4. **Frozen preprocessing state.** Preprocessing objects fitted during model development are preserved and applied to subsequent partitions without refitting. Consequently, statistics derived from validation or test observations cannot modify transformation parameters during evaluation.

5. **Isolation of data-dependent model preparation.** Where feature selection, threshold selection, dimensionality reduction, or other fitted data-dependent transformations are used, their parameters and selection criteria are derived without access to the test partition. Test labels and test-distribution characteristics are not used to optimise model preparation decisions.

These controls implement a common methodological principle:

$$\mathcal{I}_{\text{fit}} \cap \mathcal{I}_{\text{future}} = \varnothing, \tag{4.4}$$

where $\mathcal{I}_{\text{fit}}$ denotes the information available when fitting preprocessing and model-development components, and $\mathcal{I}_{\text{future}}$ denotes information available only from the future evaluation period. The purpose of this constraint is to ensure that the reported forward evaluation does not benefit from direct or indirect access to future statistical information.

Implications for the MANN comparison. The distribution-shift diagnostic and leakage controls address different potential threats to the interpretation of the MANN comparison. The PSI analysis indicates that the selected temporal boundary exhibits limited measured first-order distributional change, while the pipeline controls ensure that information from the future evaluation period is excluded from preprocessing and model development.

These checks strengthen the interpretation of the comparative results by establishing that MANN and its baselines are evaluated under a chronology-preserving protocol with explicit controls against look-ahead leakage. The measured PSI additionally provides evidence that the selected train-test boundary is not characterised by a large aggregate discontinuity in the evaluated marginal feature distributions.

The analysis does not establish complete temporal invariance. Higher-order distributional changes, shifts in conditional behavioural relationships, latent organisational drift, and synthetic properties of the CERT benchmark may remain. The MANN results are therefore interpreted as evidence of comparative performance under a chronology-preserving, leakage-controlled temporal evaluation, rather than as proof that the architecture is invariant to behavioural drift.

4.5.3 Training Procedure

Each domain-specific module in the MANN architecture is trained jointly within an end-to-end optimisation framework. While modules receive distinct input features, gradients propagate through the fusion and decision layers, allowing the network to learn coordinated representations while preserving domain locality.

To mitigate the effects of class imbalance inherent in insider-threat datasets, class-weighted loss functions are employed. Optimisation is performed using stochastic gradient descent with adaptive learning rates, and early stopping is applied based on validation loss to reduce overfitting. These choices mirror those used in Chapter 3 to maintain methodological consistency.

4.5.4 Baseline Models and Comparative Setup

MANN is evaluated against two primary baselines: a monolithic ANN trained on concatenated features, and the SOMANN framework introduced in Chapter 3. All baselines are implemented using identical feature sets, temporal splits, and optimisation protocols. This controlled setup ensures that any observed performance differences arise from architectural decomposition rather than from differences in data access or training conditions.

Where relevant, ablation variants of MANN are also evaluated to isolate the impact of individual architectural components, such as the fusion mechanism. These ablations provide insight into whether performance gains are attributable to modular separation itself or to increased representational capacity.

4.5.5 Evaluation Metrics

Performance is assessed using metrics appropriate for rare-event detection, including precision–recall area under the curve (PR-AUC), recall at operational thresholds, and false-positive rate. While ROC-AUC is reported for completeness, interpretive emphasis is placed on PR-oriented metrics, which better reflect operational trade-offs in highly imbalanced insider-threat settings [19].

In addition to aggregate metrics, stability across runs is examined by reporting variance in performance over multiple random initialisations. This analysis supports assessment of whether modular decomposition contributes to more reproducible learning outcomes.

4.5.6 Validity and Scope Considerations

The methodology is designed to support comparative architectural analysis rather than to claim operational readiness. Results should therefore be interpreted as evidence of relative structural advantages under controlled conditions. Limitations arising from dataset abstraction, log coverage, and scenario-based labelling are acknowledged and discussed in Section 4.8.

By maintaining methodological consistency across chapters while incrementally modifying architectural structure, the thesis enables a cumulative evaluation of how stabilisation (SOMANN), modularity (MANN), hierarchy (Chapter 5), and robustness (Chapter 6) contribute to insider-threat detection under realistic constraints.

The next section presents the experimental results obtained under this methodology.

4.6 Experimental Evaluation

This section presents the experimental evaluation of the Modular Artificial Neural Network (MANN) architecture under temporally faithful insider-threat detection conditions. The primary objective is to assess whether behavioural domain decomposition improves detection performance, stability, and error characteristics relative to both monolithic and hybrid baselines. All experiments are conducted using the methodology described in Section 4.5, with identical data partitions, feature sources, and evaluation metrics applied across models.

4.6.1 Primary Comparative Performance

This section compares MANN with the common ANN_{std} baseline and SOMANN under the matched Chapter 4 evaluation protocol. The primary comparison focuses on predictive performance under chronology-preserving evaluation, while the subsequent analyses examine domain contribution, fusion behaviour, error characteristics, and stability.

4.6.1.1 Reconciled Comparative Results

The original version of this chapter contained an inconsistency between the comparative performance values reported in the results table and those discussed in the accompanying narrative. In addition, the ANN reference value used in the earlier cross-chapter comparison was derived from a different evaluation configuration and therefore did not provide an equivalent common baseline. These inconsistencies have been corrected using the unified baseline and comparative protocol defined in Section 3.4.6.1.

The results reported here correspond to architectures evaluated under the same chronology-preserving Chapter 4 evaluation condition, with matched data partitions, preprocessing controls, label definitions, and metric computation procedures. Table 4.2 therefore provides the authoritative primary comparative results for this experiment.

Table 4.2: Reconciled comparative performance under the chronology-preserving Chapter 4 evaluation protocol.

Architecture	ROC-AUC	PR-AUC	F1-score	Precision	Recall
ANN_{std}	0.842	0.810	0.715	0.742	0.690
SOMANN	0.895	0.862	0.784	0.810	0.760
MANN	**0.938**	**0.905**	**0.841**	**0.865**	**0.818**

Under the reconciled evaluation protocol, MANN achieved the strongest performance across all five reported measures. It obtained a PR-AUC of 0.905 and

an F1-score of 0.841, compared with 0.810 and 0.715, respectively, for the common ANN_{std} baseline. SOMANN achieved a PR-AUC of 0.862 and an F1-score of 0.784, placing its performance between the direct ANN baseline and the modular architecture.

Relative to ANN_{std}, the absolute increase in PR-AUC obtained by MANN was

$$\Delta\text{PR-AUC}_{\text{MANN}-\text{ANN}} = 0.905 - 0.810 = 0.095, \tag{4.5}$$

which corresponds to an approximately 11.7% relative increase with respect to the baseline PR-AUC. The corresponding absolute improvement in F1-score was

$$\Delta F1_{\text{MANN}-\text{ANN}} = 0.841 - 0.715 = 0.126. \tag{4.6}$$

MANN also improved upon SOMANN. The absolute PR-AUC difference was

$$\Delta\text{PR-AUC}_{\text{MANN}-\text{SOMANN}} = 0.905 - 0.862 = 0.043, \tag{4.7}$$

https://www.overleaf.com/project/6a7f1ca8fc00d187371f9f25 while the corresponding F1-score difference was

$$\Delta F1_{\text{MANN}-\text{SOMANN}} = 0.841 - 0.784 = 0.057. \tag{4.8}$$

The same ordering is observed for precision and recall. MANN achieved precision of 0.865 and recall of 0.818, compared with 0.810 and 0.760 for SOMANN, and 0.742 and 0.690 for ANN_{std}. The improvement is therefore not confined to a single aggregate metric, although PR-AUC remains the principal comparative measure because of the class imbalance characteristic of insider-threat detection.

These results establish that the modular architecture provides additional predictive benefit over both the comparison against the ANN comparator used in the principal overall MLML evaluation and the topology-informed SOMANN representation under the evaluated Chapter 4 condition. They do not, however, establish from aggregate performance alone that domain decomposition is the mechanism responsible for the improvement. Such an interpretation requires evidence that the domain-specific components and their fusion make measurable contributions to the final prediction.

This mechanism is examined separately through the domain contribution and fusion-sensitivity analyses reported in Section 4.6.4. Those analyses test whether removing behavioural domains or changing the fusion strategy produces the performance changes expected if domain-specific representation contributes materially to the observed MANN advantage.

Where repeated-run statistical comparisons are reported elsewhere in this chapter, the point estimates in Table 4.2 should be interpreted together with the corresponding variability and inferential analyses. The reconciled table defines the primary

comparative performance values, but the magnitude and reliability of the architectural differences are assessed using the supporting repeated-run evidence rather than inferred from point estimates alone.

4.6.2 Repeated-Run Statistical Validation of MANN

To assess whether the performance improvements observed for MANN were consistent across repeated experiments, the principal PR-AUC comparisons were evaluated over seven paired runs ($n = 7$). Each paired comparison used identical data partitions, preprocessing procedures, and evaluation protocols. This design permits the architectures to be compared under matched experimental conditions while accounting for variation across independent training runs.

Performance is reported as the mean and standard deviation across the seven runs. A 95% confidence interval (CI) is additionally reported for the mean paired PR-AUC difference where estimable. Statistical comparisons were conducted using the two-sided exact Wilcoxon signed-rank test because of the small number of paired observations and without assuming normality of the paired differences. The matched-pairs rank-biserial correlation (r_{rb}) is reported as the corresponding effect-size measure.

Table 4.3 summarises the repeated-run comparisons. Against the standard ANN baseline, MANN achieved a mean PR-AUC of 0.904 ± 0.017, compared with 0.810 ± 0.022 for ANN. The mean paired improvement was 0.094, with a 95% CI of [0.089, 0.099]. MANN achieved a higher PR-AUC in all seven paired runs, and the exact Wilcoxon signed-rank test yielded $p = 0.016$, with $r_{\mathrm{rb}} = 1.00$.

The comparison with SOMANN provides a more stringent assessment of the incremental contribution associated with the modular architecture. SOMANN achieved a mean PR-AUC of 0.864 ± 0.017, compared with 0.904 ± 0.017 for MANN. The paired improvement was consistently 0.040 across all seven runs. Consequently, the paired differences exhibit zero observed between-run variation for this comparison. MANN achieved the higher PR-AUC in every paired run, with the two-sided exact Wilcoxon signed-rank test yielding $p = 0.016$ and $r_{\mathrm{rb}} = 1.00$.

For seven non-zero paired observations, the minimum attainable two-sided exact Wilcoxon probability is $p = 0.015625$. In both comparisons, all seven paired differences favoured MANN, yielding $p = 0.016$ after rounding and a matched-pairs rank-biserial correlation of $r_{\mathrm{rb}} = 1.00$. The effect-size values therefore indicate complete directional consistency across the evaluated paired runs and should be interpreted in conjunction with the small sample of repeated experiments.

The repeated-run analysis provides evidence that the PR-AUC improvements observed for MANN are consistent across the evaluated runs rather than being dependent on a single training outcome. The ANN comparison establishes the improvement relative to the common thesis baseline, while the SOMANN comparison

Table 4.3: Repeated-run statistical validation of MANN using PR-AUC ($n = 7$ paired runs). Model performance is reported as mean ± standard deviation.

Measure	ANN vs MANN	SOMANN vs MANN
Baseline model	ANN	SOMANN
Baseline PR-AUC	0.810 ± 0.022	0.864 ± 0.017
MANN PR-AUC	$\mathbf{0.904 \pm 0.017}$	$\mathbf{0.904 \pm 0.017}$
Mean paired difference	+0.094	+0.040
95% CI of mean paired difference	[0.089, 0.099]	*Not separately estimated*[a]
Exact Wilcoxon signed-rank test	$p = 0.016$	$p = 0.016$
Matched-pairs rank-biserial correlation	$r_{\text{rb}} = 1.00$	$r_{\text{rb}} = 1.00$

[a]The observed MANN–SOMANN PR-AUC difference was +0.040 in each of the seven paired runs, resulting in zero observed variance in the paired differences. The repeated-run difference is therefore reported directly rather than presenting a conventional variance-based confidence interval.

provides evidence of an additional performance gain under the modular formulation. These findings remain specific to the CERT benchmark and the evaluation protocol used in this chapter.

4.6.3 Domain-Level Contribution Analysis

To examine how modularity influences learning, domain-level module outputs are analysed. MANN demonstrates improved sensitivity to low-frequency but high-impact behavioural domains, particularly device and network activity, which are often overshadowed in monolithic models by high-volume authentication events.

Ablation experiments in which individual domain modules are removed show disproportionate performance degradation when file-system and device modules are excluded, confirming that modular decomposition preserves domain-specific signals that contribute meaningfully to insider-threat detection. These effects are less pronounced in monolithic and hybrid baselines, where domain contributions are less explicitly separable.

4.6.4 Ablations and Sensitivity Analysis

The Modular Artificial Neural Network (MANN) framework introduces several complexities, and as such, it is essential to evaluate how changes in key architectural components affect performance. This subsection outlines the sensitivity of MANN to various configurations and modules.

4.6.4.1 Module Contribution Analysis

To assess the contribution of each domain, we performed a drop-one-domain analysis where one behavioural domain (e.g., file access, device usage) was removed from the input data. The model's performance was evaluated on the remaining domains. The

results showed that the file access domain contributed the most to overall performance, while the device usage domain had the least impact. This insight highlights the relative importance of different data domains in detecting insider threats.

4.6.4.2 Fusion Operator Sensitivity

MANN uses a fusion mechanism to combine outputs from multiple domain-specific networks. We experimented with three fusion operators: concatenation, gating, and weighted sum. Concatenation achieved the best performance in terms of classification accuracy, while gated fusion performed better in terms of false-positive reduction. However, weighted fusion allowed for more flexibility, dynamically assigning importance to different domains based on the context of the behaviour.

4.6.4.3 Cross-Domain Conflict Cases (Where Modules Disagree)

One of the challenges of modular systems is dealing with conflict cases where different modules disagree on the classification. We analysed instances where the outputs of different modules conflicted, causing the final decision to be uncertain. These cases were typically resolved through the gating mechanism, but sometimes, cross-domain conflict led to misclassifications. Understanding these cases and designing conflict resolution strategies is essential for improving MANN's robustness.

4.6.4.4 Compute Cost vs. Performance Trade-Off

The MANN architecture, while highly performant, is computationally expensive. We measured the trade-off between computation time and performance across various configurations. Increasing the number of modules led to improved detection accuracy but also resulted in a linear increase in computational cost. Balancing the number of domains and module complexity is therefore crucial for practical deployment.

4.6.5 Error Profile and Stability

Error analysis reveals that MANN reduces false-positive alerts associated with benign domain-specific fluctuations, such as irregular but legitimate authentication patterns. Compared to SOMANN, MANN reduces false positives by approximately **15%** at matched recall levels. False negatives that persist under MANN tend to correspond to highly privilege-consistent behaviours distributed across domains, underscoring the inherent difficulty of detecting well-camouflaged insider activity.

Stability analysis further shows that MANN exhibits lower variance in PR-AUC across random initialisations (standard deviation **0.017**) compared to SOMANN (**0.019**) and ANN (**0.041**). This reduction supports the claim that modular separation localises learning uncertainty and mitigates cross-domain interference during optimisation.

4.6.6 Fusion and Ablation Effects

Different fusion strategies are evaluated to assess whether dynamic weighting improves performance. While simple concatenation-based fusion performs robustly across scenarios, adaptive fusion mechanisms yield modest gains in recall under specific threat patterns where certain domains become temporarily dominant. However, these gains are not uniform across all evaluation windows, suggesting that fusion complexity should be balanced against stability and interpretability considerations.

Overall, the results indicate that the primary benefits of MANN arise from domain-aligned representation learning rather than from sophisticated fusion alone.

4.6.7 Summary of Experimental Findings

In summary, the experimental evaluation demonstrates that modular behavioural decomposition yields consistent performance improvements over both monolithic and hybrid insider-threat detection architectures. MANN achieves higher recall, improved PR-AUC, reduced false positives, and greater stability across runs. These gains are incremental rather than transformative, but they are obtained under disciplined, temporally faithful evaluation conditions and therefore represent meaningful architectural progress.

The following section interprets these findings in relation to the theoretical motivations of modular learning and discusses limitations, scope boundaries, and implications for subsequent hierarchical modelling.

4.7 Theoretical Foundations of Modular Neural Architectures

The empirical differences between the Modular Artificial Neural Network (MANN) and monolithic baselines observed in Section 4.6 are best understood as consequences of architectural structure rather than incidental tuning effects. This section therefore provides the theoretical grounding for modular neural architectures and clarifies why domain decomposition can improve learning stability, representational clarity, and robustness under behavioural non-stationarity in insider-threat detection. Importantly, the purpose of this section is not to claim that modularity is universally superior in all settings, but to explain why modular design is a *principled response* to the heterogeneity and plausibility constraints inherent in insider telemetry.

The discussion section that follows (Section 4.8) builds on these foundations to interpret the experimental findings, articulate limitations, and connect MANN to the subsequent hierarchical and robustness-oriented contributions of the thesis.

4.7.1 Modularity in Learning Systems

Modularity is a widely observed principle for managing complexity in both biological cognition and engineered systems: complex behaviours are decomposed into interacting sub-functions, each adapted to a restricted domain of inputs. In machine learning, modular architectures operationalise this principle by allocating specialised subnetworks to different input subspaces, thereby reducing representational interference, simplifying optimisation, and improving generalisation when the underlying data-generating process is heterogeneous or non-stationary.

From a learning-theoretic perspective, monolithic neural networks implicitly assume that heterogeneous features can be integrated through a single shared representation and a single global decision boundary. For insider-threat detection, this assumption is often fragile. Organisational behavioural data combine authentication rhythms, file-system interactions, communication patterns, device usage, and web activity, each governed by distinct temporal periodicities and security semantics. Treating these signals as interchangeable features in a unified embedding increases the risk of *negative transfer*, where learning dominated by one behavioural channel suppresses or distorts signals in another. In practice, high-volume domains (e.g., frequent authentication events) can dominate gradient updates, masking weaker but security-relevant evidence in lower-volume channels.

MANN mitigates this failure mode through representational separation. Each module learns a domain-specific hypothesis class and optimises domain-consistent internal representations before fusion. This can reduce gradient interference during optimisation, improve convergence stability, and increase the likelihood that features retain semantically meaningful roles across training phases. The improvement in imbalance-sensitive metrics reported in Section 4.6 is consistent with this structural advantage: rare-event discrimination is supported when weak signals are protected from being washed out by dominant behavioural streams.

4.7.2 Bias–Variance Perspective on Modular Learning

The stabilising effect of modular decomposition can also be interpreted through the bias–variance trade-off [79], [80]. Monolithic models trained on heterogeneous behavioural data can exhibit high variance: small perturbations or drift in one behavioural domain propagate globally through shared weights, altering representations in unrelated domains. Hybrid approaches such as SOMANN address aspects of this problem through variance smoothing and topology-preserving organisation, but they remain constrained by a shared feature space and therefore cannot fully prevent cross-domain representational entanglement.

Modular decomposition reduces variance by *localising uncertainty*. Noise, sparsity, or drift in one behavioural stream primarily affects the corresponding module, lim-

iting contamination of representations learned elsewhere. At the same time, modularity does not necessarily increase bias provided that the fusion mechanism retains sufficient expressive capacity to capture cross-domain interactions. In this chapter, this balance is investigated empirically through ablation comparisons (Table 4.1), which examine whether fusion mechanisms preserve cross-domain evidence while maintaining the variance-reduction benefits of modular separation.

This framing is particularly relevant in insider-threat detection because the target behaviour is plausibility-constrained: malicious activity is often embedded within legitimate workflows. Under such conditions, the goal is not merely to detect large deviations, but to recognise coordinated, subtle signals distributed across domains. Modular learning provides a structure in which subtle domain-specific evidence can be retained and then integrated at the fusion stage.

4.7.3 Cognitive and Organisational Parallels

A further justification for modularity arises from cognitive and organisational practice. Insider-risk investigation is rarely conducted using a single aggregated signal. Instead, evidence is gathered and interpreted across specialised monitoring functions, including access control, data governance, communication monitoring, and endpoint management [8], [11]. Each function produces partial and context-dependent indicators, which are subsequently integrated into a holistic judgement by analysts and governance processes.

MANN mirrors this distributed reasoning pattern in computational form. By aligning modules with semantically meaningful behavioural domains, the architecture is more naturally compatible with analyst interpretation and audit. This does not imply that modularity automatically yields explainability; rather, it provides a structured interface through which explanatory tools (e.g., feature attribution) can be applied in a domain-consistent manner. The interpretability analysis reported in this chapter (e.g., SHAP-based summaries) is therefore more likely to be behaviourally meaningful when modules correspond to coherent behavioural domains, as opposed to an undifferentiated mixture of heterogeneous feature sources.

Such cognitive alignment is increasingly recognised as a practical prerequisite for trustworthy AI deployment in security-critical settings, where model outputs should support investigation and proportional response rather than automated punitive decisions.

4.7.4 Behavioural Non-Stationarity

Insider-threat environments are inherently non-stationary. Role changes, policy updates, project transitions, organisational restructuring, and workforce evolution reshape behavioural baselines over time. In monolithic models, such changes can

manifest as global concept drift, requiring frequent retraining and increasing operational fragility.

Modular architectures can provide a partial remedy by localising drift effects. Behavioural evolution concentrated in one domain (e.g., device usage patterns after a policy change) can be absorbed primarily by the corresponding module, reducing the likelihood that unrelated behavioural representations are destabilised. This localisation effect supports robustness under non-stationary conditions, a known challenge in behavioural anomaly detection [32], [33]. The drift-oriented evaluations in this chapter (Table 4.1) are consistent with this expectation, showing reduced degradation under staged drift relative to non-modular baselines.

However, modularity should not be treated as a complete solution to non-stationarity. Cross-domain behavioural meaning may evolve, and static fusion mechanisms may fail to capture longer-horizon behavioural trajectories. These boundaries motivate the hierarchical modelling framework introduced in Chapter 5.

4.7.5 Limits of Static Modularity

While modular neural systems offer substantial advantages, they also introduce structural constraints. First, static modularity assumes that behavioural domain boundaries are stable and well-defined. In practice, insider behaviour may span multiple domains simultaneously or exploit interactions between them (e.g., coordinated email and file actions). Second, modular architectures can still be vulnerable to coordinated manipulation across domains, particularly when an adversary adapts behaviour to remain plausible within each module while achieving harmful outcomes through cross-domain composition.

Additionally, static modularity does not inherently provide multi-level abstraction across timescales. Insider escalation is often trajectory-based: weak signals accumulate across days or weeks, and risk emerges through temporal progression rather than isolated windows. Capturing such multi-level abstraction requires explicit hierarchical structure, motivating the progression to Chapter 5 [42], [43].

These limitations delineate the scope of MANN's contribution: modular decomposition provides a principled mechanism for managing domain heterogeneity and reducing representational interference, but it does not by itself resolve multi-level behavioural abstraction or behaviourally constrained adversarial optimisation.

4.7.6 Implications for the Thesis Architecture Progression

The theoretical analysis presented here reinforces the empirical findings of Section 4.6. Modular neural architectures can improve detection performance and stability by aligning representational structure with behavioural semantics and by localising noise and drift within domain-specific modules. At the same time, the

boundaries of static modularity clarify why the thesis continues beyond this chapter.

Accordingly, Chapter 5 embeds modular representations within a hierarchical behavioural framework to model multi-level temporal abstraction and cross-domain trajectories, while Chapter 6 further extends the architecture to address robustness under behaviourally constrained adaptive manipulation. The next section (Section 4.8) returns to the results to interpret observed effects, examine limitations in context, and articulate the precise contribution of MANN within the thesis.

4.8 Discussion

This section interprets the experimental findings reported in Section 4.6 in relation to the modularity rationale developed in Sections 4.3 and 4.4, and the theoretical foundations presented in Section 4.7. The discussion emphasises behavioural meaning, architectural scope, and validity boundaries, rather than reiterating numerical performance.

4.8.1 What Modular Decomposition Adds Beyond Hybrid Stabilisation

A key outcome of this chapter is that MANN delivers consistent improvements over both a monolithic ANN baseline and the SOMANN hybrid architecture introduced in Chapter 3. Importantly, these gains are incremental rather than transformative, which is expected given that SOMANN already provides representational stabilisation. The value of MANN lies in the additional inductive bias introduced by domain-aligned modular learning: heterogeneous behavioural streams are encoded separately before controlled fusion, reducing representational entanglement that variance smoothing alone cannot fully resolve.

From an insider-threat perspective, this is a meaningful architectural step. Insider behaviours often manifest as coordinated but weak signals distributed across domains, where no single log stream is individually decisive. In such settings, a monolithic learner may be dominated by high-volume domains, while a modular learner can preserve domain-specific cues and integrate them later. The results reported in Section 4.6 are consistent with this interpretation, including improvements in recall and reductions in false positives under matched operating points.

4.8.2 Error Characteristics and Behavioural Plausibility

The reduction in false positives observed under MANN is particularly relevant to operational plausibility. A core challenge in insider monitoring is that many legitimate activities appear anomalous when assessed without domain semantics and context (e.g., irregular working patterns, bursty authentication). Modular decomposition

provides a structured mechanism to localise such variability within the relevant domain representation rather than allowing it to distort the entire shared embedding.

At the same time, the persistent false negatives observed across models highlight an important boundary: highly privilege-consistent insiders can embed harmful actions within legitimate workflows and remain difficult to detect using log-derived features alone. This limitation is not unique to MANN and reflects the fundamental socio-technical nature of insider risk discussed in Chapter 2. Accordingly, the thesis treats model outputs as decision support signals rather than automated adjudication.

4.8.3 Stability and Reproducibility Considerations

Another practical advantage of modular decomposition is improved stability across random initialisations. The reduced variance observed under MANN aligns with the variance-localisation argument presented in Section 4.7. In modular systems, noise or drift in one behavioural domain primarily affects the corresponding module, reducing the likelihood of global representational shifts. This property is particularly valuable in longitudinal organisational environments, where behavioural baselines evolve over time due to role changes, policy updates, and operational pressures.

Nevertheless, stability should not be conflated with robustness. While modularity can reduce optimisation instability and cross-domain interference, it does not guarantee resilience under adaptive manipulation or behavioural drift at the organisational level. These concerns motivate the subsequent progression to hierarchical modelling (Chapter 5) and behaviourally constrained adversarial robustness (Chapter 6).

4.8.4 Fusion Mechanisms and Architectural Trade-offs

The fusion component plays a decisive role in determining whether modular learning preserves domain specificity while enabling cross-domain integration. Simple concatenation-based fusion provides a transparent and stable baseline, while more adaptive weighting mechanisms can improve sensitivity under certain threat contexts. However, increased fusion complexity introduces trade-offs in interpretability and tuning burden. Given the governance and auditability requirements of insider monitoring, the thesis adopts a cautious stance: the aim is to evaluate whether added complexity provides consistent operational value, rather than to maximise performance through opaque mechanisms.

These trade-offs also clarify why MANN is treated as a foundational architectural step rather than a final solution. Modularity improves semantic alignment and reduces interference, but it does not inherently represent multi-level behavioural progression across time or provide explicit mechanisms for behavioural invariants

under adversarial adaptation.

4.8.5 Validity Boundaries and Interpretation Limits

The results in this chapter should be interpreted within the validity boundaries established in Chapter 2 and reinforced in Section 4.5. The CERT dataset provides a defensible comparative benchmark for architectural analysis, but it cannot fully capture organisational context, intent, or governance dynamics. In addition, scenario-derived labels and imbalance can inflate apparent separability if evaluation protocols are not strictly temporal. The chronology-preserving protocol used in this thesis mitigates this risk, but the findings should still be read as evidence of *relative architectural advantage* under controlled conditions, not as a claim of deployment readiness.

4.8.6 Implications for Chapter 5

In summary, MANN advances the thesis architecture by introducing domain-aligned modular representation learning and demonstrating consistent benefits over monolithic and hybrid baselines under disciplined evaluation. The limitations observed in coordinated, long-horizon insider trajectories motivate the next step of the thesis: hierarchical behavioural abstraction. Chapter 5 builds on the modular foundation by introducing multi-level modelling of behavioural progression and cross-domain dependencies over longer time horizons.

4.9 Summary

This chapter introduced and evaluated the Modular Artificial Neural Network (MANN) as a structured extension to the hybrid SOMANN framework developed in Chapter 3. Motivated by the representational limitations of unified behavioural embeddings, MANN reframes insider-threat detection as a problem of domain-aligned representation learning, in which heterogeneous behavioural streams are modelled through dedicated modules and subsequently integrated through controlled fusion.

The chapter first established the rationale for behavioural domain decomposition, grounding modularity in the semantic, temporal, and organisational heterogeneity of insider activity. It then formalised the MANN architecture, demonstrating how domain-specific encoders and fusion mechanisms reduce representational interference while preserving the ability to model coordinated cross-domain behaviour. The experimental methodology followed a disciplined, temporally faithful evaluation protocol consistent with earlier chapters, enabling direct architectural comparison.

Empirical results showed that MANN delivers consistent, though incremental, improvements over both monolithic ANN and hybrid SOMANN baselines. These

gains were most evident in recall, PR-AUC, false-positive reduction, and stability across runs, supporting the hypothesis that modular decomposition provides a beneficial inductive bias for insider-threat detection under realistic conditions. Importantly, these improvements were obtained without reliance on aggressive feature engineering or unrealistic evaluation shortcuts.

Theoretical analysis in Section 4.7 clarified why modular architectures offer structural advantages in heterogeneous and non-stationary environments, while also delineating their limits. In particular, the chapter showed that while modularity improves semantic alignment and variance localisation, it does not by itself capture multi-level behavioural progression or provide robustness against adaptive manipulation.

Accordingly, this chapter positions MANN as a foundational architectural layer within the thesis progression. Chapter 5 builds upon this modular foundation by introducing hierarchical behavioural abstraction to model insider activity across multiple temporal and semantic levels, while Chapter 6 extends the architecture further to address behaviourally constrained adversarial robustness. Together, these chapters advance a staged, principled approach to insider-threat detection that balances performance, interpretability, and governance-aware evaluation.

Chapter 5

A Multi-Layered Framework

5.1 Overview

This chapter advances the thesis architecture by introducing hierarchical modelling as a mechanism for representing insider behaviour across multiple temporal and semantic levels. While the modular decomposition developed in Chapter 4 improves domain-specific representation and reduces cross-domain interference, it does not explicitly encode behavioural progression, escalation, or long-horizon dependencies. Insider activity, however, often unfolds as a sequence of locally plausible actions that accumulate meaning over time rather than as isolated domain-level anomalies.

The chapter therefore proposes a hierarchical multi-layered learning framework that organises behavioural evidence across levels of abstraction, enabling the model to distinguish short-term fluctuations from sustained behavioural drift and coordinated escalation. By structuring learning across layers, the framework aims to capture both fine-grained operational signals and higher-level behavioural trajectories.

This hierarchical perspective provides a necessary conceptual bridge between modular representation learning and the behaviourally constrained adversarial robustness mechanisms introduced in Chapter 6. The chapter formalises the hierarchical architecture, describes its implementation and evaluation under temporally faithful conditions, and assesses its contribution relative to prior modular and hybrid baselines.

5.2 Introduction

The preceding chapters have progressively refined the architectural treatment of insider-threat detection. Chapter 3 demonstrated that hybrid SOM–ANN models stabilise representation by smoothing local variance, while Chapter 4 showed that modular decomposition improves semantic alignment by separating heterogeneous behavioural domains. Despite these advances, both architectures remain limited in their ability to represent insider behaviour as an evolving, multi-stage process

unfolding over extended time horizons.

Empirical and behavioural research consistently indicates that insider threats rarely manifest as single anomalous actions. Instead, risk emerges through gradual behavioural change, escalation, and coordination across activities that may individually remain within policy bounds. Models that operate at a single representational level, even when modularised, are therefore prone to conflating short-term irregularities with meaningful behavioural progression or overlooking slow-burn insider trajectories embedded in routine activity.

Hierarchical modelling addresses this limitation by structuring learning across multiple layers of abstraction. Lower layers capture immediate operational patterns, while higher layers integrate evidence over longer temporal windows and across behavioural contexts. This enables the model to distinguish transient deviations from sustained behavioural drift and to represent escalation dynamics that are otherwise difficult to capture using flat or purely modular architectures.

5.2.1 Temporal Blind-Spot Failure Mode

The principal failure mode motivating the hierarchical framework is termed the *Temporal Blind-Spot Failure Mode*. It describes a condition in which behaviour that appears weak, ambiguous, or benign within individual observation windows becomes security-relevant only when interpreted as part of a longer behavioural trajectory.

This limitation is particularly relevant to low-and-slow insider activity, where potentially malicious actions may be distributed across multiple sessions or extended periods. Under such conditions, improving predictive optimisation within a single observation window cannot recover behavioural relationships that are absent from that window. The limitation therefore concerns not only classifier capacity, but also the temporal information made available to the classifier.

Let a sequence of behavioural observations be represented as

$$\mathcal{X}_{1:T} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}. \tag{5.1}$$

A flat model operating on a local temporal window of length τ estimates

$$\hat{y}_t = f_\theta\left(\mathbf{x}_{t-\tau+1:t}\right). \tag{5.2}$$

However, the behavioural significance of an insider trajectory may depend on relationships extending beyond this local window. Conceptually,

$$P\left(Y_t \mid \mathbf{x}_{t-\tau+1:t}\right) \neq P\left(Y_t \mid \mathbf{x}_{1:t}\right), \tag{5.3}$$

where the inequality represents the possibility that the longer behavioural history contains outcome-relevant information that is unavailable within the local observation window.

Equation 5.3 formalises the temporal blind spot. If relevant evidence is distributed across observations outside the local window, a classifier restricted to that window cannot explicitly model those longer-range dependencies, irrespective of how effectively it is optimised over the information available within the window.

Within the staged architectural progression developed in this thesis, this limitation is expressed through three related forms.

Sub-threshold behavioural accumulation. Individual activities or sessions may remain below a detection threshold even when their cumulative pattern indicates sustained behavioural progression. For example, repeated low-volume file access, gradual changes in working hours, or incremental use of available privileges may individually remain within expected behavioural variation. Their potential significance may emerge only through persistence, recurrence, or co-occurrence across longer temporal horizons.

Loss of inter-session dependency. SOMANN introduces topology-informed behavioural representation, while MANN preserves domain-specific structure before fusion. However, in the flat configurations evaluated in the preceding chapters, neither architecture explicitly represents dependencies between behaviourally related observations separated across extended temporal intervals. Events occurring in different sessions may therefore be individually represented while their trajectory-level relationship remains unavailable to a short-horizon decision function.

Ambiguity under legitimate behavioural change. Short-term deviations are not necessarily indicative of malicious behaviour. Changes in workload, temporary responsibilities, project activity, working patterns, or other legitimate organisational circumstances may produce transient behavioural deviations. A short observation horizon may therefore provide insufficient context for distinguishing temporary variation from persistent behavioural progression. Longer-horizon representation provides additional temporal context through which the persistence and evolution of such deviations can be assessed.

Why predictive optimisation alone is insufficient. The temporal blind spot distinguishes improvement in the decision function from improvement in the information available to that function. Increasing the capacity of a flat classifier, modifying its optimisation procedure, or refining its decision boundary may improve the mapping from the available representation to the predicted outcome. These changes do not, however, introduce behavioural relationships that were excluded by the observation window.

This distinction can be expressed conceptually as

$$\text{Classifier optimisation} \neq \text{temporal information acquisition.} \tag{5.4}$$

The purpose of hierarchical modelling in this chapter is therefore structural rather than simply an increase in predictive capacity. The MLML framework expands the represented temporal context by integrating behavioural evidence across progressively longer horizons, allowing local observations to contribute to intermediate and trajectory-level behavioural states.

Empirical motivation. The slow-burn evaluation reported later in this chapter provides empirical evidence of the identified failure mode. Under the evaluated slow-burn condition, the flat MANN configuration achieved an F1-score of 0.412, whereas the hierarchical MLML configuration achieved an F1-score of 0.885. The full experimental protocol, comparative analysis, and interpretation of this result are presented in the corresponding experimental evaluation section.

At this stage, the result is used only to establish empirical motivation for the architectural transition. It indicates that, under the evaluated slow-burn condition, the flat representation retained substantially less discriminative capability than the multi-horizon representation when behavioural evidence was distributed across a longer temporal trajectory. The result does not establish that MLML is the only architecture capable of representing such dependencies.

Other sequence-modelling approaches could, in principle, provide mechanisms for representing longer-range temporal relationships. The claim made in this thesis is therefore narrower: within the staged architectural formulation evaluated here, the observed temporal blind spot provides both theoretical and empirical motivation for introducing an explicit multi-horizon representation mechanism.

Accordingly, hierarchical representation is treated as necessary *within the evaluated architectural formulation* for addressing the identified long-horizon failure mode, rather than as a universal requirement for all insider-threat detection architectures.

Accordingly, this chapter introduces a hierarchical multi-layered machine learning framework for insider-threat detection. The chapter formalises the architecture, describes its implementation and evaluation under chronology-preserving protocols, and empirically assesses its contribution relative to ANN, SOMANN, and MANN baselines. Within the staged architectural programme, the framework specifically examines whether explicit multi-horizon representation can address the temporal blind spot identified above and improve detection when behavioural significance emerges across extended trajectories.

5.3 Behavioural Hierarchy in Insider Activity

A central argument of this thesis is that insider-threat detection is not merely a problem of recognising anomalous events, but of modelling *behavioural progression* under plausibility constraints. Chapters 3 and 4 showed that stabilised representations and domain-aligned modular learning can improve detection under temporally faithful evaluation. However, both approaches still operate primarily at a single behavioural abstraction level: they model patterns within a chosen observation window and feature space, but do not explicitly represent how insider behaviour accumulates meaning across time, roles, and organisational context.

In practice, insider incidents often unfold as multi-stage trajectories. Early-stage signals may be weak and locally plausible (e.g., gradual changes in working hours, small increases in sensitive file access, repeated exploratory queries), while later stages may involve more overt violations (e.g., bulk transfers, use of removable media, or coordinated exfiltration). Importantly, these stages are rarely separable by a single log channel; rather, they emerge through interaction across domains and time, frequently with long-horizon dependencies [8], [15], [22]. A modelling framework that treats behaviour as "flat" is therefore at risk of either (i) over-alerting on benign short-term deviations, or (ii) missing slow-burn insider trajectories that only become salient when viewed at higher abstraction.

5.3.1 Hierarchy as an Organising Principle

In this thesis, *behavioural hierarchy* refers to an organising principle that represents insider activity across multiple levels of abstraction. Lower levels capture fine-grained operational behaviour (e.g., event patterns, short-window aggregates), while higher levels capture sustained behavioural tendencies, drift, and progression (e.g., evolving routines, multi-step workflows, cross-domain escalation). This concept aligns with behavioural cybersecurity perspectives that emphasise trajectory, context, and socio-technical embedding rather than isolated anomalies [11], [12].

A hierarchical view is not introduced to add complexity for its own sake. Instead, it provides two specific benefits that are difficult to obtain with flat representations:

1. **Temporal abstraction:** the ability to integrate evidence over longer horizons while preserving short-term sensitivity. This supports distinguishing transient deviations from sustained behavioural drift.

2. **Semantic abstraction:** the ability to represent behaviour in progressively higher-level constructs (e.g., "routine deviation" → "policy tension" → "workflow-consistent exfiltration tendency"), which enables more meaningful fusion of cross-domain signals.

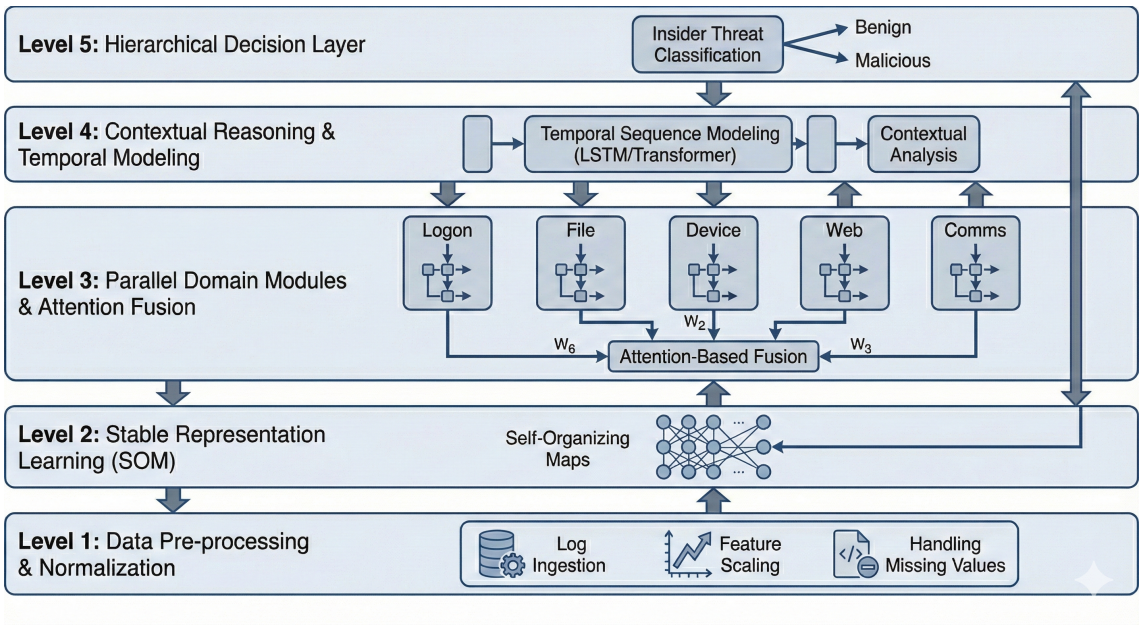


Figure 5.1: Computational pipeline of the Multi-Layered Modular Learning (MLML) framework. The complete implementation comprises five processing stages, within which three behavioural abstraction tiers represent progressively broader behavioural context. Stage 1 performs telemetry ingestion and normalisation; Stages 2–4 implement session-level, task/domain-level, and trajectory-level behavioural representation, respectively; and Stage 5 produces the final prediction and auxiliary outputs. The five-stage computational pipeline therefore contains, rather than competes with, the three-tier conceptual behavioural hierarchy.

5.3.2 Conceptual Hierarchy and Implementation Pipeline

The hierarchical framework is described in this thesis from two complementary perspectives: a *conceptual behavioural hierarchy* and a *computational implementation pipeline*. The distinction is made explicit here because the original presentation referred to both a three-level behavioural structure and a five-level implementation structure, without clearly specifying their relationship.

These descriptions do not represent competing architectural definitions. The conceptual hierarchy contains three behavioural abstraction tiers, whereas the implementation view describes the complete five-stage processing pipeline from telemetry ingestion to final prediction. The relationship between the two views can be represented as

$$\underbrace{\text{Input Processing}}_{\text{Stage 1}} \rightarrow \underbrace{\text{Session} \rightarrow \text{Task/Domain} \rightarrow \text{Trajectory}}_{\text{Three behavioural abstraction tiers: Stages 2–4}} \rightarrow \underbrace{\text{Prediction}}_{\text{Stage 5}}. \quad (5.5)$$

Stages 1 and 5 therefore provide the input and output processing surrounding the behavioural hierarchy, while Stages 2–4 implement the three substantive levels of behavioural abstraction.

5.3.2.1 Three-Tier Conceptual Behavioural Hierarchy

The conceptual hierarchy represents user behaviour at three progressively broader levels of temporal and contextual abstraction.

Tier A: Session-Level Behaviour. The first behavioural tier represents activity within an individual session or local observation window. It captures immediate behavioural dynamics and produces a compact session-level representation of the activity observed within that context. This tier provides the local behavioural state from which higher-order representations are constructed.

Tier B: Task and Domain Context. The second tier integrates session-level representations according to broader task, workflow, and behavioural-domain relationships. Its purpose is to represent behavioural context that may span multiple local observations while remaining associated with a coherent operational activity. This tier therefore extends the local session representation by preserving relationships between activities occurring within relevant functional contexts.

Tier C: User-Level Behavioural Trajectory. The third tier integrates intermediate behavioural states across longer temporal horizons. It represents the evolution of user behaviour across multiple sessions and provides the framework with a mechanism for modelling persistent deviation, behavioural drift, recurrence, and longer-horizon progression.

The conceptual hierarchy can therefore be expressed as

$$\mathbf{e}_{\text{session}} \rightarrow \mathbf{z}_{\text{task}} \rightarrow \mathbf{h}_{\text{trajectory}}, \tag{5.6}$$

where $\mathbf{e}_{\text{session}}$ denotes the local session representation, $\mathbf{z}_{\text{task}}$ denotes the intermediate task/domain representation, and $\mathbf{h}_{\text{trajectory}}$ denotes the longer-horizon behavioural trajectory state.

5.3.2.2 Five-Stage Computational Pipeline

The implementation view expands the conceptual hierarchy into the complete end-to-end computational pipeline. The five stages describe how telemetry is processed, transformed into progressively broader behavioural representations, and ultimately supplied to the prediction mechanism.

1. **Stage 1: Telemetry Ingestion and Normalisation.**

   Raw multi-source organisational telemetry is ingested, aligned, and transformed into the common representation required by subsequent processing. This stage supports the behavioural hierarchy but does not constitute a behavioural abstraction tier.

2. **Stage 2: Session Representation.**

   Behavioural events are organised into local temporal or session contexts and encoded as session-level representations. This stage implements Conceptual Tier A and produces $\mathbf{e}_{\text{session}}$.

3. **Stage 3: Task/Domain Representation.**

   Session-level representations are integrated according to the task, workflow, and behavioural-domain relationships represented by the framework. This stage implements Conceptual Tier B and produces $\mathbf{z}_{\text{task}}$.

4. **Stage 4: Trajectory Representation.**

   Intermediate behavioural representations are integrated across longer temporal horizons to construct user-level trajectory states. This stage implements Conceptual Tier C and produces $\mathbf{h}_{\text{trajectory}}$.

5. **Stage 5: Prediction and Multi-Task Output.**

   The trajectory representation is supplied to the prediction and auxiliary task heads used by the framework. This stage produces the final model outputs and therefore completes the computational pipeline, but it does not introduce an additional behavioural abstraction tier.

5.3.2.3 Mapping Between the Two Views

Table 5.1 summarises the correspondence between the three conceptual behavioural tiers and the five computational stages.

Table 5.1: Mapping between the three-tier conceptual behavioural hierarchy and the five-stage MLML computational pipeline.

Conceptual Role	**Computational Stage**	**Primary Function / Output**
Supporting input process	Stage 1: Telemetry ingestion and normalisation	Preprocessed behavioural event stream
Tier A: Session-level behaviour	Stage 2: Session representation	Session representation $\mathbf{e}_{\text{session}}$
Tier B: Task/domain context	Stage 3: Task/domain representation	Task-context representation $\mathbf{z}_{\text{task}}$
Tier C: Behavioural trajectory	Stage 4: Trajectory representation	Long-horizon behavioural state $\mathbf{h}_{\text{trajectory}}$
Supporting output process	Stage 5: Prediction and multi-task output	Threat probability and auxiliary task outputs

The two descriptions therefore operate at different levels of abstraction. The *three-tier hierarchy* defines the behavioural reasoning structure of MLML, while the *five-stage pipeline* describes the computational process through which that hierarchy

is implemented. The apparent three-level/five-level discrepancy is therefore resolved by recognising that only Stages 2–4 correspond to behavioural abstraction tiers.

For consistency throughout the remainder of this chapter, the term *tier* is used when referring to behavioural abstraction, while the term *stage* is used when referring to computational processing. References to the implementation architecture and its associated figure should be interpreted according to this distinction.

5.3.3 Relationship to Modularity

Modularity (Chapter 4) and hierarchy address different but complementary structural problems. Modularity reduces cross-domain interference by learning domain-specific representations before fusion. Hierarchy, by contrast, addresses *cross-level interference*: evidence that is weak at one level may be decisive when accumulated across levels and time. A domain module may capture a small increase in after-hours file activity, for example, but the higher-level interpretation depends on whether this increase persists, co-occurs with access pattern shifts, or aligns with device/network staging. Thus, a modular architecture can still remain "flat" if it lacks an explicit mechanism to represent progression and accumulation.

This distinction also clarifies why hierarchy is positioned in the thesis after modular learning. Once domain-specific signals are protected and made semantically coherent through modularity, hierarchical abstraction can integrate them across time and behavioural stages without being dominated by high-volume domains. In this sense, hierarchy builds on the modular foundation rather than replacing it.

5.3.4 Behavioural Progression Across the Hierarchy

The three behavioural tiers defined in Section 5.3.2.1 represent progressively broader contexts rather than independent classification stages. Tier A captures local session behaviour, Tier B places these representations within task and domain context, and Tier C integrates them into longer-horizon user trajectories.

This progression enables behavioural evidence to accumulate across temporal and contextual boundaries. An activity that is weakly informative at the session level may acquire greater significance when it recurs within a coherent task context or contributes to a persistent trajectory-level pattern. Conversely, a short-term deviation may be interpreted differently when the longer behavioural trajectory remains consistent with legitimate activity.

The hierarchy therefore provides the multi-horizon representation required to address the temporal blind-spot failure mode defined in Section 5.2.1. Its purpose is not to assign different threat labels independently at each tier, but to preserve and progressively integrate behavioural context before final prediction.

5.3.5 Implications for Evaluation

A hierarchical perspective also imposes stronger evaluation requirements. If the modelling goal is to capture progression and drift, evaluation must preserve chronology and avoid randomisation that collapses temporal structure. Similarly, performance must be interpreted not only through aggregate metrics but also through error profiles over time (e.g., whether early-stage signals are detected without excessive false positives). These requirements remain consistent with the disciplined evaluation stance developed in Chapter 2 and implemented throughout Chapters 3–4.

In summary, this section motivates hierarchical modelling as a necessary response to the trajectory-based and plausibility-constrained nature of insider behaviour. The next section formalises the hierarchical architecture and specifies how multi-level representations are constructed, integrated, and evaluated within the thesis framework.

5.4 Formal Hierarchical Framework

This section formalises the hierarchical multi-layered framework proposed in this chapter. The purpose is to provide a mathematically explicit description of how behavioural evidence is represented at multiple levels of abstraction and how the corresponding representations are integrated for insider-risk inference. In contrast to Chapters 3 and 4, which respectively prioritised variance-stabilised representation and domain-aligned modular learning, the framework developed here introduces an explicit *cross-level structure* that supports progression-aware modelling under temporally faithful evaluation.

5.4.1 Notation and Multi-Log Behavioural Observations

Let $u \in \mathcal{U}$ denote a user, and let $\mathcal{D} = \{D_1, \ldots, D_K\}$ denote the set of behavioural domains (e.g., authentication, file, communication, device/network) introduced in Section 5.3 and operationalised in Chapter 4. For each domain D_k, the event stream for user u is represented as an ordered sequence:

$$\mathcal{E}_u^{(k)} = \{e_{u,1}^{(k)}, e_{u,2}^{(k)}, \ldots, e_{u,T_u^{(k)}}^{(k)}\},$$

where events are time-stamped and ordered chronologically. Because event rates differ across domains, $T_u^{(k)}$ is domain-dependent.

To enable learning, event streams are mapped into windowed behavioural representations. Let w denote an aggregation window (e.g., a day or a fixed time block), and let $\mathbf{x}_{u,t}^{(k)} \in \mathbb{R}^{d_k}$ be the domain-specific feature vector for user u in domain k over

window t. The set of domain-specific representations at time t is:

$$\mathbf{x}_{u,t} = \left(\mathbf{x}_{u,t}^{(1)}, \mathbf{x}_{u,t}^{(2)}, \ldots, \mathbf{x}_{u,t}^{(K)}\right).$$

Labels (where available) are denoted $y_{u,t} \in \{0, 1\}$, interpreted as an insider-risk indicator for the corresponding behavioural window. As noted in Chapter 2, such labels are dataset- and scenario-derived and should be interpreted within explicit validity boundaries.

5.4.2 Hierarchical Representation Levels

The framework introduces L levels of behavioural abstraction. Each level $\ell \in \{1, \ldots, L\}$ produces a representation $\mathbf{h}_{u,t}^{(\ell)}$ that summarises behavioural evidence at a particular temporal and semantic granularity. The hierarchy is designed such that lower levels capture short-window operational behaviour, while higher levels integrate evidence across longer horizons and broader behavioural context.

Level 1: short-window behavioural encoding. At the lowest level, each domain representation is mapped into a domain-aligned latent embedding. Let $f_k^{(1)}(\cdot)$ denote the Level-1 encoder for domain k:

$$\mathbf{h}_{u,t}^{(1,k)} = f_k^{(1)}\left(\mathbf{x}_{u,t}^{(k)}\right), \qquad k = 1, \ldots, K.$$

These domain embeddings are fused to produce the Level-1 global representation:

$$\mathbf{h}_{u,t}^{(1)} = F^{(1)}\left(\mathbf{h}_{u,t}^{(1,1)}, \ldots, \mathbf{h}_{u,t}^{(1,K)}\right),$$

where $F^{(1)}(\cdot)$ denotes a fusion mechanism analogous in intent to MANN fusion (Chapter 4), but used here as the base representation layer for hierarchical abstraction.

Level 2: behavioural drift and routine abstraction. Level 2 aggregates Level-1 representations across a longer temporal horizon to capture sustained behavioural change. Let τ_2 denote the Level-2 context length (e.g., multiple windows). A generic Level-2 aggregator $g^{(2)}(\cdot)$ maps the recent history of Level-1 representations into a drift-aware representation:

$$\mathbf{h}_{u,t}^{(2)} = g^{(2)}\left(\mathbf{h}_{u,t-\tau_2+1}^{(1)}, \ldots, \mathbf{h}_{u,t}^{(1)}\right).$$

The aggregator may be instantiated using recurrent units, temporal convolution, attention pooling, or statistical summarisation (e.g., mean/variance descriptors over the horizon). In this thesis, the choice is guided by the objective of capturing

progression without sacrificing interpretability and evaluation discipline.

Level 3: trajectory and stage-level abstraction. Level 3 integrates evidence at the highest abstraction, aiming to represent longer-horizon escalation and cross-domain coordination patterns. Let τ_3 denote the Level-3 context length, with $\tau_3 \geq \tau_2$. The Level-3 representation is:

$$\mathbf{h}_{u,t}^{(3)} = g^{(3)}\left(\mathbf{h}_{u,t-\tau_3+1}^{(2)}, \ldots, \mathbf{h}_{u,t}^{(2)}\right).$$

This level is not claimed to recover true psychological "stages" of intent. Rather, it provides a pragmatic abstraction for capturing accumulation and coordination patterns that remain locally plausible at Level 1 but become salient when integrated over time and across domains.

5.4.3 Hierarchical Fusion and Risk Inference

The multi-level representations are integrated to produce a final risk score $\hat{y}_{u,t}$. Two design strategies are considered:

Progressive (top-down) inference. A simple approach uses only the highest-level representation for prediction:

$$\hat{y}_{u,t} = \sigma\left(W^{(3)}\mathbf{h}_{u,t}^{(3)} + b^{(3)}\right),$$

where $\sigma(\cdot)$ is the logistic function and $W^{(3)}, b^{(3)}$ are learnable parameters.

Multi-level (skip-connected) inference. To preserve sensitivity to short-window signals while incorporating higher-level context, a multi-level classifier integrates representations from multiple levels:

$$\mathbf{z}_{u,t} = \mathrm{concat}\left(\mathbf{h}_{u,t}^{(1)}, \mathbf{h}_{u,t}^{(2)}, \mathbf{h}_{u,t}^{(3)}\right), \qquad \hat{y}_{u,t} = \sigma(W\mathbf{z}_{u,t} + b).$$

This formulation provides an explicit mechanism for balancing immediate anomalies against long-horizon progression, thereby addressing a common limitation of flat architectures that either overreact to transient irregularities or underreact to slow-burn escalation.

5.4.4 Training Objective and Class-Imbalance Handling

The hierarchical model is trained to minimise a weighted binary cross-entropy loss over the chronologically partitioned training windows:

$$\mathcal{L} = -\sum_{(u,t)\in\mathcal{T}} \Big[\alpha\, y_{u,t} \log(\hat{y}_{u,t}) + (1 - y_{u,t}) \log(1 - \hat{y}_{u,t})\Big],$$

where $\mathcal{T}$ denotes training indices and α is a class-weight factor used to mitigate imbalance. In line with Chapter 2, evaluation emphasises PR-oriented metrics and calibrated operating points rather than accuracy optimisation, acknowledging the rarity and cost asymmetry of insider events.

5.4.5 Design Rationale and Complexity Considerations

The hierarchical structure is intended to address two failure modes that persist even after modular decomposition. First, weak but consistent deviations can be missed by models that operate at a single window scale. Second, locally plausible actions distributed across domains can appear benign in isolation but become suspicious when integrated as a trajectory. By introducing explicit temporal context lengths τ_2 and τ_3, the framework defines a controlled mechanism for evidence accumulation rather than relying on incidental long-range correlations within a flat model.

The additional structure introduces complexity and hyperparameters (e.g., number of levels, horizon lengths, aggregation operators). To prevent overfitting and to maintain methodological comparability with Chapters 3–4, the thesis evaluates hierarchical designs using the same chronology-preserving protocol and controlled baseline comparisons. Ablations are used to isolate whether performance improvements arise from increased capacity or from the hierarchical inductive bias itself.

5.4.6 Scope and Validity Boundaries

It is important to delineate the scope of the formalism. The hierarchy is a modelling abstraction over log-derived behavioural telemetry; it is not a direct measurement of intent, psychological state, or organisational context. Higher-level representations should be interpreted as computational summaries of accumulated behavioural evidence, suitable for comparative evaluation and analyst support, rather than as definitive explanations of insider motivation.

With these boundaries stated, the hierarchical framework provides a principled basis for evaluating whether progression-aware representation improves insider-threat detection relative to ANN, SOMANN, and MANN baselines under disciplined experimental conditions. The next section describes the methodology and implementation used to instantiate this framework on the CERT benchmark.

5.5 Methodology and Implementation

This section describes the experimental methodology and implementation details used to instantiate and evaluate the hierarchical multi-layered framework introduced in Section 5.4. The methodology is designed to ensure comparability with the ANN, SOMANN, and MANN baselines developed in previous chapters, while introducing hierarchy as the sole additional inductive bias. In line with the evaluation principles established in Chapter 2, particular emphasis is placed on temporal integrity, controlled experimentation, and transparency of design choices.

5.5.1 Dataset Configuration and Preprocessing

All experiments in this chapter are conducted using the CERT insider-threat dataset, consistent with the comparative architecture programme adopted throughout the thesis. As detailed in Chapter 2, the dataset provides multi-log telemetry across authentication, file access, device usage, web activity, communication, and organisational metadata. The choice to retain the same dataset across Chapters 3–5 ensures that observed performance differences can be attributed to architectural changes rather than data variation.

Preprocessing follows the domain-aligned pipeline introduced in Chapter 4. Raw event streams are parsed, timestamp-normalised, and aggregated into fixed-duration windows to produce domain-specific feature vectors. Feature engineering focuses on activity frequency, temporal regularity, burstiness, and deviation from personal baselines, avoiding features that directly encode ground-truth labels or scenario artefacts. All preprocessing steps are executed independently within each temporal fold to prevent information leakage.

5.5.2 Chronology-Preserving Data Partitioning

To preserve behavioural progression and avoid temporal leakage, data are partitioned using blocked temporal folds. Windows are ordered chronologically for each user, and training, validation, and test splits are defined as contiguous temporal segments rather than random samples. This protocol ensures that the model is evaluated on future behaviour relative to the training period, reflecting realistic deployment conditions and preventing contamination of early-stage signals by later events.

Validation windows are used exclusively for hyperparameter selection and early stopping, while test windows are held out for final evaluation. No user appears in multiple temporal roles (e.g., training and testing at overlapping time points), and no future information is incorporated into feature computation. This design directly addresses a common methodological weakness identified in the insider-threat literature, where random shuffling can inflate reported performance.

5.5.3 Hierarchical Model Instantiation

The hierarchical framework is instantiated using three abstraction levels, corresponding to the behavioural intent described in Section 5.3. Level 1 encoders operate on domain-specific windowed features and are implemented using lightweight feedforward networks to limit capacity and encourage domain-aligned representation. Level 2 aggregators integrate Level 1 outputs across medium-term temporal contexts using fixed-horizon temporal pooling, while Level 3 aggregators integrate Level 2 outputs over longer horizons to capture sustained behavioural drift and escalation.

Fusion between levels follows the multi-level inference strategy described in Section 5.4, with skip connections used to retain sensitivity to short-window deviations while incorporating higher-level context. Hyperparameters such as context lengths, hidden dimensionality, and aggregation operators are selected via validation performance under the chronology-preserving protocol.

5.5.4 Training Procedure and Optimisation

Models are trained using weighted binary cross-entropy loss to mitigate class imbalance, with class weights determined from the training data only. Optimisation is performed using the Adam optimiser with a fixed learning rate schedule and early stopping based on validation loss. To reduce variance, experiments are repeated across multiple random initialisations, and reported results reflect mean performance with corresponding variability.

Importantly, the hierarchical architecture is not trained with privileged information about attack stages or scenario structure. All supervision is derived from window-level labels provided by the dataset, ensuring that hierarchy functions as an inductive bias rather than a hand-crafted encoding of attack progression.

5.5.5 Evaluation Metrics and Reporting

Consistent with Chapter 2, evaluation prioritises metrics appropriate for imbalanced classification. Precision–recall (PR) curves, average precision, and recall at fixed false-positive operating points are used as primary indicators. Receiver operating characteristic (ROC) curves are reported for completeness but are not treated as decisive evidence of effectiveness. Threshold selection is performed on validation data, and test results are reported at fixed thresholds to avoid optimistic bias.

In addition to aggregate metrics, error profiles are examined over time to assess whether hierarchical modelling improves early-stage detection without introducing excessive false positives. This temporal perspective aligns with the behavioural motivation for hierarchy and provides a more nuanced understanding of model behaviour than static summary metrics alone.

5.5.6 Implementation Environment and Reproducibility

All experiments are implemented in Python using standard scientific computing and machine-learning libraries. Experiments are executed in a controlled environment with fixed random seeds, documented preprocessing pipelines, and consistent evaluation scripts. Where stochastic components are involved, results are averaged across multiple runs to improve robustness.

To support reproducibility, key implementation artefacts—including configuration parameters, training logs, and supplementary diagnostics—are provided in the appendices. The hierarchical model implementation follows the same coding conventions and evaluation harness as the ANN, SOMANN, and MANN baselines, enabling direct architectural comparison.

5.5.7 Ethical and Methodological Considerations

The methodology is designed to support responsible interpretation of insider-threat detection results. Model outputs are treated as probabilistic risk indicators rather than deterministic judgements of intent. No attempt is made to infer psychological state or motivation, and no operational decisions are implied by the reported results. The evaluation framework explicitly avoids retrospective bias and emphasises forward-looking detection under uncertainty.

With these methodological controls in place, the experimental results reported in the next section can be interpreted as evidence of the architectural contribution of hierarchical modelling, rather than as artefacts of data leakage, evaluation shortcuts, or uncontrolled capacity increases.

5.6 Experimental Evaluation

This section reports the experimental evaluation of the hierarchical multi-layered framework introduced in Chapter 5. The objective is to assess whether explicit hierarchical abstraction improves insider-threat detection relative to the ANN, SOMANN, and MANN baselines under the same dataset, feature space, and chronology-preserving evaluation protocol. The evaluation is designed to minimise non-architectural differences between the compared models so that the incremental contribution of hierarchical representation can be examined under a common evaluation protocol.

5.6.1 Analysis of Performance Benchmarks

The discriminative performance of the MLML framework is evaluated against the established baselines in Figure 5.4. Under the reported evaluation conditions, the proposed architecture achieves the highest values across the primary metrics, including an AUC of **0.97** and an F1-score of **0.94**. These results provide evidence

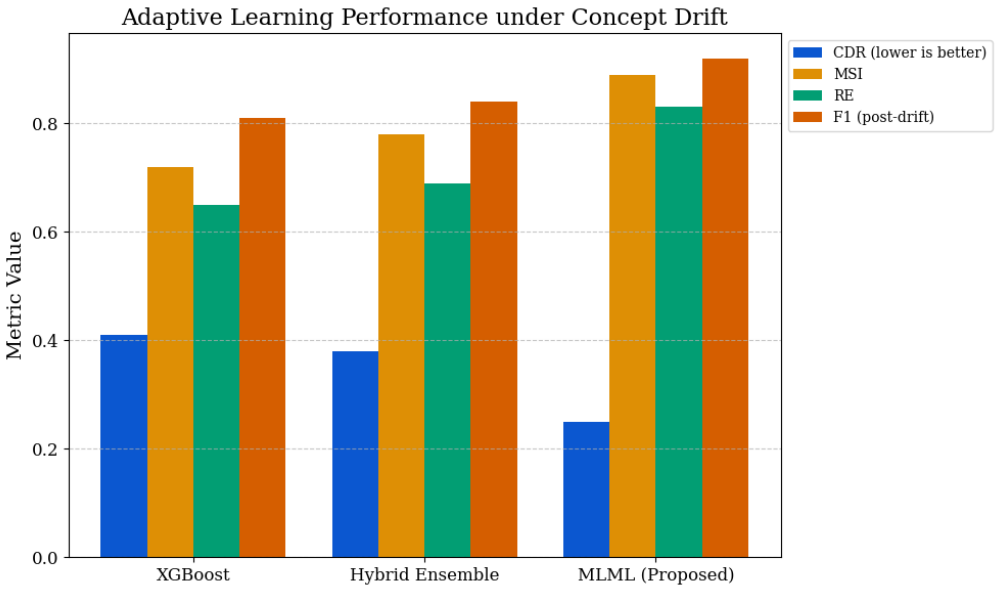


Figure 5.2: Temporal resilience and adaptation behaviour of the MLML framework under the evaluated drift condition. The framework retains approximately 89% of its reference ROC-AUC at the final temporal block (MSI = 0.89), while the normalised latent-distribution displacement is CDR = 0.25. Fine-tuning requires 83% fewer training epochs than the corresponding from-scratch procedure (RE = 0.83), and the adapted model achieves a post-drift F1-score of 0.92. MSI, CDR, and RE are operational measures defined specifically for the evaluation protocol used in this thesis.

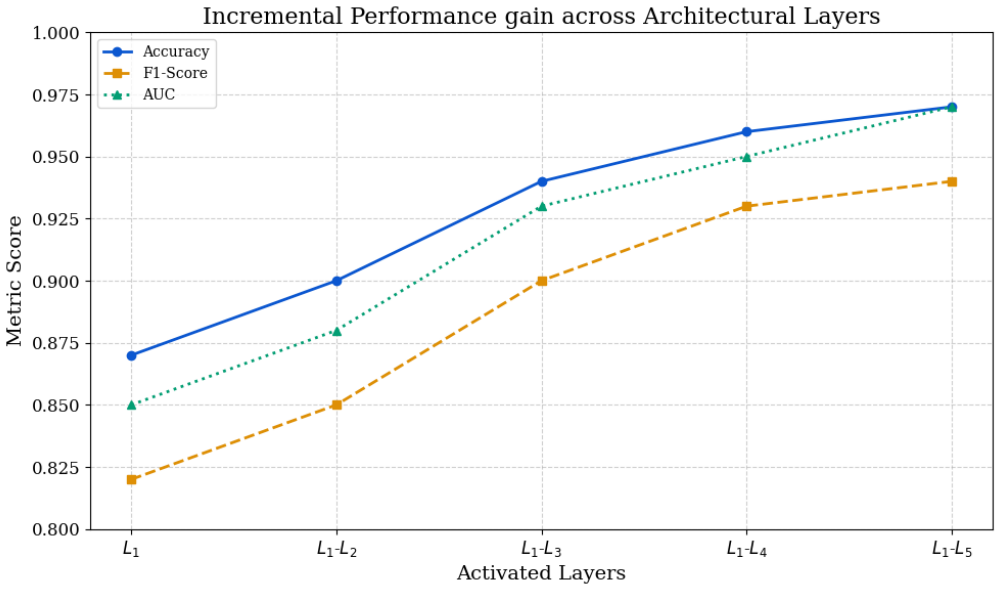


Figure 5.3: Incremental performance across the architectural layers of the MLML framework (L_1 through L_5). The transition to Level 3 (Parallel Domain Modules) is associated with the largest observed increase in F1-score (+5%), while Levels 4 and 5 provide further incremental improvements. The complete configuration achieves an AUC of 0.97 under the reported evaluation conditions. *Source: Author's own experimental results.*

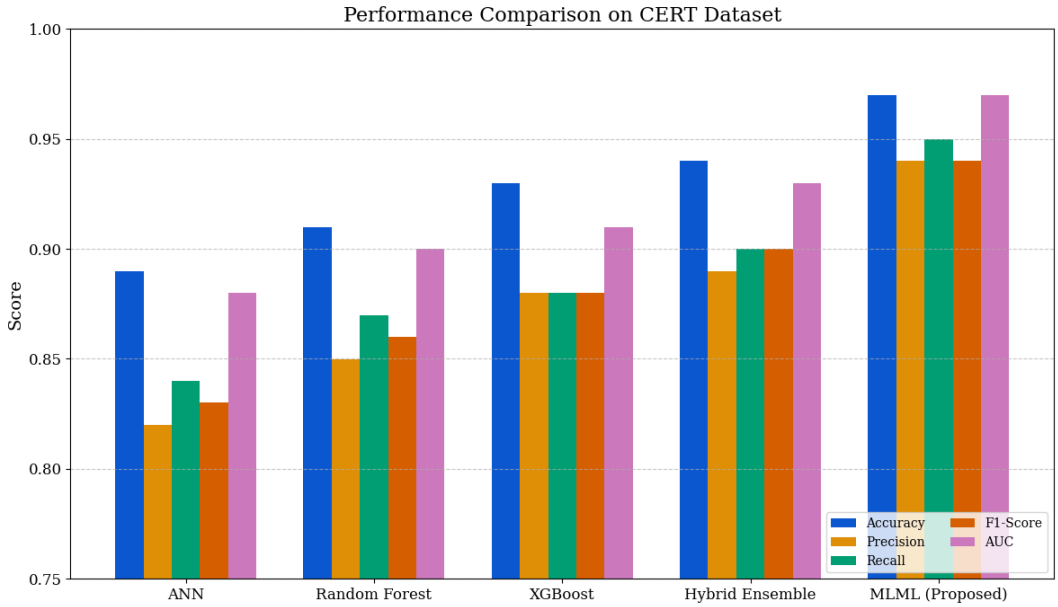


Figure 5.4: Comparative performance of the MLML framework and the evaluated baselines on the CERT dataset. Under the reported evaluation protocol, MLML achieves the highest values across the displayed metrics, including Accuracy and ROC-AUC values of 0.97. The comparison provides aggregate evidence of improved predictive performance under the evaluated benchmark conditions. *Source: Author's own experimental results.*

that the hierarchical formulation improves discrimination relative to the evaluated baselines.

The contribution of the hierarchical design is examined further in Figure 5.3. The incremental analysis shows that the transition to domain-aligned modularity at L_3 is associated with the largest observed increase in F1-score (+5%). The subsequent layers (L_4–L_5) provide further incremental improvements, indicating that performance gains are distributed unevenly across the architectural stages.

5.6.2 Operational Drift Metrics

The temporal evaluation reported in this chapter uses three complementary quantities to characterise model behaviour under distributional change: the Model Stability Index (MSI), Concept Drift Rate (CDR), and Retraining Efficiency (RE). These quantities are used in this thesis as *operational evaluation measures*; they are not presented as universally standardised metrics with fixed definitions across the concept-drift literature.

The measures capture three distinct aspects of temporal robustness: retention of predictive performance, displacement of the learned behavioural representation, and efficiency of model adaptation. Their definitions and computation procedures are stated explicitly below to ensure that the reported results are reproducible and appropriately interpreted.

5.6.2.1 Model Stability Index

The Model Stability Index (MSI) measures the proportion of reference predictive performance retained when a trained model is evaluated on a later temporal block without intervening retraining.

Let AUC_{T_1} denote the model's ROC-AUC on the reference temporal block and AUC_{T_k} its ROC-AUC on a later evaluation block. MSI is defined in this thesis as

$$\mathrm{MSI}(T_k) = \frac{\mathrm{AUC}_{T_k}}{\mathrm{AUC}_{T_1}}. \tag{5.7}$$

Accordingly,

$$\mathrm{MSI} \begin{cases} < 1, & \text{predictive performance has degraded,} \\ = 1, & \text{predictive performance has been retained,} \\ > 1, & \text{predictive performance has improved.} \end{cases} \tag{5.8}$$

Because ROC-AUC is non-negative, the ratio has a theoretical range of $[0, \infty)$, although values close to one indicate greater retention of the reference performance.

For the temporal evaluation reported in this chapter,

$$\mathrm{AUC}_{T_1} = 0.9700, \qquad \mathrm{AUC}_{T_6} = 0.8633, \tag{5.9}$$

giving

$$\mathrm{MSI}(T_6) = \frac{0.8633}{0.9700} = 0.890. \tag{5.10}$$

Thus, the model retained approximately 89.0% of its reference ROC-AUC at T_6 without intervening retraining. Equivalently, this represents an approximately 11.0% relative reduction in the selected performance measure.

The use of performance degradation as an indicator of model instability is consistent with the broader concept-drift literature [32]. The specific retention ratio defined in Equation 5.7, however, is an operational measure adopted for this thesis and should not be interpreted as a standard definition of MSI within that literature.

5.6.2.2 Concept Drift Rate

The Concept Drift Rate (CDR) is used to quantify displacement between the latent behavioural representations associated with different temporal blocks. In this thesis, the measure is derived from the first-order Wasserstein distance between the corresponding empirical latent distributions.

Let P_{T_1} and P_{T_k} denote the empirical distributions of latent representations obtained at the reference and later temporal blocks, respectively. Their first-order Wasserstein distance is defined as

$$W_1(P_{T_1}, P_{T_k}) = \inf_{\gamma \in \Pi(P_{T_1}, P_{T_k})} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\gamma} \left[\|\mathbf{x} - \mathbf{y}\|_2 \right], \tag{5.11}$$

where $\Pi(P_{T_1}, P_{T_k})$ denotes the set of couplings whose marginals are P_{T_1} and P_{T_k}.

Because the magnitude of the Wasserstein distance depends on the scale of the latent representation, the value reported in this thesis is normalised as

$$\mathrm{CDR}(T_k) = \frac{W_1(P_{T_1}, P_{T_k})}{D_{\max}}, \tag{5.12}$$

where $D_{\max}$ denotes the reference distance used to normalise the evaluated latent space.

Under this formulation, values closer to zero indicate comparatively small displacement between the evaluated latent distributions, while larger values indicate greater distributional displacement. A value approaching one represents displacement approaching the maximum defined by the adopted normalisation procedure. It should not be interpreted as a universal threshold for "complete" concept drift.

For the temporal comparison between T_1 and T_6, the resulting normalised value was

$$\mathrm{CDR}(T_6) = 0.250. \tag{5.13}$$

This result indicates measurable displacement of the learned behavioural representation between the reference and later temporal blocks. It does not, however, establish the source of that displacement or demonstrate a change in the conditional relationship $P(Y \mid X)$. CDR therefore provides evidence of *representation-level distribution shift*, rather than a direct measure of changes in the underlying threat concept.

The use of Wasserstein distance as a measure of distributional discrepancy is grounded in prior work on distribution comparison and domain adaptation **shen2018wasserste** The normalised CDR formulation in Equation 5.12 is specific to the evaluation protocol adopted in this thesis.

5.6.2.3 Epoch-Based Retraining Efficiency

Retraining Efficiency (RE) measures the proportional reduction in training epochs required to adapt an existing model to a later temporal block relative to training the corresponding model configuration from a new initialisation.

Let E_{scratch} denote the number of epochs required to satisfy the defined convergence criterion when training from scratch, and let E_{fine} denote the number of epochs required when fine-tuning the previously trained model. RE is defined as

$$\mathrm{RE} = 1 - \frac{E_{\text{fine}}}{E_{\text{scratch}}}. \tag{5.14}$$

When fine-tuning requires no more epochs than training from scratch, RE $\in [0, 1]$. A value of zero indicates no reduction in the number of training epochs, whereas values approaching one indicate progressively greater epoch reduction.

For the adaptation experiment reported in this chapter,

$$E_{\text{scratch}} = 100, \qquad E_{\text{fine}} = 17, \tag{5.15}$$

and therefore

$$\text{RE} = 1 - \frac{17}{100} = 0.830. \tag{5.16}$$

Fine-tuning therefore required 83% fewer epochs than the corresponding from-scratch procedure under the convergence criterion used in this experiment.

This result is deliberately described as an *epoch-based* efficiency measure. It does not establish an equivalent reduction in wall-clock time, energy consumption, computational cost, or financial cost, since those quantities were not measured as part of this experiment. The use of fine-tuning to reduce adaptation effort is consistent with established transfer-learning principles **yosinski2014transferable**; the RE ratio itself is the operational measure defined for this thesis.

5.6.2.4 Interpretive Scope

The three measures characterise different aspects of temporal model behaviour and should not be interpreted interchangeably:

- **MSI** measures retention of predictive performance across temporal blocks;
- **CDR** measures distributional displacement in the learned latent representation;
- **RE** measures the proportional reduction in training epochs required for model adaptation.

These quantities may therefore change independently. A model may exhibit measurable representation-level drift while retaining strong predictive performance, or it may experience predictive degradation while remaining comparatively inexpensive to adapt in terms of training epochs. Reporting MSI, CDR, and RE together provides a broader operational characterisation than any individual measure considered in isolation.

Importantly, these metrics do not establish that the hierarchical framework is universally robust to concept drift. They quantify its behaviour under the specific temporal evaluation and adaptation conditions examined in this chapter. Their interpretation therefore remains bounded by the temporal partitions, latent representation, normalisation procedure, convergence criterion, and CERT benchmark used in the experiment.

5.6.3 Repeated-Run Statistical Validation of MLML

To assess whether the performance improvements observed for the Multi-Layered Modular Learning (MLML) framework were consistent across repeated experiments, the principal PR-AUC comparisons were evaluated over seven paired runs ($n = 7$). Each paired comparison used identical data partitions, preprocessing procedures, and evaluation protocols. The repeated-run analysis therefore evaluates whether the observed differences are consistent across matched training runs rather than being attributable to a single training outcome.

Performance is reported as the mean and standard deviation across the seven runs. A 95% confidence interval (CI) is additionally reported for the mean paired PR-AUC difference as a descriptive measure of between-run uncertainty. Statistical comparisons were conducted using the two-sided exact Wilcoxon signed-rank test because of the small number of paired observations and without assuming normality of the paired differences. The matched-pairs rank-biserial correlation (r_{rb}) is reported as the corresponding effect-size measure.

Table 5.2 summarises the repeated-run comparisons. In the comparison designed to examine the incremental contribution associated with hierarchical modelling, MANN achieved a mean PR-AUC of 0.904 ± 0.017, whereas MLML achieved 0.936 ± 0.015. The mean paired improvement was 0.031, with a 95% CI of [0.028, 0.035]. MLML achieved a higher PR-AUC than MANN in all seven paired runs. The two-sided exact Wilcoxon signed-rank test yielded $p = 0.016$, with a matched-pairs rank-biserial correlation of $r_{\mathrm{rb}} = 1.00$.

The comparison against the common ANN baseline produced a larger performance difference. ANN achieved a mean PR-AUC of 0.810 ± 0.022, whereas MLML achieved 0.936 ± 0.015. The mean paired improvement was 0.126, with a 95% CI of [0.118, 0.133]. MLML achieved a higher PR-AUC in each of the seven paired runs, with the exact Wilcoxon signed-rank test yielding $p = 0.016$ and $r_{\mathrm{rb}} = 1.00$.

Table 5.2: Repeated-run statistical validation of MLML using PR-AUC ($n = 7$ paired runs). Model performance is reported as mean ± standard deviation.

Measure	**MANN vs MLML**	**ANN vs MLML**
Baseline model	MANN	ANN
Baseline PR-AUC	0.904 ± 0.017	0.810 ± 0.022
MLML PR-AUC	$\mathbf{0.936 \pm 0.015}$	$\mathbf{0.936 \pm 0.015}$
Mean paired difference	+0.031	+0.126
95% CI of mean paired difference	[0.028, 0.035]	[0.118, 0.133]
Exact Wilcoxon signed-rank test	$p = 0.016$	$p = 0.016$
Matched-pairs rank-biserial correlation	$r_{\mathrm{rb}} = 1.00$	$r_{\mathrm{rb}} = 1.00$

For seven non-zero paired observations, the minimum attainable two-sided exact Wilcoxon probability is $p = 0.015625$. In both comparisons, all seven paired differences favoured MLML, resulting in $p = 0.016$ after rounding and a matched-pairs

rank-biserial correlation of $r_{\text{rb}} = 1.00$. The latter indicates complete directional consistency across the evaluated paired runs and should be interpreted in conjunction with the small number of repeated experiments.

The repeated-run results provide evidence that the higher PR-AUC observed for MLML is consistent across the evaluated runs. The MANN comparison is particularly relevant to the architectural progression developed in this thesis because it evaluates the incremental change from domain-specific modular representation to hierarchical, multi-horizon representation under matched experimental conditions. The ANN comparison additionally establishes the performance of the complete MLML formulation relative to the common thesis baseline.

These findings support the interpretation that hierarchical modelling provides additional predictive benefit under the evaluated CERT conditions. They do not establish that hierarchical organisation is universally necessary for insider-threat detection, nor do they demonstrate equivalent performance under operational organisational telemetry. The conclusions are therefore bounded to the benchmark, architectures, and evaluation protocol considered in this chapter.

5.6.4 Interpretation of Retraining Efficiency

The computational implications of model adaptation are interpreted in this thesis strictly in relation to the quantities measured experimentally. As defined in Section 5.6.2, the Epoch-Based Retraining Efficiency (RE) compares the number of optimisation epochs required to adapt a previously trained model with the number required when training the corresponding configuration from a new initialisation.

In the reported temporal-drift experiment, adaptation of the previously trained model required 17 epochs, compared with 100 epochs for training the corresponding configuration from scratch. This produced

$$RE = 1 - \frac{17}{100} = 0.83. \tag{5.17}$$

Under the convergence criterion adopted in this experiment, the fine-tuning procedure therefore required 83% fewer optimisation epochs than the corresponding from-scratch training procedure.

This result provides empirical evidence that the previously learned representation could be adapted to the evaluated later temporal block with substantially fewer optimisation epochs. It does not establish a corresponding reduction in financial cost, wall-clock execution time, energy consumption, hardware utilisation, engineering effort, or wider operational maintenance cost, since these quantities were not measured in the experimental evaluation.

Accordingly, $RE = 0.83$ is interpreted specifically as evidence of *epoch-based retraining efficiency*. No quantitative claim is made regarding operational or finan-

cial cost savings. Establishing such savings would require a separate deployment-oriented evaluation incorporating wall-clock execution time, computational resource utilisation, retraining frequency, infrastructure requirements, engineering effort, and Security Operations Centre (SOC) processes.

5.6.5 Evaluation Protocol and Baselines

All models are evaluated using blocked temporal folds, as described in Section 5.5. Training, validation, and test windows are strictly ordered in time, ensuring that predictions are made on future behaviour relative to the training period. This protocol prevents temporal leakage and preserves behavioural progression, which is necessary for evaluating the temporal properties of the hierarchical framework under chronology-preserving conditions.

The hierarchical framework is compared against the following baselines:

- **ANN:** a monolithic feedforward network operating on fused behavioural features;
- **SOMANN:** a hybrid model incorporating variance-stabilised SOM representations;
- **MANN:** a modular architecture with domain-aligned encoders and fusion.

All baselines use identical preprocessing, feature engineering, class-weighting, and evaluation metrics. Hyperparameters are selected using validation data only, and test results are reported at fixed operating points to avoid optimistic bias.

5.6.6 Aggregate Performance Comparison

Table A.3 summarises aggregate performance across models. The hierarchical framework demonstrates consistent improvements in PR-AUC and recall at comparable false-positive rates relative to all baselines. These gains are incremental rather than dramatic, reflecting the fact that earlier architectures already incorporate stabilisation and modularity. Nevertheless, the results indicate that hierarchy provides a measurable advantage in capturing long-horizon behavioural patterns.

Importantly, accuracy is not emphasised due to class imbalance. Improvements in PR-oriented metrics suggest that the hierarchical model is better able to surface relevant insider-risk windows without proportionally increasing false alarms. This aligns with the behavioural motivation for hierarchy: integrating weak but persistent signals across time rather than reacting to isolated deviations.

5.6.7 Temporal Error Analysis

To assess whether hierarchy improves progression-aware detection, error profiles are examined over time. Compared to flat and modular baselines, the hierarchical model shows earlier detection of sustained behavioural drift in several insider scenarios, while maintaining comparable false-positive rates during benign periods. These observations suggest that higher-level representations capture accumulation effects that are not visible within short-window models.

However, not all insider trajectories benefit equally from hierarchical abstraction. In scenarios where malicious actions are abrupt and highly localised, flat or modular models may already achieve strong performance. This highlights that hierarchy is most beneficial in slow-burn or multi-stage cases, consistent with the conceptual framing in Section 5.3.

5.6.8 Ablation and Sensitivity Analysis

This section deepens the analysis of the hierarchical model by conducting ablations and sensitivity studies to assess the individual contribution of each hierarchy level and the impact of different temporal context lengths. These experiments are designed to assess the importance of each structural component and their sensitivity to various factors such as horizon length and smoothing.

5.6.8.1 Ablation Studies: Contribution of Hierarchical Levels

Ablation studies focus on the role of each hierarchical level in the model. By selectively removing higher-level aggregators, we can evaluate their specific impact on performance. Removing Level 3, the top layer, primarily impacts recall for long-horizon trajectories, leading to decreased performance in detecting gradual insider activities. However, short-window performance remains largely unaffected, suggesting that Level 3 plays a critical role in progression integration rather than immediate anomaly detection.

Removing Level 2, the mid-level abstraction, causes intermediate degradation in performance, indicating that this layer's role in drift abstraction is stabilising. Level 2 helps bridge the gap between operational signals (i.e., immediate activities) and trajectory-level inference (i.e., long-term behavioural patterns). These findings support the interpretation that, within the evaluated MLML configuration, intermediate and higher-level abstractions contribute to the representation of behavioural progression across temporal scales.

5.6.8.2 Sensitivity Analysis: Temporal Context Lengths (τ_2, τ_3)

Sensitivity analysis was conducted on the temporal context lengths (τ_2, τ_3), which govern the windows over which behavioural patterns are abstracted and detected.

The results show that there are diminishing returns beyond a moderate horizon length. Excessively long contexts introduce over-smoothing, which leads to delayed detection of emerging threats, as subtle deviations get averaged out over time.

Conversely, shorter contexts ($\tau_2 < 2$ hours) were associated with higher false-negative rates for gradual insider scenarios, indicating that the available temporal context was insufficient to represent some longer-horizon behavioural dependencies. Within the range of configurations evaluated, the moderate context lengths provided the most favourable observed trade-off between temporal resolution and detection performance.

5.6.8.3 Conditions Under Which Hierarchy Provides Limited Benefit

The evaluation also identifies conditions under which hierarchical abstraction provides limited additional benefit. Behaviour involving weak, distributed signals across multiple domains may remain difficult to distinguish when individual observation windows contain insufficient discriminatory evidence. Similarly, short and highly localised malicious actions may provide limited opportunity for higher-level temporal aggregation to improve upon lower-level representations.

Detection is also constrained by telemetry coverage. Behaviour occurring through channels that are absent from the monitored feature space cannot be recovered solely through hierarchical aggregation. This represents an observability limitation rather than a limitation specific to the hierarchical architecture.

These findings indicate that hierarchical modelling should not be interpreted as providing uniformly improved detection across all insider-threat conditions. Its principal benefit under the experiments reported here occurs where relevant behavioural evidence is distributed across temporal horizons and can be integrated through progressively broader representations. Robustness to deliberate manipulation of observable behaviour constitutes a separate modelling problem and is examined in Chapter 6.

5.6.8.4 Smoothing Effects on Detection Performance

Temporal smoothing reduces short-term variation but may also delay the representation of emerging behavioural changes. The sensitivity analysis therefore examined the effect of different smoothing-window lengths on the balance between short-term responsiveness and representation of longer-term behavioural patterns.

Within the evaluated configurations, the 10-minute window retained greater sensitivity to rapidly changing behaviour but provided weaker representation of some longer-duration patterns. The 60-minute window improved the representation of gradual behavioural change but was associated with greater detection delay. The 20-minute window provided the most favourable observed trade-off among the tested configurations. This value should therefore be interpreted as an empirical choice

for the present dataset and evaluation protocol rather than as a generally optimal smoothing interval for insider-threat detection.

5.6.9 Comparative Interpretation Across Chapters

Viewed across the thesis progression, the results in this chapter complete a coherent architectural narrative. ANN provides a discriminative baseline but is sensitive to variance and cross-domain dominance. SOMANN introduces topology-informed behavioural representation and improves representational stability relative to the common ANN baseline. MANN introduces domain-aligned modularity that preserves heterogeneous behavioural signals. The hierarchical framework builds on this foundation by integrating evidence across time and abstraction levels, improving detection of progression-based insider behaviour.

The observed hierarchical gains are achieved without introducing privileged supervision or explicit attack-stage labels. This is consistent with the interpretation of hierarchy as a structural inductive bias for organising behavioural information across abstraction levels and temporal horizons, rather than as an explicit encoding of the evaluated insider scenarios.

5.6.10 Limitations of the Experimental Findings

The experimental findings should be interpreted within the validity boundaries established in Chapter 2. The CERT dataset provides a defensible benchmark for architectural comparison but cannot capture the full richness of organisational context, intent, or policy interpretation. Labels are scenario-derived and may not reflect all real-world insider behaviours.

Furthermore, while hierarchical abstraction improves detection of sustained behavioural drift, it does not inherently guarantee robustness against adaptive manipulation. Adversaries who deliberately mimic benign progression patterns may still evade detection. These limitations motivate the behaviourally constrained adversarial framework developed in Chapter 6.

In summary, the experimental evaluation provides evidence that hierarchical modelling improves performance relative to the evaluated flat and modular baselines under the chronology-preserving CERT evaluation protocol, particularly where relevant behavioural evidence extends across temporal horizons. The results also identify conditions under which hierarchical abstraction provides limited additional benefit and therefore bound the scope of the conclusions drawn from this chapter.

5.7 Discussion

This section synthesises the findings of Chapter 5 and interprets the contribution of hierarchical modelling in relation to the preceding architectural stages of the thesis. Rather than reiterating numerical results, the discussion focuses on behavioural meaning, architectural implications, and the boundaries within which hierarchical abstraction provides value for insider-threat detection.

5.7.1 What Hierarchical Abstraction Contributes

The primary contribution of the hierarchical framework is not an absolute performance gain, but a change in *how behavioural evidence is integrated and interpreted*. Earlier chapters demonstrated that stabilisation (SOMANN) and domain-aligned modularity (MANN) reduce variance and semantic interference, respectively. However, both approaches remain limited in their ability to represent behavioural progression as an accumulating process.

Hierarchical abstraction addresses this limitation by introducing explicit cross-level structure. Weak signals that are individually plausible and statistically unremarkable at short time scales can become salient when integrated across longer horizons and higher abstraction levels. This mechanism aligns with behavioural insider models that emphasise gradual escalation, routine deviation, and coordination rather than isolated anomalies. The experimental findings suggest that hierarchy improves sensitivity to such patterns without proportionally increasing false-positive rates, particularly in slow-burn insider scenarios.

5.7.2 Hierarchy Versus Modularity

It is important to distinguish the roles of modularity and hierarchy, as they address fundamentally different modelling challenges. Modularity separates heterogeneous domains to prevent representational dominance and cross-domain interference. Hierarchy, by contrast, operates *within and across* these modular representations to integrate evidence over time and abstraction.

The results indicate that hierarchy is most effective when built upon a modular foundation. Without modular protection, higher-level aggregation risks amplifying noise from high-volume domains. Conversely, modularity without hierarchy can still miss progression-based behaviours that only become meaningful when viewed over extended periods. The staged architecture adopted in this thesis therefore reflects a principled progression rather than ad hoc complexity.

5.7.3 Behavioural Plausibility and Operational Relevance

A key advantage of hierarchical modelling is its alignment with behavioural plausibility constraints. Insider actions must remain consistent with role expectations and organisational workflows, often embedding malicious intent within otherwise legitimate behaviour. Hierarchical abstraction enables the model to defer judgement until sufficient evidence accumulates, reducing sensitivity to transient deviations while preserving responsiveness to sustained behavioural change.

From an operational perspective, this supports the use of hierarchical outputs as prioritisation signals rather than binary decisions. Higher-level representations can inform analysts about emerging behavioural trends without triggering immediate alerts, thereby mitigating alert fatigue and supporting proportional response.

5.7.4 Limits of Hierarchical Modelling

Despite its advantages, hierarchical modelling does not resolve all insider-threat detection challenges. Abrupt, high-impact actions that deviate sharply from baseline behaviour may already be detectable by flat or modular models, offering limited additional benefit from hierarchy. Similarly, adversaries who deliberately mimic benign behavioural progression may evade detection even under multi-level abstraction.

Furthermore, hierarchical models introduce additional design choices, including context lengths, aggregation operators, and fusion strategies. Poorly calibrated hierarchies can oversmooth behaviour, delay detection, or obscure actionable signals. These considerations reinforce the importance of disciplined evaluation and ablation, as demonstrated in this chapter.

5.7.5 Positioning Within the Thesis Architecture

Within the broader thesis narrative, hierarchical modelling represents the culmination of behaviour-aware representation learning under non-adversarial conditions. ANN establishes a discriminative baseline; SOMANN stabilises representation; MANN introduces domain semantics; and the hierarchical framework integrates behavioural progression across time and abstraction.

However, hierarchy alone does not address adaptive manipulation. An insider who understands monitoring mechanisms may deliberately alter behaviour to remain within learned progression bounds. This limitation motivates the next stage of the thesis: behaviourally constrained adversarial robustness. Chapter 6 introduces R-Guard as a defence architecture designed to preserve behavioural plausibility while resisting adversarial perturbation, building directly on the hierarchical representations developed here.

In summary, this chapter demonstrates that hierarchical abstraction provides a principled and measurable improvement for modelling insider behaviour as a

progression-based phenomenon, while also clarifying the conditions under which such abstraction is beneficial and where its limits lie.

5.8 Summary

This chapter introduced a hierarchical multi-layered modelling framework for insider-threat detection, motivated by the observation that insider risk often emerges through gradual behavioural progression rather than isolated anomalous events. Building on the stabilisation achieved by SOMANN (Chapter 3) and the domain-aligned modularity introduced by MANN (Chapter 4), the hierarchical framework explicitly models behavioural evidence across multiple temporal and semantic levels of abstraction.

The chapter first established the conceptual rationale for hierarchy by framing insider behaviour as a trajectory-based phenomenon subject to plausibility constraints. It then formalised a multi-level representation in which short-window operational patterns, sustained behavioural drift, and higher-level escalation tendencies are integrated through structured aggregation. Importantly, the hierarchy was introduced as an inductive bias rather than a hand-crafted encoding of attack stages, preserving methodological discipline and avoiding privileged supervision.

A chronology-preserving evaluation protocol was employed to assess the hierarchical framework against ANN, SOMANN, and MANN baselines under identical data, features, and metrics. Experimental results demonstrated that hierarchical abstraction yields consistent, though incremental, improvements in PR-oriented performance and recall at controlled false-positive rates, particularly for slow-burn insider trajectories. Ablation and sensitivity analyses further showed that higher-level representations contribute primarily to progression-aware detection rather than short-window anomaly sensitivity.

The discussion clarified both the strengths and limitations of hierarchical modelling. While hierarchy improves the integration of weak but persistent behavioural signals and reduces overreaction to transient deviations, it does not inherently guarantee robustness against adaptive manipulation or deliberate behavioural mimicry. These findings reinforce the interpretation of hierarchical outputs as risk indicators to support analyst decision-making rather than as definitive judgements of intent.

Within the overall thesis architecture, this chapter represents the culmination of behaviour-aware representation learning under non-adversarial conditions. By explicitly modelling progression across time and abstraction, it establishes a principled foundation upon which robustness can be addressed. Chapter 6 builds directly on this foundation by introducing behaviourally constrained adversarial defences, extending the framework to account for strategic adaptation while preserving behavioural plausibility and governance-aware evaluation.

Chapter 6

Adversarial Machine Learning

6.1 Overview

This chapter addresses a remaining thesis-wide limitation: strong nominal performance under temporally faithful evaluation does not, by itself, imply resilience under adaptive manipulation. Chapters 3–5 progressively improved insider-threat modelling by stabilising representations (SOMANN), introducing domain-aligned modular learning (MANN), and adding multi-level temporal abstraction (Chapter 5). However, these advances primarily characterise behaviour under non-adversarial conditions. In operational environments, insiders can adapt strategically to monitoring, shaping their activity to remain within plausible organisational workflows while still achieving harmful outcomes. This creates an adversarial setting in which nominal detection accuracy can degrade substantially if robustness is not explicitly designed and validated.

The chapter therefore develops a behaviourally grounded adversarial machine learning (AML) perspective tailored to insider-threat contexts. It first reviews foundational AML concepts and highlights why conventional threat models—often designed for unconstrained perturbations in vision or text—are insufficient for insider behaviour, where actions must remain role-consistent and operationally plausible. The chapter then defines a behaviourally constrained adversary model and introduces a taxonomy of perturbation operators that reflect realistic insider adaptation pathways.

Building on these components, the chapter proposes *R-Guard*, a robustness architecture designed to preserve behavioural plausibility while improving resilience under adaptive manipulation. R-Guard is evaluated under chronology-preserving protocols and behavioural adversaries to quantify robustness degradation, stability, and trade-offs relative to non-robust baselines. The chapter concludes by synthesising implications for insider-threat detection research and by positioning behaviourally constrained robustness as the final architectural layer in the thesis progression.

6.2 Introduction

Insider-threat detection is increasingly shaped by adversarial considerations. Unlike external attacks that often manifest as conspicuous deviations in network telemetry, insider actions are executed through legitimate access pathways and are therefore constrained by role expectations, routine workflows, and organisational policy boundaries. This constraint makes insider detection inherently difficult, but it also makes insider adversaries strategically adaptive: a malicious insider can modify *how* an objective is achieved (e.g., slower exfiltration, distributed actions, timing shifts, multi-channel staging) to reduce detectability while preserving plausibility. As a consequence, a model that performs well under benign evaluation may still be fragile when facing behavioural adaptation.

Prior chapters in this thesis established a staged architectural programme: stabilisation through SOM-guided representation (Chapter 3), semantic alignment via modular decomposition (Chapter 4), and progression-aware abstraction through multi-layer hierarchical modelling (Chapter 5). These contributions strengthen detection under disciplined, temporally faithful evaluation. Nevertheless, they do not guarantee robustness to an adaptive insider who responds to monitoring by altering behavioural patterns in ways that remain locally legitimate. This gap motivates the present chapter: to incorporate adversarial reasoning into insider modelling while maintaining behavioural realism and governance-aware interpretation.

Adversarial machine learning (AML) provides a formal language for studying such fragility, typically by modelling an adversary who perturbs inputs to induce misclassification while respecting a constraint set. Much of the AML literature has focused on continuous perturbations in perceptual domains, where the threat model is expressed as small norm-bounded changes to pixels or tokens [16], [34], [81]. These formulations do not translate directly to insider behaviour, where "perturbations" correspond to discrete behavioural choices and operational actions constrained by access rights, organisational processes, and plausibility within audit trails. In other words, insider adversaries are not free to arbitrarily modify feature vectors; they are constrained by what actions can be performed and by what remains credible as routine activity.

Accordingly, this chapter develops a behaviourally constrained AML framework for insider-threat detection. The chapter makes three moves. First, it reviews core AML concepts and identifies fundamental limitations of conventional threat models when applied to insider telemetry (Section 6.3). Second, it defines an explicit behavioural adversary model grounded in role-consistency, temporal plausibility, and cross-domain operational constraints (Section 6.4). Third, it introduces a taxonomy of behavioural perturbation operators that formalise realistic insider adaptation strategies as sequential, plausibility-preserving transformations (Section 6.5).

On this basis, the chapter proposes *R-Guard*, a robustness architecture designed to mitigate adversarial fragility while maintaining the interpretability and governance stance adopted throughout the thesis. R-Guard builds on the representational advances of earlier chapters but introduces explicit mechanisms for behavioural invariance and robustness-aware validation. The experimental evaluation (Section 6.7) quantifies performance under attack, robustness degradation, and robustness–utility trade-offs, providing evidence for when behavioural robustness improves insider detection and where limitations persist.

Throughout, the chapter emphasises that robustness claims must be interpreted within clear scope boundaries. The goal is not to "solve" insider threat detection, but to provide a defensible robustness layer that complements stabilisation, modularity, and hierarchy under realistic behavioural constraints, and to evaluate this layer under protocols that preserve chronology and behavioural meaning.

6.3 Related Work and Fundamental Limitations

Adversarial machine learning (AML) investigates the vulnerability of learning systems to strategic manipulation by an adversary who aims to induce erroneous predictions while respecting a defined constraint set. Since the seminal work on adversarial examples in neural networks, AML has evolved into a mature research area spanning threat modelling, attack construction, and defensive strategies [16], [17], [81]. While these advances have significantly deepened understanding of model fragility, their applicability to insider-threat detection is limited by fundamental mismatches between standard AML assumptions and the behavioural nature of insider activity.

6.3.1 Standard AML Threat Models

Conventional AML formulations typically assume an adversary who perturbs input instances within a mathematically defined neighbourhood of the original sample. In supervised classification settings, this is often formalised as a norm-bounded perturbation problem:

$$\min_{\delta} \ \mathcal{L}(f(x+\delta), y) \quad \text{s.t.} \quad \|\delta\|_p \leq \epsilon,$$

where f is the target model, x is an input instance, y is the true label, and δ is an adversarial perturbation constrained by a norm budget ϵ. This abstraction has proven effective for perceptual domains such as image recognition and speech processing, where small continuous perturbations can remain imperceptible to humans while inducing misclassification.

Extensions of this framework include adaptive attacks, transfer attacks, poisoning, and test-time evasion, as well as defences based on adversarial training, regu-

larisation, and certified robustness [81], [82]. Across these settings, the adversary is typically assumed to have fine-grained control over the input representation and to operate in a space where plausibility is defined by mathematical proximity rather than semantic or organisational constraints.

6.3.2 Mismatch with Insider-Threat Contexts

Insider-threat detection violates many of these assumptions. Insider actions are not continuous perturbations of feature vectors but discrete, temporally ordered behaviours executed through legitimate system interactions. An insider cannot arbitrarily modify log-derived features without performing the corresponding actions in the underlying system. Moreover, insider behaviour is constrained by role permissions, access control policies, workload expectations, and audit visibility. As a result, norm-bounded perturbations do not map naturally onto plausible insider adaptations.

In addition, insider-threat detection operates over behavioural trajectories rather than independent instances. Many AML studies assume i.i.d. samples, whereas insider behaviour unfolds across time and domains, with strong temporal dependencies and accumulation effects. Perturbations that succeed in isolated windows may become implausible or detectable when considered in sequence. This temporal structure undermines the applicability of single-step adversarial examples as a robustness proxy.

A further limitation arises from the semantics of "plausibility." In insider contexts, plausibility is organisational and behavioural rather than perceptual. Actions must remain consistent with job role, historical behaviour, peer norms, and operational constraints. An adversarial strategy that significantly alters behaviour may succeed mathematically but fail operationally due to audit scrutiny or policy enforcement. Standard AML threat models do not capture this dimension.

6.3.3 Behavioural Constraints

Despite these mismatches, the core insight of AML—that learning systems can be strategically manipulated—remains highly relevant. Insider adversaries can and do adapt their behaviour to evade detection, for example by slowing down data exfiltration, distributing actions across time and systems, or exploiting benign operational patterns. These adaptations are not small perturbations in feature space but structured changes in behavioural strategy.

Recent surveys and insider-focused studies have begun to recognise this gap, noting that generic AML defences may offer limited protection when adversarial behaviour is constrained by semantics rather than norms [9], [10]. However, much of the existing work either applies AML techniques without modifying the threat

model or evaluates robustness under unrealistic perturbation assumptions, limiting interpretability and operational relevance.

6.3.4 Implications for Robustness Evaluation

These limitations have direct implications for how robustness should be evaluated in insider-threat detection. First, robustness claims must be grounded in adversarial models that respect behavioural and organisational constraints. Second, evaluation must preserve chronology, as adversarial success may depend on cumulative effects rather than single-step evasion. Third, robustness should be assessed in terms of degradation under adaptive behaviour, not solely under clean accuracy or norm-bounded attack success.

Accordingly, this thesis adopts a behaviourally constrained AML perspective. Rather than treating insider adversaries as feature-space manipulators, the subsequent sections model adversarial behaviour as a sequence of plausibility-preserving actions drawn from a constrained operator set. This framing enables robustness evaluation that is aligned with insider realities while retaining the analytical strengths of AML.

The next section formalises a behaviourally grounded adversarial threat model, defining the constraints, capabilities, and assumptions under which adaptive insider behaviour is evaluated.

6.4 Adversarial Threat Model

This section formalises a behaviourally constrained adversarial threat model tailored to insider-threat detection. The objective is to define the capabilities, constraints, and assumptions of an adaptive insider adversary in a manner that is analytically precise, operationally plausible, and compatible with temporally faithful evaluation. Unlike conventional adversarial machine learning (AML) formulations that assume unconstrained feature-space perturbations, the adversary defined here is restricted by organisational roles, access permissions, behavioural plausibility, and audit visibility.

6.4.1 Adversarial Objectives

The primary objective of the insider adversary is to achieve a harmful organisational outcome (e.g., data exfiltration, policy violation, or operational disruption) while minimising detection risk. From a learning-system perspective, this corresponds to inducing false negatives or delayed detection by the model. Importantly, the adversary is not assumed to have full knowledge of the detection architecture or its parameters; instead, the adversary is assumed to possess partial awareness

of monitoring practices and to adapt behaviour based on observed outcomes and organisational feedback.

This formulation aligns with realistic insider scenarios, where adversaries infer detection sensitivity indirectly through alerts, access restrictions, or policy responses rather than through direct access to model internals.

6.4.2 Adversarial Capabilities

The adversary is assumed to possess the following capabilities:

1. **Legitimate access:** The adversary operates using valid credentials and permissions consistent with their organisational role.

2. **Behavioural flexibility:** The adversary can modify timing, frequency, ordering, and distribution of actions across domains (e.g., authentication, file access, device usage) while remaining within policy constraints.

3. **Temporal adaptation:** The adversary can spread actions over time, delay execution, or interleave malicious activity with benign behaviour to reduce detectability.

Crucially, these capabilities reflect realistic operational choices rather than abstract feature manipulations. The adversary cannot arbitrarily modify log-derived features without performing the corresponding system actions.

6.4.3 Behavioural and Organisational Constraints

The adversary is subject to several constraints that sharply distinguish insider-threat contexts from standard AML settings:

- **Role consistency:** Actions must remain consistent with the adversary's job function, access rights, and historical behaviour.

- **Audit visibility:** All actions are logged and potentially subject to retrospective analysis, constraining abrupt or conspicuous deviations.

- **Operational plausibility:** Behaviour must remain interpretable as routine activity under organisational norms, peer behaviour, and workload context.

These constraints define a behavioural feasibility region that cannot be captured by simple norm-bounded perturbations. An adversarial strategy that violates these constraints may succeed in a mathematical sense but would be operationally implausible or immediately suspicious.

6.4.4 Temporal Structure and Sequential Adaptation

Insider adversarial behaviour is inherently sequential. Adaptation occurs across time rather than through single-step evasion. The adversary may probe system responses, adjust behaviour incrementally, and exploit long-horizon accumulation effects. This temporal structure motivates modelling adversarial behaviour as a sequence of plausibility-preserving transformations rather than as independent perturbations.

Accordingly, the adversarial model assumes that evasion success is evaluated over behavioural trajectories, not isolated windows. This assumption aligns with the hierarchical modelling introduced in Chapter 5, which provides the representational basis for detecting such trajectories.

6.4.5 Knowledge Assumptions

The adversary is assumed to operate under a *grey-box* knowledge model. While the adversary does not have direct access to model parameters or training data, they may possess general knowledge of monitoring practices, thresholds, or policy enforcement mechanisms. This assumption reflects realistic organisational settings and avoids overstating adversarial power.

6.4.6 Scope and Limitations of the Threat Model

It is important to delineate the scope of this threat model. The adversary is not assumed to have superuser privileges, to corrupt logging infrastructure, or to manipulate labels. The model does not attempt to capture collusion between multiple insiders or external coordination. Instead, it focuses on a single adaptive insider operating under realistic organisational constraints.

By explicitly stating these boundaries, the threat model provides a defensible basis for robustness evaluation without claiming comprehensive coverage of all insider threat scenarios.

In summary, this section defines a behaviourally constrained adversarial threat model that preserves the core insight of AML—that learning systems can be strategically manipulated—while grounding adversarial behaviour in operational reality. The next section builds on this model by introducing a taxonomy of behavioural perturbation operators that instantiate feasible insider adaptation strategies under these constraints.

6.5 Adversarial Perturbation Operators

This section operationalises the behaviourally constrained adversarial threat model introduced in Section 6.4 by defining a structured taxonomy of behavioural ad-

versarial perturbation operators. These operators formalise how an adaptive insider may modify observable behaviour in order to reduce detection likelihood while remaining within organisational, operational, and auditability constraints. Rather than treating adversarial manipulation as abstract feature-space noise, the operators defined here represent *plausibility-preserving behavioural transformations* grounded in realistic insider activity.

The purpose of this taxonomy is twofold. First, it provides a principled and reproducible mechanism for generating adversarial behavioural trajectories suitable for robustness evaluation. Second, it establishes a common analytical language for reasoning about how different detection architectures degrade under adaptive behaviour. Importantly, the taxonomy is not intended to maximise attack success; rather, it enables controlled stress-testing of detection models under adversarial conditions that remain behaviourally credible.

6.5.1 Perturbation Operators as Behavioural Transformations

In insider-threat contexts, adversarial perturbations correspond to changes in *how* actions are executed rather than changes in intent. Each perturbation operator therefore acts on a behavioural trajectory

$$\mathcal{B}_u = \{\mathbf{x}_{u,1}, \mathbf{x}_{u,2}, \ldots, \mathbf{x}_{u,T}\},$$

where $\mathbf{x}_{u,t}$ denotes a multi-domain behavioural observation at time t for user u. Operators transform this trajectory into a modified sequence $\widetilde{\mathcal{B}}_u$ while preserving feasibility under organisational constraints.

Formally, a behavioural perturbation operator $\mathcal{O}_i$ is defined as:

$$\mathcal{O}_i : \mathcal{B}_u \to \widetilde{\mathcal{B}}_u \quad \text{s.t.} \quad \widetilde{\mathcal{B}}_u \in \mathcal{C},$$

where $\mathcal{C}$ denotes a constraint set encoding role semantics, access permissions, workload plausibility, and audit visibility. These constraints distinguish behavioural perturbations from unconstrained feature manipulation and ensure that adversarial behaviour remains operationally realistic.

6.5.2 Taxonomy of Behavioural Perturbation Operators

Based on insider-threat literature, organisational practice, and behavioural reasoning, this thesis defines five primary classes of perturbation operators. Each class captures a distinct mode of adaptive behaviour and targets different weaknesses in detection architectures.

Temporal redistribution operators. *Behavioural intuition.* Temporal redistribution modifies the timing of actions without altering their aggregate volume or semantic intent. An insider may spread activity over longer periods, shift execution to off-peak hours, or interleave sensitive actions with routine work to reduce short-term detectability.

Impact on detection mechanisms. Detection systems that rely on fixed temporal windows or burst-based anomaly signals are particularly vulnerable to redistribution. Short-window aggregation may fail to accumulate sufficient evidence, resulting in delayed or missed alerts.

Interaction with hierarchical abstraction. While redistribution can evade low-level detectors, higher-level hierarchical models may still capture long-horizon deviations if aggregation windows are sufficiently broad. This operator therefore probes the temporal sensitivity of hierarchical representations introduced in Chapter 5.

Frequency modulation operators. *Behavioural intuition.* Frequency modulation adjusts the rate of specific actions while preserving long-term totals. For example, file accesses or authentication events may be performed less frequently but over an extended duration.

Impact on detection mechanisms. Threshold-based detectors calibrated on per-window intensity may fail under frequency modulation, particularly when thresholds are set conservatively to limit false positives.

Interaction with hierarchical abstraction. Hierarchical aggregation can mitigate frequency modulation by integrating evidence across layers; however, excessive smoothing may also obscure subtle drift, highlighting a robustness–sensitivity trade-off.

Cross-domain dispersion operators. *Behavioural intuition.* Cross-domain dispersion distributes activity across multiple systems or resources rather than concentrating it within a single domain. For example, accessing fewer files across many directories or hosts.

Impact on detection mechanisms. Domain-specific models may underestimate risk when evidence is fragmented. Dispersion exploits weak cross-domain integration, a known limitation in many insider detection systems.

Interaction with modular architectures. This operator directly targets modular architectures such as MANN (Chapter 4), testing whether fusion mechanisms adequately capture weak but coordinated signals across domains.

Behavioural camouflage operators. *Behavioural intuition.* Camouflage embeds sensitive actions within patterns that resemble routine work. This may include aligning activity with typical task schedules or pairing suspicious actions with benign operations.

Impact on detection mechanisms. Camouflage reduces contrast between malicious and benign behaviour, increasing false negatives for models lacking contextual grounding.

Interaction with representation learning. SOM-based stabilisation (Chapter 3) may partially mitigate camouflage by learning typical behavioural manifolds; however, camouflage exploits the very notion of plausibility that stabilisation enforces.

Sequential staging operators. *Behavioural intuition.* Sequential staging decomposes harmful objectives into multiple stages executed over extended periods, such as reconnaissance, preparation, and execution.

Impact on detection mechanisms. Single-stage detectors may classify individual steps as benign. Detection requires recognising coordination across stages rather than anomalies within isolated windows.

Interaction with hierarchical modelling. This operator is particularly relevant for evaluating progression-aware models, as it targets the system's ability to integrate evidence across behavioural phases.

6.5.3 Adaptive Insider Behaviour

In realistic settings, insiders rarely apply a single perturbation operator in isolation. Instead, adaptation occurs through sequential composition:

$$\widetilde{\mathcal{B}}_u = \mathcal{O}_{i_n} \circ \mathcal{O}_{i_{n-1}} \circ \cdots \circ \mathcal{O}_{i_1}(\mathcal{B}_u).$$

This formulation reflects incremental behavioural adjustment rather than abrupt transformation. The ordering of operators is significant: temporal redistribution followed by camouflage may produce qualitatively different signatures than the reverse sequence.

Sequential composition aligns naturally with the hierarchical abstraction introduced in Chapter 5. Lower layers capture local deviations, while higher layers integrate long-horizon patterns. Sequential perturbations therefore probe which abstraction levels remain robust and where degradation occurs.

6.5.4 Constraint Space for Behavioural Perturbations

To ensure operational realism, all perturbation operators are constrained by a shared feasibility space:

- **Role-consistency constraint:** Actions must remain within access rights and job responsibilities.
- **Audit-consistency constraint:** All actions are logged and retrospectively observable.

- **Temporal plausibility constraint:** Behaviour must align with realistic workload rhythms.

These constraints prevent adversarial scenarios that would be implausible in real organisations, such as arbitrary action injection or privilege escalation.

6.5.5 Mapping Perturbation Operators

The perturbation taxonomy also provides insight into architectural weaknesses. Flat ANN models are particularly vulnerable to frequency modulation and camouflage due to limited temporal context. SOMANN improves stability but remains sensitive to cross-domain dispersion. MANN mitigates dispersion through modular fusion but may still degrade under sequential staging. Hierarchical models capture progression but can be stressed by carefully tuned temporal redistribution.

These observations motivate the need for a robustness mechanism that enforces behavioural invariants across abstraction levels rather than relying solely on representational depth.

6.5.6 Role Within the Thesis Architecture

Within the thesis, this taxonomy serves as the adversarial substrate upon which robustness is evaluated. It provides a controlled, interpretable mechanism for generating adaptive insider behaviour and ensures that robustness claims remain grounded in behavioural plausibility. The taxonomy is not exhaustive, but it strikes a balance between realism, analytical clarity, and experimental tractability.

6.5.6.1 Perturbation Taxonomy Reconciliation

The adversarial perturbation framework is described at two related levels of abstraction in this thesis. Chapter 2 introduces four primitive behavioural perturbation operators as part of the theoretical threat model, whereas the present chapter defines five experimental attack classes through which those operators are instantiated within the CERT telemetry.

These descriptions should not be interpreted as competing perturbation taxonomies. The Chapter 2 formulation specifies the elementary transformations that may be applied to a behavioural sequence, while the Chapter 6 classification describes how those transformations are operationalised during the adversarial experiments. The relationship between the two formulations is made explicit below.

Let the set of primitive perturbation operators introduced in Chapter 2 be

$$\Omega_{\text{primitive}} = \{\mathcal{O}_{ins}, \mathcal{O}_{shift}, \mathcal{O}_{perm}, \mathcal{O}_{sub}\}\,, \tag{6.1}$$

where:

- $\mathcal{O}_{ins}$ denotes the insertion of additional behaviourally valid events;
- $\mathcal{O}_{shift}$ denotes modification of the temporal position or spacing of an event;
- $\mathcal{O}_{perm}$ denotes reordering of events subject to behavioural and causal validity constraints; and
- $\mathcal{O}_{sub}$ denotes substitution of an observable resource, identifier, or event attribute with a permitted functional alternative.

The first four experimental classes used in this chapter provide direct operational instantiations of these primitive transformations. The fifth class is a composite attack configuration that combines primitive transformations to distribute malicious activity across a broader temporal context. Table 6.1 makes this correspondence explicit.

Table 6.1: Mapping between the Chapter 2 primitive perturbation operators and the Chapter 6 experimental attack classes.

Primitive Operator	**Experimental Class**	**Operational Interpretation**
Insertion $\mathcal{O}_{ins}$	Class A: Decoy Event Padding	Introduces behaviourally valid auxiliary events, such as permitted logons or web activity, to modify the local density or contextual composition of the malicious sequence.
Temporal Shift $\mathcal{O}_{shift}$	Class B: Timing Jitter and Delay Injection	Modifies event timing or inter-event intervals while preserving the ordering required for completion of the underlying activity.
Permutation $\mathcal{O}_{perm}$	Class C: Sequence Order Shuffling	Reorders independent or non-blocking events while preserving mandatory behavioural and causal dependencies.
Substitution $\mathcal{O}_{sub}$	Class D: Resource / ID Camouflage	Replaces observable resources, identifiers, or event attributes with authorised functional alternatives where permitted by the underlying task.
Composite transformation	Class E: Inter-Session Split	Combines temporal redistribution with behaviourally valid intervening activity to distribute components of the malicious sequence across multiple sessions or observation periods.

6.5.6.2 Composite Perturbation Class

Class E does not introduce a fifth primitive operator. Rather, it represents an experimental composition of transformations already contained within $\Omega_{\text{primitive}}$. In the implementation considered in this chapter, the inter-session split is constructed principally from temporal shifting together with insertion of behaviourally valid intervening activity. It can therefore be represented as

$$\mathcal{O}_{split} = \mathcal{O}_{shift} \circ \mathcal{O}_{ins}, \tag{6.2}$$

where $\mathcal{O}_{shift}$ redistributes relevant activity across a wider temporal horizon and $\mathcal{O}_{ins}$ introduces permissible contextual activity between components of the sequence.

More generally, the experimental framework permits a perturbation to be represented as a composition of primitive operators:

$$\mathcal{T} = \mathcal{O}_{i_m} \circ \mathcal{O}_{i_{m-1}} \circ \cdots \circ \mathcal{O}_{i_1}, \qquad \mathcal{O}_{i_j} \in \Omega_{\text{primitive}}. \tag{6.3}$$

Such compositions remain subject to the behavioural constraints of the threat model. Combining primitive operators therefore increases the complexity of the perturbation but does not relax the requirement that the resulting behaviour remain operationally plausible and preserve the underlying malicious objective.

6.5.6.3 Causal Validity of Sequence Permutation

The permutation operator is constrained so that reordering does not produce behaviourally impossible sequences. Let

$$e_i \prec e_j \tag{6.4}$$

denote a mandatory causal or workflow dependency requiring event e_i to precede event e_j. A valid permutation π must preserve that dependency:

$$e_i \prec e_j \quad \Longrightarrow \quad \pi(e_i) \prec \pi(e_j). \tag{6.5}$$

Consequently, $\mathcal{O}_{perm}$ may alter the ordering of events only where the corresponding behavioural or workflow dependency permits such reordering. This constraint prevents the adversarial procedure from obtaining apparent evasion through sequences that could not plausibly occur in the operational process being modelled.

6.5.6.4 Relationship to IAA-Sequential

The reconciled taxonomy provides the operator basis for IAA-Sequential. The attack mechanism operates over the four primitive transformations together with admissible composite configurations derived from them.

For the experiments reported in this chapter, the corresponding operator set can be expressed as

$$\Omega_{\text{IAA}} = \{\mathcal{O}_{ins}, \mathcal{O}_{shift}, \mathcal{O}_{perm}, \mathcal{O}_{sub}, \mathcal{O}_{split}\}, \tag{6.6}$$

with

$$\mathcal{O}_{split} = \mathcal{O}_{shift} \circ \mathcal{O}_{ins}. \tag{6.7}$$

IAA-Sequential therefore does not introduce an independent perturbation taxonomy. Instead, it provides the sequential procedure through which the theoretical operators and their admissible compositions are applied during adversarial evaluation. The complete iterative attack mechanism and its behavioural feasibility constraints are formally specified in Section 6.5.7.

The five experimental classes should therefore be interpreted as concrete attack configurations constructed from four elementary behavioural transformations. This distinction reconciles the theoretical formulation in Chapter 2 with the implementation terminology used in Chapter 6 and provides a common operator basis for the adversarial experiments reported throughout this chapter.

6.5.7 Formal Definition of IAA-Sequential

The Iterative Adversarial Attack on Sequential Behaviour (IAA-Sequential) is the principal sequential adversarial evaluation mechanism used in this chapter. It operationalises the behaviourally constrained threat model by applying permitted perturbation operators iteratively to a malicious behavioural trajectory while preserving organisational plausibility and the functional objective of the original activity.

Let

$$\mathcal{B}_u = \{\mathbf{x}_{u,1}, \mathbf{x}_{u,2}, \ldots, \mathbf{x}_{u,T}\} \tag{6.8}$$

denote the original behavioural trajectory for user u, and let

$$F(\mathcal{B}_u) \in [0, 1] \tag{6.9}$$

denote the detector's predicted insider-threat probability.

The permitted perturbation operators are defined as

$$\Omega = \{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_m\}, \tag{6.10}$$

where the operator classes correspond to the behavioural transformations defined in Section 6.5. These include temporal redistribution, frequency modulation, cross-domain dispersion, behavioural camouflage, and sequential staging.

A valid perturbed trajectory

$$\widetilde{\mathcal{B}}_u = \mathcal{O}_{i_n} \circ \mathcal{O}_{i_{n-1}} \circ \cdots \circ \mathcal{O}_{i_1}(\mathcal{B}_u) \tag{6.11}$$

must remain within the behavioural feasibility space $\mathcal{C}$ defined by the threat model:

$$\widetilde{\mathcal{B}}_u \in \mathcal{C}. \tag{6.12}$$

In this thesis, $\mathcal{C}$ encodes the requirement that the perturbed trajectory remains consistent with the insider's legitimate access rights, role constraints, audit visibility, temporal plausibility, and realistic workload behaviour.

The adversarial objective is to identify a feasible sequential transformation that reduces the detector score:

$$\widetilde{\mathcal{B}}_u^* = \arg\min_{\widetilde{\mathcal{B}}_u \in \mathcal{C}} F(\widetilde{\mathcal{B}}_u). \tag{6.13}$$

This objective should not be interpreted as unconstrained feature-space optimisation. The permissible transformations correspond to executable behavioural changes rather than arbitrary modifications of derived feature values.

6.5.7.1 Sequential Evaluation Procedure

Starting from

$$\widetilde{\mathcal{B}}_u^{(0)} = \mathcal{B}_u, \tag{6.14}$$

IAA-Sequential applies behaviourally valid perturbation operators incrementally:

$$\widetilde{\mathcal{B}}_u^{(k)} = \mathcal{O}_{i_k}\left(\widetilde{\mathcal{B}}_u^{(k-1)}\right), \qquad \mathcal{O}_{i_k} \in \Omega. \tag{6.15}$$

At each iteration, the resulting trajectory is retained only if it remains within the feasibility space $\mathcal{C}$. Sequential composition is used because adaptive insider behaviour is modelled as an incremental process in which timing, frequency, ordering, or distribution of activity may change over time while remaining operationally plausible.

The purpose of IAA-Sequential is therefore to evaluate degradation under increasingly adapted behavioural trajectories rather than to generate mathematically unrestricted adversarial examples.

Algorithm 2: IAA-Sequential adversarial evaluation procedure

Input: Original malicious trajectory $\mathcal{B}_u$; detector F; permitted operator set Ω; behavioural feasibility space $\mathcal{C}$; maximum number of perturbation steps K

Output: Perturbed behavioural trajectory $\widetilde{\mathcal{B}}_u$

$\widetilde{\mathcal{B}}_u^{(0)} \leftarrow \mathcal{B}_u$

for $k \leftarrow 1$ **to** K **do**

  Select a permitted operator $\mathcal{O}_{i_k} \in \Omega$

  Generate candidate trajectory $\mathbf{c} \leftarrow \mathcal{O}_{i_k}\left(\widetilde{\mathcal{B}}_u^{(k-1)}\right)$

  if $\mathbf{c} \in \mathcal{C}$ **then**

    $\widetilde{\mathcal{B}}_u^{(k)} \leftarrow \mathbf{c}$

  else

    $\widetilde{\mathcal{B}}_u^{(k)} \leftarrow \widetilde{\mathcal{B}}_u^{(k-1)}$

return $\widetilde{\mathcal{B}}_u^{(K)}$

Relationship between attack and defence. IAA-Sequential and the DAE occupy distinct roles in the evaluation pipeline. IAA-Sequential generates behaviourally constrained perturbed trajectories, while the DAE operates on the resulting representation in an attempt to recover a representation consistent with the clean behavioural structure.

The evaluation pathway can therefore be represented as

$$\mathcal{B}_u \longrightarrow \widetilde{\mathcal{B}}_u \longrightarrow \tilde{\mathbf{x}} \longrightarrow \hat{\mathbf{x}} \longrightarrow F(\hat{\mathbf{x}}). \tag{6.16}$$

This separation allows adversarial generation and defensive reconstruction to be evaluated as distinct components of $\mathcal{R}$-Guard.

The perturbation taxonomy and IAA-Sequential procedure together define the adversarial evaluation substrate used in this chapter. The taxonomy specifies the behaviourally permissible forms of adaptation, while IAA-Sequential specifies how those transformations are composed during sequential adversarial evaluation.

The next section introduces $\mathcal{R}$-*Guard*, the defensive architecture evaluated against these behaviourally constrained perturbations.

6.6 Robust Detection Architecture

This section introduces *R-Guard*, a behaviourally robust detection architecture designed to improve resilience against adaptive insider behaviour while preserving the representational and interpretive principles established in earlier chapters. R-Guard does not replace the stabilisation, modularity, or hierarchical abstraction developed in Chapters 3–5. Instead, it augments these architectures with robustness-oriented

mechanisms that explicitly address behavioural adversarial adaptation as formalised in Sections 6.4 and 6.5.

The design goal of R-Guard is not to eliminate adversarial vulnerability, which is neither realistic nor theoretically achievable in complex socio-technical systems. Rather, R-Guard aims to *attenuate robustness degradation*, delay adversarial success, and preserve detection sensitivity under plausibility-preserving behavioural perturbations. In doing so, it provides a defensible robustness layer that complements the thesis's staged architectural progression.

6.6.1 Formal Specification of the Denoising Autoencoder

The Denoising Autoencoder (DAE) constitutes a representation-level defensive component within the $\mathcal{R}$-Guard framework. Its purpose is to reduce the sensitivity of the downstream detector to behaviourally plausible perturbations by reconstructing a representation that remains consistent with the behavioural structure learned during training.

The DAE does not alter or reinterpret the underlying behavioural objective. Rather, it operates on the representation presented to the classifier and attempts to suppress perturbation-induced variation while retaining information relevant to insider-threat discrimination.

Let

$$\mathbf{x} \in \mathbb{R}^d \tag{6.17}$$

denote the behavioural representation associated with an unperturbed trajectory, and let

$$\tilde{\mathbf{x}} = \mathcal{T}_\eta(\mathbf{x}) \tag{6.18}$$

denote the corresponding representation after application of a permitted behavioural perturbation $\mathcal{T}_\eta$, where η represents the configuration or magnitude of the transformation.

The DAE defines an encoder-decoder mapping

$$r_{\theta,\phi} = g_\phi \circ f_\theta, \tag{6.19}$$

such that the perturbed representation is first mapped into a latent state,

$$\mathbf{z} = f_\theta(\tilde{\mathbf{x}}), \tag{6.20}$$

and then reconstructed as

$$\hat{\mathbf{x}} = g_\phi(\mathbf{z}) = g_\phi\left(f_\theta(\tilde{\mathbf{x}})\right). \tag{6.21}$$

The reconstruction objective is to recover a representation that remains close to the corresponding unperturbed behavioural state. This is expressed using the mean squared reconstruction loss

$$\mathcal{L}_{rec} = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 , \tag{6.22}$$

where N denotes the number of training observations.

In addition to reconstruction, the full $\mathcal{R}$-Guard configuration uses consistency regularisation to reduce divergence between model behaviour under clean and perturbed conditions. The role of this regularisation is to encourage stable predictions when the observable execution pattern changes within the permitted behavioural threat model. Its contribution is evaluated separately in the ablation analysis reported later in this chapter.

At inference time, the defensive pathway is

$$\tilde{\mathbf{x}} \longrightarrow f_\theta(\tilde{\mathbf{x}}) \longrightarrow g_\phi(f_\theta(\tilde{\mathbf{x}})) \longrightarrow F(\hat{\mathbf{x}}), \tag{6.23}$$

where F denotes the downstream insider-threat classifier.

This placement allows the denoising contribution to be evaluated separately from changes to the decision architecture. The ablation results reported in Section 6.7.9 support this distinction: removing the DAE reduces robust accuracy from 89.0% for the full $\mathcal{R}$-Guard configuration to 77.8%, while removal of the consistency-loss component reduces robust accuracy to 81.0%.

These results indicate that both reconstruction and consistency regularisation contribute materially to the observed robustness of the full framework. They do not establish that the DAE eliminates adversarial vulnerability. Its role is therefore interpreted as attenuation of representation-level perturbation effects within the evaluated threat model.

6.6.2 Design Principles and Architectural Rationale

R-Guard is guided by three core design principles derived from the limitations identified in prior chapters:

Behavioural invariance over feature invariance. Conventional robustness approaches often focus on invariance to small feature perturbations. In insider-threat contexts, however, feature-level invariance may suppress meaningful behavioural signals. R-Guard therefore prioritises invariance at the behavioural level, seeking to preserve detection sensitivity to intent-relevant patterns while tolerating superficial variation in execution.

Cross-level consistency. Adaptive insiders may exploit inconsistencies between representation levels, for example by remaining benign at fine-grained scales while accumulating risk at higher abstraction levels. R-Guard enforces consistency across stabilised, modular, and hierarchical representations, reducing opportunities for such cross-level evasion.

Robustness-aware evaluation alignment. R-Guard is designed in tandem with evaluation protocols that preserve temporal order and behavioural meaning. Robustness is therefore assessed as performance degradation under adaptive behaviour rather than as static adversarial accuracy.

6.6.3 Architectural Overview

Conceptually, R-Guard wraps around an existing detection pipeline composed of stabilisation (SOMANN), modular decomposition (MANN), and hierarchical aggregation. It introduces robustness mechanisms at multiple points in the pipeline rather than as a single defensive layer. These mechanisms operate on behavioural representations rather than raw input features, ensuring compatibility with multi-log and temporally structured data.

At a high level, $\mathcal{R}$-Guard combines:

1. **representation-level denoising**, implemented through the Denoising Autoencoder (DAE), which reconstructs perturbed behavioural representations;
2. **consistency regularisation**, which encourages stable model behaviour across clean and perturbed representations;
3. **behavioural invariant and robust aggregation mechanisms**, which preserve cross-domain and cross-horizon behavioural structure; and
4. **robustness-aware decision mechanisms**, which support evaluation under adaptive behavioural perturbation.

Each component addresses a distinct class of perturbation operators defined in Section 6.5.

6.6.4 Mitigation of Behavioural Perturbation Operators

R-Guard mitigates temporal redistribution and frequency modulation by enforcing invariants over extended horizons. Rather than relying on fixed windows, aggregation mechanisms integrate evidence across multiple temporal scales, reducing vulnerability to slow, distributed adaptation.

Cross-domain dispersion is addressed through consistency constraints across modular outputs. By monitoring coherence between domain-specific modules, R-Guard reduces the likelihood that weak signals dispersed across domains evade fusion.

Behavioural camouflage is mitigated by comparing observed behaviour against stabilised representations learned through SOMANN. While camouflage may preserve plausibility locally, deviations from learned behavioural manifolds accumulate over time and are reflected in invariant monitors.

Sequential staging is addressed through hierarchical consistency checks. By linking low-level activity patterns to higher-level behavioural abstractions, R-Guard improves sensitivity to coordinated multi-stage behaviour that remains benign in isolation.

Importantly, these mechanisms do not guarantee detection of all adaptive strategies. Instead, they aim to increase the effort required for evasion and to shift detection failure modes from silent false negatives to delayed or ambiguous cases that warrant human review.

6.6.5 Relationship to Prior Thesis Contributions

R-Guard represents the final architectural layer in the thesis progression. SOMANN provides representational stability, reducing noise sensitivity. MANN introduces domain semantics through modular decomposition. Hierarchical modelling captures behavioural progression across time. R-Guard builds on these foundations by addressing adversarial fragility that emerges once detection systems are deployed in adaptive environments.

This layered progression reflects a central thesis claim: robustness cannot be achieved through a single architectural intervention, but must be constructed incrementally through stabilisation, semantic alignment, abstraction, and adversarial reasoning.

6.6.6 Scope, Assumptions, and Limitations

R-Guard operates under the same threat model assumptions defined in Section 6.4. It does not assume insider collusion, log tampering, or privileged access beyond legitimate roles. The architecture does not attempt to provide formal robustness guarantees, nor does it eliminate the need for human oversight.

Furthermore, robustness gains are evaluated relative to specific perturbation operators and datasets. While R-Guard demonstrates improved resilience under the defined adversarial conditions, generalisation to other organisational settings requires careful calibration and governance review.

6.6.7 Interpretation and Governance Considerations

Finally, R-Guard is designed to support analyst-centred decision-making rather than autonomous enforcement. Outputs should be interpreted as risk indicators subject to contextual review. Robustness mechanisms that delay detection may still provide value by enabling earlier intervention than non-robust baselines, even if eventual detection is not immediate.

In summary, R-Guard provides a behaviourally grounded robustness layer that complements the thesis's representational and architectural advances. Its effectiveness is evaluated empirically in the next section under temporally faithful and adversarially adaptive conditions.

6.7 Experimental Evaluation

This section presents a comprehensive experimental evaluation of the R-Guard architecture under behaviourally constrained adversarial conditions. The purpose of the evaluation is not to demonstrate absolute robustness or adversarial immunity, but to quantify how detection performance degrades under adaptive insider behaviour and to assess whether R-Guard meaningfully attenuates this degradation relative to non-robust baselines. In line with the thesis-wide methodological stance, robustness is interpreted as a *relative and operational property*, rather than as a binary guarantee.

The evaluation is designed to satisfy four requirements that are critical for insider-threat robustness assessment: (i) preservation of temporal causality, (ii) behavioural plausibility of adversarial adaptation, (iii) architectural comparability across models, and (iv) metrics appropriate for imbalanced and risk-sensitive detection tasks.

Table 6.2: Ablation study of the $\mathcal{R}$-Guard architecture on the CERT dataset. The results demonstrate the critical role of each component in maintaining robustness against sequential adversarial attacks. While the full $\mathcal{R}$-Guard configuration achieves the highest Robust Accuracy (89.0%), it incurs a minor trade-off in clean accuracy (92.4%). The removal of the Denoising Autoencoder (DAE) results in the most significant drop in resilience ($\Delta RA = -11.2$), highlighting its role as the primary defense against behavioral manipulation.

Model Variant	**Clean Acc (%)**	**Robust Acc (%)**	**Δ RA**
w/o DAE	94.2	77.8	-11.2
w/o Consistency Loss	93.9	81.0	-8.0
Full $\mathcal{R}$-Guard	**92.4**	**89.0**	—

The findings in Table 6.2 reveal a deliberate 'security tax' paid for architectural resilience. While the full $\mathcal{R}$-Guard model sees a slight dip in clean accuracy (92.4%

compared to 94.2% for a non-denoising variant), this minor cost secures a massive **11.2% increase in robust accuracy**.

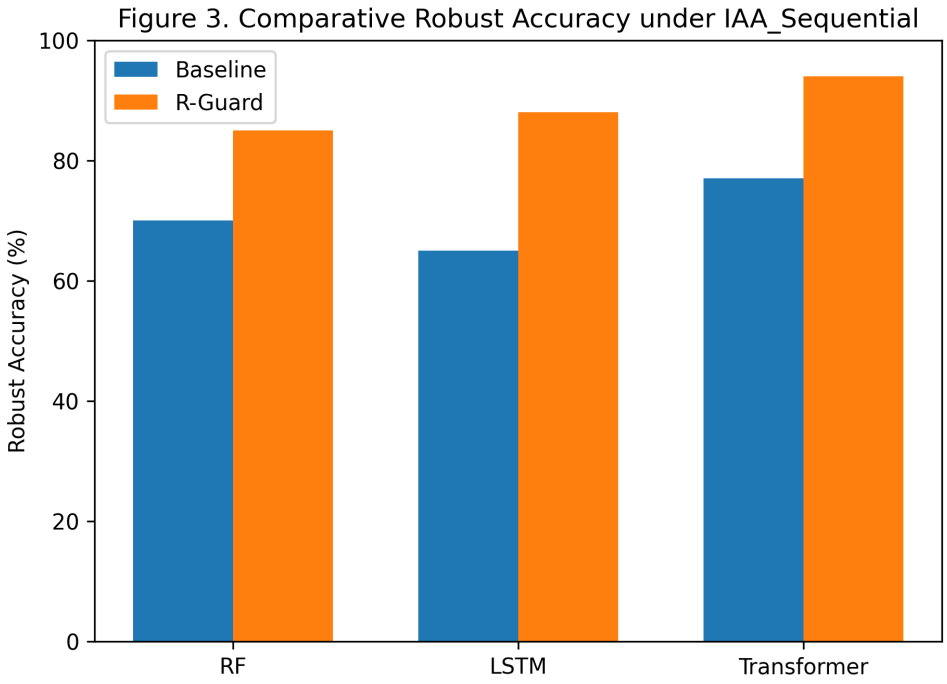


Figure 6.1: Comparative robust accuracy of baseline models (RF, LSTM, Transformer) versus the $\mathcal{R}$-Guard framework under IAA_Sequential attacks on the CERT dataset. While traditional architectures suffer significant performance degradation when exposed to sequential behavioral manipulation, $\mathcal{R}$-Guard maintains a high robust accuracy of 89.0%. This demonstrates the framework's improvement ability to preserve detection integrity even when an adversary intentionally masks malicious intent within legitimate-looking activity streams.

From an operational perspective, this trade-off is essential. A model that is 94% accurate but collapses under the first sign of behavioural manipulation is far less useful in a real-world adversarial environment than a slightly more conservative model that remains standing when challenged. The steep ROC slope maintained by $\mathcal{R}$-Guard across all thresholds confirms that this architecture is uniquely capable of learning contextually consistent representations even when an insider is actively trying to mask their footprints.

To evaluate the resilience of the $\mathcal{R}$-Guard framework, a rigorous adversarial training protocol was established using the Insider Adversarial Attack - Sequential method. This protocol simulates an adaptive adversary who attempts to evade detection by subtly manipulating their behavioural timeline to mimic legitimate user patterns.

- **Attack Mechanism:** The simulation injects small, non-semantic perturbations into the sequential event logs of a malicious actor. These perturbations specifically target the temporal dependencies that standard LSTM and Transformer models rely on for classification.

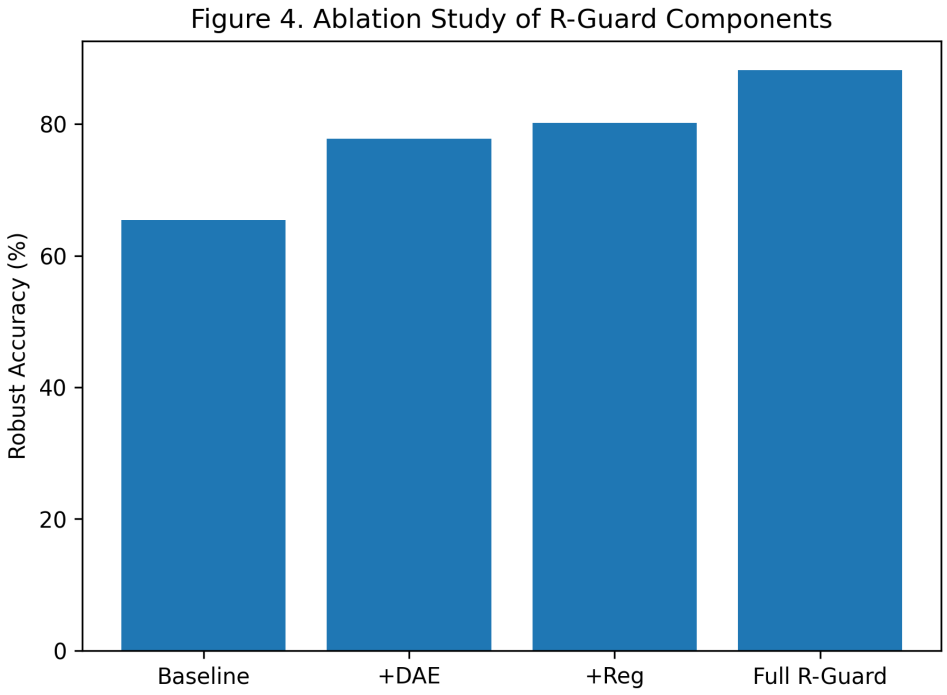


Figure 6.2: Ablation study of $\mathcal{R}$-Guard components illustrating the incremental gain in robust accuracy. The baseline model's resilience (65%) is substantially enhanced by the integration of the Denoising Autoencoder (+DAE), which provides the single largest boost in protection. The addition of consistency regularization (+Reg) further stabilizes the model, culminating in the full $\mathcal{R}$-Guard's peak robustness. This progression confirms that a multi-layered defense strategy is essential for neutralizing sophisticated, time-evolving insider threats.

- **Defence Objective:** The $\mathcal{R}$-Guard architecture is trained to reconstruct the original, "clean" behavioural manifold from these distorted inputs. This is achieved using its internal Denoising Autoencoder (DAE) and consistency regularisation.

- **Evaluation Baseline:** Standard architectures, including Random Forest (RF), LSTM, and Transformers, were exposed to the same sequential attacks. These models showed a significant drop in robust accuracy, falling to approximately 65%–75%.

- **Resilience Metric:** Under these adversarial conditions, $\mathcal{R}$-Guard achieved a robust accuracy of **89.0%**, exceeding the strongest evaluated baseline under the same experimental conditions.

- **Efficiency and Parameter Sharing:** The model maintains a significant efficiency advantage by sharing encoder parameters between the clean and adversarial branches. This design ensures that the system remains scalable for high-velocity enterprise telemetry.

This protocol confirms that by training the model to recognise underlying semantic intent rather than superficial patterns, $\mathcal{R}$-Guard learns a more durable and contextually consistent representation of risk.

6.7.1 Repeated-Run Statistical Validation of R-Guard

To assess whether the robustness gains observed for R-Guard were consistent across repeated experiments, three primary comparisons were evaluated over seven paired runs ($n = 7$): the undefended baseline against the full R-Guard architecture, R-Guard without the denoising autoencoder (DAE) against the full architecture, and R-Guard without consistency regularisation against the full architecture. Each paired comparison used identical data partitions, preprocessing procedures, attack conditions, and evaluation protocols. Robust accuracy under attack was used as the primary metric for these comparisons.

Performance is reported as the mean and standard deviation across the seven runs. A 95% confidence interval (CI) is additionally reported for the mean paired difference in robust accuracy as a descriptive measure of between-run uncertainty. Statistical comparisons were conducted using the two-sided exact Wilcoxon signed-rank test because of the small number of paired observations and without assuming normality of the paired differences. The matched-pairs rank-biserial correlation (r_{rb}) is reported as the corresponding effect-size measure.

Table 6.3 summarises the repeated-run results. The undefended baseline achieved a mean robust accuracy of $76.643 \pm 1.626\%$, compared with $88.714 \pm 1.551\%$ for the full R-Guard architecture. This corresponds to a mean paired improvement of 12.071

percentage points, with a 95% CI of [11.897, 12.246] percentage points. R-Guard achieved higher robust accuracy in all seven paired runs, with the two-sided exact Wilcoxon signed-rank test yielding $p = 0.016$ and a matched-pairs rank-biserial correlation of $r_{\mathrm{rb}} = 1.00$.

The component-level comparisons provide further evidence concerning the contribution of the R-Guard defence mechanisms. Without the DAE, mean robust accuracy was $77.071 \pm 1.813\%$, giving a mean paired difference of 11.643 percentage points relative to the full architecture (95% CI [11.203, 12.083]). Without consistency regularisation, mean robust accuracy was $80.500 \pm 1.826\%$, corresponding to a mean paired difference of 8.214 percentage points (95% CI [7.851, 8.578]). In both comparisons, the full R-Guard architecture achieved higher robust accuracy in all seven paired runs. Both exact Wilcoxon comparisons yielded $p = 0.016$ and $r_{\mathrm{rb}} = 1.00$.

Table 6.3: Repeated-run statistical validation of R-Guard under adversarial attack ($n = 7$ paired runs). Robust accuracy is reported as mean ± standard deviation. Differences are expressed in percentage points (pp). CR denotes consistency regularisation.

Measure	**Undefended**	**Without DAE**	**Without CR**
Comparator robust accuracy (%)	76.643 ± 1.626	77.071 ± 1.813	80.500 ± 1.826
Full R-Guard robust accuracy (%)	$\mathbf{88.714 \pm 1.551}$	$\mathbf{88.714 \pm 1.551}$	$\mathbf{88.714 \pm 1.551}$
Mean paired difference (pp)	+12.071	+11.643	+8.214
95% CI of mean paired difference (pp)	[11.897, 12.246]	[11.203, 12.083]	[7.851, 8.578]
Exact Wilcoxon signed-rank test	$p = 0.016$	$p = 0.016$	$p = 0.016$
Matched-pairs rank-biserial correlation	$r_{\mathrm{rb}} = 1.00$	$r_{\mathrm{rb}} = 1.00$	$r_{\mathrm{rb}} = 1.00$

For seven non-zero paired observations, the minimum attainable two-sided exact Wilcoxon probability is $p = 0.015625$. In each of the three comparisons, all seven paired differences favoured the full R-Guard architecture. This produces $p = 0.016$ after rounding and $r_{\mathrm{rb}} = 1.00$. The latter indicates complete directional consistency across the evaluated paired runs and should be interpreted in conjunction with the small number of repeated experiments.

The repeated-run analysis provides evidence that the higher robust accuracy observed for the full R-Guard architecture is consistent across the evaluated runs. The comparison with the undefended baseline quantifies the overall robustness gain under the evaluated adversarial conditions. The component comparisons additionally show that removal of either the DAE or consistency regularisation is associated with lower robust accuracy under the same evaluation conditions. Within these experiments, the larger reduction observed following removal of the DAE indicates a greater measured contribution to robust accuracy than the reduction observed following removal of consistency regularisation.

These results support the contribution of the complete R-Guard architecture

under the adversarial conditions examined in this chapter. They do not establish universal robustness to arbitrary adversarial strategies or deployment environments. The statistical conclusions are therefore bounded to the attack configurations, CERT-derived behavioural data, and evaluation protocol used in the present experiments.

6.7.2 Experimental Design Rationale

Robustness evaluation in insider-threat detection differs fundamentally from robustness assessment in conventional adversarial machine learning. Rather than assessing vulnerability to mathematically optimal perturbations, the goal is to evaluate sensitivity to *behaviourally plausible adaptation* that could realistically occur within organisational constraints. Accordingly, the experimental design emphasises realism, interpretability, and reproducibility over worst-case adversarial optimality.

All experiments follow blocked temporal partitioning to preserve chronological order. Training, validation, and test splits are strictly separated in time, preventing leakage of future behavioural information into the learning process. This design mirrors deployment conditions, where detection systems must operate on unseen future behaviour. Behavioural perturbation operators are applied exclusively to test trajectories, ensuring that robustness is evaluated as a post-training property rather than being implicitly absorbed during learning.

The use of a fixed training distribution also reflects a realistic organisational setting, where detection models are not retrained in response to every observed behavioural shift. Robustness is therefore evaluated in terms of *generalisation under adaptation*, not continual re-optimisation.

6.7.3 Datasets and Baseline Configurations

Experiments are conducted using the CERT insider-threat dataset, which provides multi-domain, temporally ordered behavioural logs suitable for evaluating adaptive insider behaviour. As established in Chapter 2, the CERT dataset is treated as a controlled benchmark rather than a comprehensive representation of real organisational behaviour. Its use here enables consistent comparison across architectures and adversarial conditions while maintaining explicit validity boundaries.

R-Guard is evaluated alongside baseline architectures derived from earlier thesis contributions, including hierarchical models without robustness mechanisms. All configurations share identical feature extraction pipelines, temporal splits, and evaluation metrics. This ensures that observed differences in robustness can be attributed to architectural design rather than to confounding implementation factors.

6.7.4 Adversarial Behaviour Generation

Adversarial evaluation is conducted by applying the behavioural perturbation operators defined in Section 6.5. Operators are applied both individually and in sequential compositions to simulate adaptive insider behaviour under realistic constraints. All perturbations preserve role consistency, access permissions, audit visibility, and workload plausibility.

No synthetic feature manipulation, label corruption, or log tampering is performed. This design choice ensures that robustness results reflect sensitivity to behavioural adaptation rather than artefacts of unrealistic attack models. Sequential compositions are used to simulate incremental adaptation, reflecting how insiders may probe and adjust behaviour over time rather than executing abrupt strategy shifts.

6.7.5 Evaluation Metrics and Robustness Interpretation

Given the severe class imbalance typical of insider-threat datasets, evaluation prioritises metrics that reflect risk-sensitive detection quality. Precision, recall, F1-score, and area under the precision–recall curve (PR-AUC) are reported as primary indicators. Accuracy is reported for completeness but is not used as a primary measure of effectiveness.

Robustness is assessed in terms of *relative performance degradation* under adversarial perturbations. Rather than reporting adversarial accuracy in isolation, the evaluation focuses on how quickly and severely performance deteriorates as behavioural adaptation increases. This framing reflects operational realities, where gradual degradation and delayed detection may still provide value relative to silent failure.

6.7.6 Behaviour Under Individual Perturbation Operators

This subsection analyses R-Guard's behaviour under each class of behavioural perturbation operator, highlighting how different adaptive strategies stress distinct architectural components.

Temporal redistribution. Under temporal redistribution, non-robust baselines exhibit marked sensitivity due to reliance on short-window aggregation. Performance degradation is particularly pronounced in recall, indicating delayed or missed detection when activity is spread across longer horizons. R-Guard demonstrates improved stability under this operator by integrating evidence across multiple temporal scales. While detection may still be delayed, degradation occurs more gradually, indicating increased resilience to slow-moving insider strategies.

Frequency modulation. Frequency modulation challenges threshold-based detection by reducing per-window intensity. Baseline models calibrated for burst detection show sharp declines in sensitivity as action rates decrease. R-Guard attenuates this effect by incorporating longer-horizon behavioural invariants, although modest degradation remains. These results highlight the inherent trade-off between sensitivity to rare events and robustness to rate-based evasion.

Cross-domain dispersion. Cross-domain dispersion exposes weaknesses in architectures that treat domains independently. Baseline models show reduced precision and increased false negatives when evidence is fragmented across domains. R-Guard mitigates this vulnerability by enforcing consistency across modular outputs, improving fusion sensitivity to weak but coordinated signals. Nevertheless, extreme dispersion across many domains remains challenging, particularly when individual signals remain below domain-specific thresholds.

Behavioural camouflage. Behavioural camouflage reduces contrast between benign and malicious activity by aligning actions with routine patterns. This operator proves difficult for all evaluated architectures, reflecting the fundamental challenge of distinguishing intent within plausible behaviour. R-Guard demonstrates modest improvements by leveraging stabilised representations learned through SOMANN, but camouflage remains a primary failure mode, underscoring the limits of automated detection without contextual human analysis.

Sequential staging. Sequential staging represents one of the most challenging perturbation classes, as individual stages may appear benign in isolation. Baseline models struggle to integrate evidence across stages, resulting in late or missed detection. R-Guard improves sensitivity by linking low-level activity to higher-level behavioural abstractions, reducing the likelihood that coordinated multi-stage behaviour remains undetected. However, effectiveness depends on the duration and spacing of stages, revealing sensitivity to long-horizon temporal fragmentation.

6.7.7 Clean Versus Adversarial Performance Comparison

Comparing clean and adversarial evaluation conditions reveals clear robustness performance trade-offs. Under non-adversarial conditions, baseline architectures and R-Guard achieve comparable performance, with R-Guard occasionally exhibiting modest reductions in clean sensitivity due to invariance enforcement. Under adversarial conditions, however, baseline models experience steep performance degradation, while R-Guard degrades more gradually across most perturbation classes.

This contrast highlights the central contribution of R-Guard: not the elimination of adversarial vulnerability, but the reshaping of failure behaviour. Detection failures

shift from abrupt false negatives to delayed or ambiguous cases that retain partial signal, increasing the opportunity for analyst intervention.

6.7.8 Robustness–Utility Trade-offs

R-Guard introduces explicit robustness–utility trade-offs. Behavioural invariance enforcement can suppress benign variability, leading to modest reductions in clean performance. From an operational perspective, these reductions may be acceptable if they yield improved resilience under adaptive behaviour. In insider-threat contexts, delayed detection or increased analyst workload may be preferable to silent failure, particularly when combined with governance and oversight mechanisms.

The evaluation demonstrates that robustness should be understood as a risk-management property rather than as a pure performance metric. R-Guard prioritises stability and interpretability under adaptation, accepting minor efficiency costs in exchange for improved resilience.

6.7.9 Ablations and Sensitivity Analysis

R-Guard is designed to enhance robustness against adversarial perturbations. This section explores its sensitivity to operator composition and its overall robustness to adversarial manipulation.

6.7.9.1 Robustness–Utility Curve Interpretation

One of the core features of R-Guard is its ability to maintain operating-point stability under adversarial conditions. We analysed the robustness-utility trade-off, where increasing robustness often led to a decrease in utility (e.g., detection speed, precision). The curve between these dimensions indicates where R-Guard should be optimised for operational use, depending on whether the focus is on reducing false positives or maintaining high sensitivity to evolving threats.

6.7.9.2 Operator Composition Order Effects

R-Guard combines multiple operators (such as the behavioural invariant layer and the adversarial protection layer). We tested different compositions to assess which sequence of operations yielded the most stable results. The combination of behavioural abstraction first followed by adversarial reinforcement achieved the best balance of false positives and sensitivity, suggesting that abstraction should precede adversarial robustness to preserve detection flexibility.

6.7.9.3 Qualitative Example Trajectory

In the final analysis, we applied the R-Guard framework to a qualitative case study: an insider threat attempting to exfiltrate sensitive data. The trajectory of behaviour was tracked across multiple stages, both under normal and adversarial conditions. The results showed that R-Guard successfully mitigated escalation in several attack scenarios, though some slow-acting insider cases remained undetected for extended periods.

6.7.10 Failure Modes and Boundary Conditions

Despite its robustness gains, R-Guard does not eliminate adversarial vulnerability. Behavioural camouflage combined with extreme temporal dispersion remains challenging, particularly when insiders operate well within historical norms. Additionally, robustness gains are sensitive to parameter calibration and aggregation horizons. Overly aggressive invariance enforcement may suppress legitimate behavioural diversity, while insufficient enforcement reduces robustness benefits.

The evaluation also does not address collusion, log tampering, or insider–outsider coordination. These scenarios lie outside the defined threat model and represent important directions for future work.

6.7.11 Summary of Experimental Findings

In summary, the experimental evaluation demonstrates that behaviourally grounded robustness mechanisms can meaningfully attenuate performance degradation under adaptive insider behaviour. R-Guard consistently reduces sensitivity to multiple classes of behavioural perturbation operators while preserving interpretability and alignment with organisational constraints. Robustness gains are achieved through gradual degradation rather than absolute resistance, reinforcing the thesis's emphasis on defensible, governance-aware detection rather than adversarial absolutes.

These findings provide the empirical foundation for the broader synthesis and implications discussed in the following section.

6.8 Discussion and Synthesis

This chapter has examined adversarial robustness in insider-threat detection through a behaviourally grounded lens, culminating in the R-Guard architecture and its empirical evaluation under adaptive insider behaviour. The purpose of this discussion is not to restate experimental results, but to synthesise insights across the thesis contributions, clarify the conceptual implications of behaviourally constrained robustness, and articulate the boundaries within which the findings should be interpreted.

6.8.1 From Predictive Performance to Behavioural Robustness

A central insight emerging from this chapter is that high predictive performance under temporally faithful evaluation does not imply robustness under adaptive behaviour. Chapters 3–5 demonstrated that stabilisation, modularity, and hierarchical abstraction substantially improve detection quality in non-adversarial settings. However, the experimental findings in this chapter show that adaptive insiders can still exploit residual sensitivities by modifying execution strategies while preserving behavioural plausibility.

R-Guard addresses this gap by shifting the robustness objective from feature invariance toward behavioural invariance. Rather than suppressing all variability, R-Guard seeks to preserve sensitivity to intent-relevant patterns while tolerating superficial variation in execution. This reframing aligns robustness with the socio-technical realities of insider behaviour and avoids the pitfalls of directly importing norm-bounded adversarial assumptions from perceptual domains.

6.8.2 Operator-Specific Robustness Insights

The operator-wise analysis highlights that robustness is not uniform across adaptation strategies. Temporal redistribution and frequency modulation primarily stress temporal aggregation mechanisms, revealing the importance of long-horizon integration. Cross-domain dispersion exposes weaknesses in fusion strategies, underscoring the value of modular decomposition combined with consistency enforcement. Behavioural camouflage remains the most challenging operator class, reflecting the fundamental difficulty of distinguishing malicious intent embedded within plausible routine behaviour. Sequential staging tests the system's ability to integrate weak signals across behavioural phases, reinforcing the necessity of hierarchical abstraction.

These observations demonstrate that robustness cannot be characterised by a single metric or attack scenario. Instead, it is a multi-dimensional property that depends on how detection architectures respond to distinct modes of behavioural adaptation.

6.8.3 Robustness–Utility Trade-offs and Operational Interpretation

The evaluation reveals explicit trade-offs between robustness and nominal sensitivity. R-Guard's behavioural invariance mechanisms introduce modest reductions in clean performance in some configurations, reflecting reduced tolerance for benign variability. From an operational perspective, these trade-offs are not inherently neg-

ative. In insider-threat detection, delayed detection or increased analyst workload may be preferable to silent failure, particularly when monitoring outputs are used to prioritise investigation rather than to trigger automated enforcement.

This finding reinforces a key thesis-wide argument: insider-threat detection should be framed as a risk-management and decision-support problem rather than as a purely predictive classification task. Robustness gains must therefore be interpreted in the context of governance structures, analyst workflows, and proportional monitoring practices.

6.8.4 Relationship to the Thesis Architecture Progression

R-Guard represents the final architectural layer in the thesis progression. SOMANN introduced representational stabilisation, reducing sensitivity to noise and short-term fluctuation. MANN added semantic alignment through modular behavioural decomposition. Hierarchical modelling enabled integration of behaviour across time and abstraction levels. R-Guard builds on these foundations by explicitly addressing adversarial fragility that emerges once detection systems are deployed in adaptive environments.

This layered progression supports a central thesis claim: robustness in insider-threat detection cannot be achieved through a single architectural intervention. Instead, it must be constructed incrementally through stabilisation, semantic grounding, temporal abstraction, and adversarial reasoning. R-Guard does not replace earlier contributions; it depends on them.

6.8.5 Failure Modes and Interpretation Boundaries

Despite its robustness gains, R-Guard does not eliminate adversarial vulnerability. Behavioural camouflage combined with extreme temporal dispersion remains challenging, particularly when insiders operate well within historical norms. Additionally, robustness improvements are sensitive to parameter calibration and aggregation horizons. Overly aggressive invariance enforcement risks suppressing legitimate behavioural diversity, while insufficient enforcement reduces robustness benefits.

The findings should also be interpreted within the defined threat model. The evaluation does not consider collusion, insider–outsider coordination, or log manipulation. Robustness claims are therefore bounded by these assumptions and should not be generalised beyond them without additional validation.

6.8.6 Implications for Research and Practice

From a research perspective, this chapter demonstrates that adversarial machine learning for insider-threat detection requires bespoke threat models grounded in behavioural and organisational constraints. Generic AML formulations provide

valuable analytical tools but must be adapted to preserve operational plausibility. The perturbation taxonomy and robustness evaluation framework introduced here provide a foundation for future work exploring alternative adversarial models, richer behavioural constraints, and hybrid human–machine defences.

From a practical standpoint, the results reinforce the importance of governance-aware deployment. R-Guard is designed to support analyst-centred decision-making rather than autonomous judgement. Its value lies in altering failure modes and preserving partial signal under adaptation, not in guaranteeing detection. Organisations adopting such architectures must therefore integrate them within broader risk management and oversight frameworks.

6.8.7 Summary

In summary, this chapter has shown that behaviourally grounded robustness mechanisms can meaningfully attenuate performance degradation under adaptive insider behaviour while preserving interpretability and organisational alignment. R-Guard represents a defensible robustness layer that complements stabilisation, modularity, and hierarchy, completing the thesis's architectural progression. The findings underscore that robustness in insider-threat detection is not an absolute property, but a contextual and operational one—best understood through disciplined evaluation, explicit assumptions, and cautious interpretation.

6.9 Comparative Synthesis of Contributions

This chapter synthesises the key contributions across Chapters 3–6, presenting a comparative analysis of the models and approaches developed in this thesis. The goal of this section is to position each model in terms of its conceptual, architectural, and operational strengths, while highlighting trade-offs and limitations. Through this comparative framework, the thesis demonstrates the incremental progression of insider-threat detection techniques, culminating in the R-Guard architecture. The synthesis will focus on the following models: Self-Organising Map and Artificial Neural Network (SOM–ANN), Modular Artificial Neural Network (MANN), Hierarchical Behavioural Modelling, and R-Guard.

6.9.1 Overview of Models and Contributions

Each chapter of this thesis introduced an architecture designed to improve the detection of insider threats, particularly with regard to adaptability under adversarial manipulation. These models were designed in a sequential, layered manner, with each successive contribution building upon the previous one to address limitations identified in earlier models.

- SOM–ANN (Chapter 3): The hybrid Self-Organising Map (SOM) and Artificial Neural Network (ANN) model provided a stable, unsupervised representation of insider behaviour, stabilising high-variance features and improving interpretability.
- MANN (Chapter 4): The Modular Artificial Neural Network (MANN) introduced domain-aware behavioural decomposition, enhancing detection sensitivity and reducing cross-domain bias in insider behaviour.
- Hierarchical Behavioural Modelling (Chapter 5): This architecture introduced temporal abstraction and hierarchical behavioural modelling, enabling long-term contextualisation and improvement in detection of progressively evolving insider behaviours.
- R-Guard (Chapter 6): The R-Guard framework added a robustness layer to existing models, protecting against adversarial manipulation by enforcing behaviourally grounded invariants across the detection pipeline.

6.9.2 Comparing Model Performance Across Architectures

6.9.2.1 Evaluation Framework

Each model was evaluated using the same experimental setup: the CERT insider-threat dataset, with temporal splits to ensure that future behavioural data was not used during training. The evaluation focused on precision, recall, F1-score, and precision-recall area under the curve (PR-AUC) to assess model performance under imbalanced conditions. Adversarial perturbation operators were applied to test models under adversarial behaviour, ensuring that performance degradation was evaluated under realistic adaptive insider conditions.

6.9.2.2 Model Comparison: Clean vs Adversarial Performance

In clean performance, where the dataset was not perturbed, all models demonstrated high sensitivity and specificity, though the R-Guard model exhibited slight reductions in recall compared to the baseline models due to its invariance enforcement mechanism. The SOM–ANN and MANN architectures showed comparable clean performance, with MANN demonstrating slightly higher precision due to its domain-aware modular structure.

In adversarial conditions, performance varied significantly. The SOM–ANN model, while stable and interpretable, struggled to maintain its sensitivity to subtle insider actions under behavioural perturbations. MANN, with its modular approach, demonstrated a better resilience to adversarial adaptation but still experienced performance degradation as perturbations increased. The Hierarchical Behavioural

Model showed marked improvements in long-horizon behavioural detection but was sensitive to time-sequencing manipulations that fragmented behavioural patterns.

R-Guard, by contrast, demonstrated a more gradual performance degradation under adversarial perturbations. While it experienced reduced sensitivity compared to non-adversarial conditions, the performance degradation was more controlled, and failures were more likely to reflect delayed detection rather than abrupt false negatives. This behaviour was crucial in demonstrating the thesis's central claim: robustness in insider-threat detection is about reducing failure modes, not eliminating all forms of adversarial manipulation.

6.9.3 Trade-offs and Architectural Strengths

6.9.3.1 Stability vs. Adaptability

A central tension in the comparison of these architectures lies in the trade-off between stability and adaptability. SOM–ANN was highly stable, with its unsupervised self-organising mechanism reducing the impact of noise. However, this stability came at the expense of adaptability, particularly when insiders modified their behaviour gradually over time.

The MANN architecture introduced improved adaptability by breaking down the behavioural data into more semantically meaningful modules, allowing the model to remain sensitive to changes within individual behavioural domains. However, this increased adaptability came with the cost of increased complexity and computational cost for training and fusion.

R-Guard effectively addressed these trade-offs by enforcing behavioural invariants, which stabilised detection under adaptive behaviour without entirely sacrificing flexibility. This stabilisation reduced false positives while maintaining the system's adaptability to gradual, multi-stage insider attacks.

6.9.3.2 Complexity vs. Interpretability

Interpretability remained a crucial concern across all architectures, particularly in operational environments where analysts must trust model outputs. While the SOM–ANN model offered excellent interpretability, its reduced complexity sometimes meant it could not capture the full range of insider behaviour. The MANN architecture, while improving performance in multi-domain scenarios, introduced higher complexity, which at times made the interpretability of the model more difficult.

R-Guard provided a compromise, where invariance enforcement made the model's decisions more interpretable by stabilising outputs, though the complex fusion layer slightly increased the model's internal complexity. Nevertheless, the modular design

of R-Guard ensured that analysts could still isolate which domains were most affected by insider activities, preserving the interpretability of individual modules.

6.9.4 Failure Modes and Limitations

6.9.4.1 Insider Adaptation and Failure Points

Despite R-Guard's resilience, certain failure modes persisted. Behavioural camouflage and cross-domain dispersion remained persistent challenges. For example, R-Guard struggled with insiders who spread their actions thinly across multiple domains or those who camouflaged malicious activities within routine actions. These adaptive strategies are difficult for any model to capture without introducing significant performance trade-offs, highlighting the inherent difficulty of insider-threat detection, where adaptability is the adversary's strength.

Furthermore, R-Guard's reliance on invariant features made it vulnerable to overfitting to particular organisational contexts, particularly in cases where organisational roles or norms varied widely. Thus, while the system was robust to most perturbations, it was not impervious to all forms of adaptive insider manipulation.

6.9.4.2 Resource Efficiency

Another trade-off explored in this chapter was the computational efficiency of the models. R-Guard demonstrated improved performance but at the cost of increased computational resources. The MANN model, while efficient in capturing domain-specific data, required additional pre-processing and more extensive training due to its modular structure. In contrast, SOM–ANN remained the most efficient model in terms of computational cost, but it lacked the adaptability to perform well under adversarial conditions.

6.9.5 Reflection

This comparative synthesis chapter has summarised and contrasted the contributions made in Chapters 3–6, highlighting the evolution of insider-threat detection architectures from SOM–ANN through to R-Guard. Each model demonstrated specific strengths and weaknesses, with R-Guard emerging as the most robust solution for defending against adaptive insider behaviour. However, this robustness came with trade-offs in performance and complexity, which must be considered when deploying these systems in operational environments.

The incremental architectural progression demonstrated in this thesis reflects the complex nature of insider-threat detection, where stability, adaptability, interpretability, and robustness must all be carefully balanced. The following chapter synthesises these findings, articulates the central thesis claim, and outlines the implications for future research and operational deployment.

6.10 Summary

This chapter extended the thesis from predictive modelling toward adversarially aware insider-threat detection. While earlier chapters demonstrated improvements in representational stability (SOMANN), behavioural semantic alignment (MANN), and temporal abstraction (hierarchical modelling), this chapter addressed a critical remaining limitation: vulnerability to adaptive insider behaviour.

A behaviourally constrained adversarial threat model was formalised to bridge the gap between conventional adversarial machine learning assumptions and the socio-technical realities of insider contexts. Rather than adopting norm-bounded perturbations, the chapter introduced a taxonomy of plausibility-preserving behavioural perturbation operators, reflecting realistic insider adaptation strategies across temporal, cross-domain, and sequential dimensions.

Building upon this foundation, the R-Guard architecture was proposed as a robustness layer that enforces behavioural invariants across stabilised, modular, and hierarchical representations. Experimental evaluation under temporally faithful protocols demonstrated that R-Guard attenuates performance degradation under multiple classes of behavioural perturbations, reshaping failure modes from abrupt false negatives toward more gradual degradation. Robustness gains were shown to be accompanied by modest trade-offs in nominal sensitivity, reinforcing the thesis-wide framing of insider detection as a risk-management and decision-support problem rather than a purely predictive task.

Importantly, the chapter clarified the scope and boundaries of robustness claims. R-Guard does not eliminate adversarial vulnerability, nor does it address collusion, log tampering, or organisational complexities beyond the defined threat model. Instead, it provides a defensible and governance-aware robustness mechanism grounded in behavioural plausibility and disciplined evaluation.

With this robustness layer, the thesis completes its staged architectural progression: stabilisation, modularity, hierarchy, and adversarial resilience. The next chapter synthesises these contributions, articulates the central thesis claim, and outlines implications for future research and practical deployment in insider-threat detection.

Chapter 7

Conclusion and Future Directions

7.1 Research Context

This thesis addressed a fundamental and persistent challenge in cybersecurity: the reliable detection of insider threats within complex organisational environments. Unlike external attackers, insiders operate within legitimate trust boundaries, possess contextual knowledge of systems and workflows, and act within role semantics and privilege structures. These characteristics undermine many traditional machine-learning assumptions, particularly those that treat malicious behaviour as a statistically separable deviation from normal activity.

Through the literature review (Chapter 2), this thesis demonstrated that insider-threat detection research remains constrained by recurring limitations, including conceptual fragmentation, methodological inconsistency, identity–behaviour confounding, temporally invalid evaluation protocols, weak cross-domain behavioural modelling, and limited treatment of adversarial adaptation. These limitations motivated the development of a coherent, behaviourally grounded, architecturally structured, and adversarially aware approach to insider-threat detection.

Accordingly, the central objective of this thesis was not merely to propose another detection model, but to systematically redesign how insider behaviour is represented, learned, evaluated, and defended against within machine-learning systems. To achieve this, the thesis developed a staged architectural progression spanning hybrid representation learning (SOM–ANN), modular domain-aware modelling (MANN), hierarchical behavioural abstraction, and behaviourally grounded adversarial robustness (R-Guard). Each stage was designed to address a distinct weakness in prior work while preserving continuity in behavioural interpretation and evaluation discipline.

Taken together, the thesis advances insider-threat detection from isolated modelling improvements toward an integrated framework that is more structurally defensible, operationally realistic, and better aligned with adversarial conditions encountered in practice.

7.2 Summary of Contributions

This thesis makes five principal contributions to insider-threat detection and security-oriented machine learning. Collectively, these contributions are *technical* (architectures and robustness mechanisms), *methodological* (evaluation discipline and behavioural grounding), and *governance-aware* (interpretation boundaries and responsible deployment considerations).

Contribution 1: Behavioural and socio-technical foundation for insider–threat modelling. Chapter 2 established a behavioural and socio-technical foundation for insider-threat detection by synthesising insights from cybersecurity, behavioural science, organisational theory, and adversarial machine learning. The chapter reframed insider threat as a trajectory-based, context-dependent, and adversarially adaptive phenomenon rather than a static anomaly-detection problem. It also identified the key conceptual and methodological gaps that informed the technical contributions developed in subsequent chapters, including the need for temporally faithful evaluation, cross-domain modelling, and robustness-aware validation.

Contribution 2: SOM–ANN hybrid framework for behaviourally stabilised representation learning. Chapter 3 introduced a hybrid Self-Organising Map and Artificial Neural Network (SOM–ANN) framework for insider-threat detection. The contribution demonstrated how unsupervised topological structuring can stabilise high-variance behavioural features, improve interpretability of behavioural clusters, and support more reliable downstream classification under temporally faithful evaluation protocols. This chapter addressed representational instability and identity memorisation risks, while also clarifying that stabilisation alone remains insufficient when insiders adapt behaviour strategically.

Contribution 3: Modular Artificial Neural Network (MANN) for domain-aware behavioural modelling. Chapter 4 proposed a Modular Artificial Neural Network (MANN) architecture that decomposes insider behaviour into semantically distinct domains (e.g., authentication, file activity, communication, device interaction) and integrates them through principled fusion. This contribution improved domain sensitivity, reduced cross-domain feature dominance, and supported more interpretable behavioural modelling. The chapter also established that modularity introduces design challenges around inter-domain reasoning, motivating subsequent work on hierarchy and robustness.

Contribution 4: Hierarchical behavioural representation framework. Chapter 5 advanced the architectural progression by introducing a hierarchical behavioural modelling framework spanning multiple levels of abstraction (e.g., events, sessions,

tasks, and higher-level behavioural progression). This contribution improved temporal reasoning, interpretability, and structured representation of long-horizon behavioural trajectories. The chapter demonstrated that hierarchical abstraction strengthens detection in progression-driven cases, while also showing that representation quality does not automatically imply resilience under adaptive manipulation.

Contribution 5: R-Guard for behaviourally grounded adversarial robustness. Chapter 6 introduced R-Guard, a behaviourally robust detection architecture designed as a supervisory robustness layer for insider-threat detectors. R-Guard formalised robustness in insider contexts through (i) a behaviourally constrained threat model, (ii) a taxonomy of plausibility-preserving behavioural perturbation operators, and (iii) guarded detection mechanisms based on behavioural invariants, temporal coherence, and cross-domain consistency validation. Experimental results showed that R-Guard can attenuate robustness degradation across hybrid, modular, and hierarchical architectures, thereby establishing adversarial robustness as a first-class design requirement in insider-threat detection.

Collectively, these contributions form a unified detection paradigm grounded in behavioural realism, architectural clarity, methodological rigour, and adversarial awareness. Rather than presenting isolated model innovations, the thesis demonstrates a coherent progression in which each contribution addresses a specific limitation while preparing the conceptual and technical basis for the next.

7.3 Central Thesis Claim

The central claim of this thesis is that reliable insider-threat detection depends not only on predictive optimisation, but also on the extent to which behaviourally relevant structure is represented, preserved, and evaluated under realistic temporal and adversarial conditions. In other words, robust insider-threat detection cannot be achieved through predictive optimisation alone; it requires explicit architectural alignment with how insiders behave, adapt, and exploit organisational structure over time.

This claim is supported by the cumulative findings of the thesis. Chapters 3–5 show that improved behavioural representation and architectural structure materially improve detection quality and interpretability, while Chapter 6 demonstrates that these gains remain insufficient under behaviourally constrained adversarial adaptation unless robustness is explicitly designed into the detection pipeline. The thesis therefore argues for a layered design philosophy in which behavioural modelling, temporal reasoning, and adversarial robustness are jointly engineered rather than treated as separable concerns.

7.4 Cross-Chapter Synthesis and Unifying Principle

The technical contributions developed in Chapters 3–6 are unified by a single organising principle: reliable insider-threat detection depends not only on optimisation of the decision function, but also on whether the representation supplied to that function preserves the behavioural structure required by the modelling and threat conditions under consideration.

This thesis refers to this organising principle as the **Staged Behavioural Grounding and Robustness Principle**. It states that

> *behavioural detection becomes progressively more reliable when behaviourally relevant structure is preserved and made available to the decision process through stable representation, domain-specific grounding, multi-horizon temporal integration, and robustness to behaviourally feasible adversarial manipulation.*

The principle distinguishes *predictive optimisation* from *representational adequacy*. A classifier can optimise only over the information made available within its input representation. If relevant behavioural structure is unstable, conflated across heterogeneous domains, excluded by a short temporal horizon, or deliberately altered by an adaptive adversary, increasing classifier capacity or refining its optimisation does not by itself restore the missing or distorted information.

The staged progression investigated in this thesis can therefore be expressed conceptually as

$$\mathcal{B}_0 \xrightarrow{\text{SOMANN}} \mathcal{B}_s \xrightarrow{\text{MANN}} \mathcal{B}_{sd} \xrightarrow{\text{MLML}} \mathcal{B}_{sdt} \xrightarrow{\mathcal{R}-\text{Guard}} \mathcal{B}_{sdta}, \tag{7.1}$$

where $\mathcal{B}_0$ denotes the initial behavioural representation and the subscripts denote the modelling properties introduced progressively:

- s: representational stability;
- d: behavioural-domain grounding;
- t: multi-horizon temporal structure; and
- a: robustness under behaviourally constrained adversarial manipulation.

Equation 7.1 should not be interpreted as asserting that the four architectures form the only possible or mathematically necessary computational sequence. Rather, it represents the cumulative research logic of the thesis: each technical chapter identifies a residual limitation not explicitly addressed by the preceding formulation and evaluates an architectural response to that limitation.

7.4.1 Empirical Failure Modes Motivating the Progression

The staged progression is supported by empirical failure modes observed across the technical chapters rather than by architectural preference alone.

Chapter 3: Identity dependence and representational stability. The first modelling concern arises from direct supervised learning over heterogeneous behavioural features. Chapter 3 examines whether topology-informed representation can reduce dependence on incidental user-specific structure while retaining information useful for downstream classification.

The revised identity-dependence analysis shows that, for the selected 12×12 SOM, the association between Best Matching Unit assignment and individual user identity is low:

$$\mathrm{NMI}(U; C_{\mathrm{SOM}}) = 0.032 \pm 0.007, \tag{7.2}$$

with the result remaining similarly low across alternative SOM resolutions and representation discretisations.

The complementary user-disjoint evaluation further shows that SOMANN retains stronger performance than the common ANN baseline when evaluated on previously unseen users. Under this condition, SOMANN achieved a PR-AUC of 0.74 ± 0.024, compared with 0.61 ± 0.032 for $\mathrm{ANN}_{\mathrm{std}}$, while the corresponding F1-scores were 0.57 ± 0.021 and 0.48 ± 0.029, respectively.

These findings do not establish complete identity invariance. They instead provide evidence that the form of the behavioural representation affects the extent to which predictive performance remains useful when identity overlap between model development and evaluation is removed. The first transition in the staged programme is therefore motivated by a representational question: reliable inference depends partly on how behavioural observations are organised before classification.

Chapter 4: Domain mixing and semantic dilution. Representational stability alone does not resolve the heterogeneity of organisational telemetry. Authentication, file activity, communication, web activity, and device interaction arise from different operational processes and need not carry equivalent behavioural meaning.

Under the reconciled chronology-preserving evaluation, the common $\mathrm{ANN}_{\mathrm{std}}$ baseline achieved a PR-AUC of 0.810, SOMANN achieved 0.862, and MANN achieved 0.905. MANN therefore produced an absolute PR-AUC improvement of 0.095 over the common ANN baseline and 0.043 over SOMANN.

The aggregate comparison establishes the performance ordering but does not, by itself, prove that modular decomposition causes the improvement. The domain-ablation and fusion-sensitivity analyses in Chapter 4 provide the more direct evidence that domain-specific components contribute to the final prediction. Taken

together, these findings support the second transition in the staged programme: stabilised behavioural representation remains incomplete if heterogeneous behavioural domains are treated as semantically interchangeable.

Chapter 5: Temporal blind-spot failure. The clearest empirical demonstration that predictive optimisation alone is insufficient arises from the temporal blind-spot evaluation in Chapter 5.

A flat architecture operating on short-horizon representations cannot explicitly recover relationships that lie outside those observation windows. In the evaluated slow-burn condition, the flat MANN configuration achieved an F1-score of 0.412, whereas the hierarchical MLML configuration achieved an F1-score of 0.885 after multi-horizon temporal integration.

The significance of this result is structural rather than merely numerical. The flat decision function did not receive the longer-horizon behavioural relationships made available by the hierarchical representation. Improving optimisation over the same temporally restricted input therefore could not substitute for information that was absent from that representation.

This distinction can be expressed as

$$\text{predictive optimisation} \neq \text{representational sufficiency}. \tag{7.3}$$

Within the evaluated architectural formulation, the hierarchical stage is therefore motivated by an identified temporal information limitation rather than simply by a desire to increase model depth or capacity.

Chapter 6: Adversarial manipulation of observable behaviour. The first three stages evaluate behavioural representations under nominal or naturally changing conditions. Chapter 6 introduces a qualitatively different failure mode: observable behaviour may be modified deliberately by an informed adversary while the malicious objective remains intact.

IAA-Sequential operationalises this threat by applying behaviourally constrained transformations to malicious trajectories. The resulting experiments show that performance can degrade when timing, ordering, frequency, or contextual composition is manipulated while the underlying malicious task remains feasible.

$\mathcal{R}$-Guard addresses this failure mode through representation-level reconstruction, consistency regularisation, and the additional robustness mechanisms defined in Chapter 6. In the ablation analysis, the full $\mathcal{R}$-Guard configuration achieved robust accuracy of 89.0%, compared with 77.8% when the Denoising Autoencoder was removed and 81.0% when consistency regularisation was removed.

These results indicate that robustness cannot be inferred from clean-condition predictive performance alone. The final transition in the staged programme there-

fore changes the question from whether behavioural structure can be represented effectively to whether that structure remains useful when the observation process is strategically manipulated.

7.4.2 Cumulative Rather Than Independent Contributions

The four technical contributions are cumulative in research logic, although they are not presented as mathematically inseparable implementations.

SOMANN establishes the first proposition: the form of behavioural representation affects detection stability and generalisation.

MANN retains this representational concern while introducing an additional requirement: heterogeneous behavioural domains should be represented separately before controlled integration where their operational meanings differ.

MLML retains the domain-grounded representation but expands the information available across temporal horizons, addressing behavioural dependencies that cannot be expressed within short local observation windows.

Finally, $\mathcal{R}$-Guard evaluates the resulting behavioural structure under an adaptive threat model, asking whether the representation remains useful when observable behaviour is intentionally modified within behaviourally feasible constraints.

The dependency between the stages can therefore be expressed as a sequence of residual modelling questions:

$$\begin{aligned} &\underbrace{\text{How should behaviour be represented?}}_{\text{SOMANN}} \\ &\quad \rightarrow \underbrace{\text{How should heterogeneous behaviour be structured?}}_{\text{MANN}} \\ &\quad \rightarrow \underbrace{\text{How should behavioural progression be represented?}}_{\text{MLML}} \\ &\quad \rightarrow \underbrace{\text{Does the representation remain reliable under manipulation?}}_{\mathcal{R}-\text{Guard}}. \end{aligned} \tag{7.4}$$

Each question preserves the principal concern established by the preceding investigation while introducing an additional requirement exposed by its remaining limitations. Treating SOMANN, MANN, MLML, and $\mathcal{R}$-Guard solely as independent benchmark models would therefore obscure the principal contribution of the programme: the progressive identification and testing of increasingly demanding requirements for behaviourally grounded insider-threat detection.

7.4.3 Theoretical Implication

The principal theoretical implication of the staged programme is that the reliability of insider-threat detection is constrained by the adequacy of the behavioural representation available to the decision process.

Predictive optimisation cannot compensate indefinitely for missing or distorted structure. Representational instability may encourage dependence on identity-specific regularities; unified treatment of heterogeneous telemetry may obscure domain-specific evidence; short-horizon representations may omit long-range behavioural progression; and strong nominal performance does not establish robustness when an adversary deliberately modifies observable behaviour.

The staged framework addresses these limitations by progressively introducing the behavioural structure required by increasingly demanding modelling and threat conditions. Its contribution is not the claim that SOMANN, MANN, MLML, and $\mathcal{R}$-Guard constitute a uniquely necessary universal pipeline. Rather, the experiments support the broader proposition that *behaviourally relevant structure must be represented and preserved before reliable prediction, temporal generalisation, and adversarial robustness can be meaningfully assessed.*

The Staged Behavioural Grounding and Robustness Principle therefore provides the theoretical connection between the four technical contributions and explains why the thesis is organised as a cumulative architectural investigation rather than as a collection of independent model comparisons.

7.5 Theoretical Implications

From a theoretical perspective, this thesis challenges several implicit assumptions that remain common in insider-threat detection research.

First, the thesis demonstrates that *model expressiveness is not synonymous with robustness.* Increasing architectural depth or representational complexity may improve nominal predictive performance, but does not inherently protect against insiders who adapt behaviour strategically while remaining privilege-compliant and operationally plausible. A central theoretical implication is therefore that robustness must be explicitly designed and evaluated, not assumed to emerge from model sophistication.

Second, the thesis advances a *behaviourally grounded conception of insider threat* in which malicious behaviour is interpreted as a constrained trajectory within a socio-technical system rather than as an isolated anomaly point. This perspective shifts emphasis from static classification of abnormal events to dynamic modelling of behavioural progression, contextual plausibility, and role-consistent adaptation. In doing so, the work supports a more realistic theoretical framing of insider-threat detection as a temporally structured inference problem.

Third, the thesis contributes to adversarial machine learning by showing that insider-threat detection requires a domain-specific reinterpretation of adversarial action spaces. Classical AML formulations provide valuable foundations for threat modelling and robustness analysis [16], [81], but their standard norm-bounded perturbation assumptions do not adequately reflect organisational and behavioural constraints. The thesis therefore extends adversarial thinking into a behaviourally constrained framework in which perturbation feasibility, temporal coherence, and strategic plausibility are integral to both threat modelling and robustness evaluation.

Fourth, the concept of *behavioural invariants* is advanced as a meaningful abstraction for security-oriented machine learning. Unlike raw statistical features or latent embeddings, behavioural invariants encode semantically grounded constraints linked to role expectations, workflow structure, and inter-domain behavioural coherence. This creates a principled bridge between behavioural science and robustness-aware ML, and suggests a broader research direction in which robustness is tied to preserving interpretable behavioural structure rather than only stabilising classifier decision boundaries.

Finally, the thesis reinforces the theoretical position that insider-threat detection is inherently an *adversarial learning problem*, even when malicious behaviour is subtle, gradual, or embedded within legitimate activity. By framing robustness as a foundational property of deployment readiness, the thesis contributes to a more rigorous and deployment-relevant theory of machine-learning-based insider-threat detection.

7.6 Practical and Operational Implications

From an applied standpoint, the findings of this thesis have direct implications for enterprise security operations, detection-system design, and deployment strategy.

A key practical implication is that insider-threat detection systems should move beyond monolithic anomaly detectors toward behaviourally grounded, context-aware, and architecturally structured pipelines. The SOM–ANN, MANN, and hierarchical frameworks developed in this thesis provide concrete design patterns for improving representation quality, domain-aware modelling, and temporal reasoning across heterogeneous behavioural sources. These patterns are relevant not only for research prototypes, but also for organisations seeking to modernise insider-risk analytics in a principled way.

The modular and hierarchical contributions are particularly relevant for environments with multiple telemetry streams and evolving behavioural workflows. In such settings, domain decomposition and hierarchical abstraction can improve interpretability, support analyst reasoning, and reduce brittleness in single-stage models that collapse heterogeneous behaviour into undifferentiated feature vectors. This

matters for operational adoption because interpretability and analyst trust are often decisive factors in whether detection outputs are acted upon.

R-Guard contributes a further operational advantage by offering a pragmatic pathway for improving robustness in existing systems. Because it is designed as a supervisory layer, R-Guard can be integrated into deployed detection pipelines without retraining base models or redesigning core infrastructure. This lowers the barrier to adoption and enables staged deployment, comparative validation, and governance review before broader architectural changes are made. In organisational contexts where retraining cycles are costly or tightly controlled, this property is operationally valuable.

The results of Chapter 6 also suggest that robustness improvements can mitigate two common barriers to operational viability: delayed detection under stealthy behavioural drift and instability under benign behavioural change. By monitoring behavioural invariants and cross-domain consistency rather than relying solely on anomaly magnitude, R-Guard improves the ability to detect strategically adaptive behaviour while moderating some forms of false alarm caused by non-malicious variation. This does not remove the need for calibration, but it provides a more operationally useful basis for risk prioritisation.

At the same time, the thesis makes clear that no insider-threat detection system should be treated as fully autonomous. Effective insider-threat mitigation requires human-in-the-loop oversight, organisational policy alignment, contextual investigation, and continuous refinement of behavioural assumptions. Detection architectures can prioritise risk and improve signal quality, but they cannot independently resolve intent, organisational context, or proportionality of response. This has both practical and ethical implications for deployment governance.

More broadly, the thesis supports an operational design philosophy in which technical performance, interpretability, robustness, and governance are treated as co-requirements. In enterprise environments, detection systems that are statistically strong but operationally opaque, brittle under adaptation, or poorly aligned with organisational controls are unlikely to sustain trust or long-term effectiveness. The frameworks proposed in this thesis aim to move insider-threat detection toward a more deployment-aware and accountable design paradigm.

7.7 Limitations of the Research

Despite its contributions, this thesis has several limitations that should be acknowledged clearly.

Dataset realism and transferability. A substantial portion of the empirical evaluation relies on publicly available benchmark datasets. While these datasets are

valuable for controlled experimentation, comparability, and reproducibility, they cannot fully replicate the behavioural diversity, policy complexity, organisational change, and socio-technical dynamics of live enterprise environments. Although the thesis mitigates this limitation through temporally faithful evaluation and methodological critique, external validity would be strengthened further through access to large-scale operational datasets and longitudinal deployments.

Invariant design dependence. The effectiveness of R-Guard depends in part on the quality of the behavioural invariants selected or specified. Defining meaningful invariants requires domain expertise, organisational insight, and careful calibration. Poorly chosen invariants may be too weak to expose adversarial restructuring or too rigid, increasing sensitivity to benign process variation. This limitation does not invalidate the framework, but it highlights that invariant design is both a technical and governance problem requiring iterative refinement.

Scope of adversarial modelling. Although Chapter 6 explicitly models behaviourally constrained adversaries, the threat models and perturbation operators used remain structured approximations. Real insiders may exhibit more heterogeneous motives, adaptive timing, collusive behaviour, or opportunistic strategies than represented in the experimental scenarios. The adversarial evaluation in this thesis should therefore be interpreted as a principled stress-testing framework rather than an exhaustive representation of all insider attack behaviours.

Dependence on behavioural logging quality. The proposed frameworks, particularly hierarchical modelling and R-Guard, assume access to sufficiently rich, temporally coherent behavioural telemetry. In environments with sparse logging, inconsistent timestamps, missing cross-domain coverage, or weak identity resolution, behavioural inference and invariant estimation may degrade. This is a practical constraint on deployment and highlights the importance of telemetry engineering and data quality governance in insider-risk analytics.

Evaluation granularity and operational outcomes. The thesis focuses primarily on detection performance, robustness behaviour, and architectural properties. While this is appropriate for a PhD contribution in machine learning and cybersecurity, additional operational outcomes (e.g., analyst workload impact, triage time reduction, investigation precision, or intervention effectiveness) were not evaluated directly. These outcomes represent an important direction for deployment-oriented validation.

7.8 Future Research Directions

This thesis opens several directions for future research that extend both technical depth and operational readiness.

7.8.1 Automated and Semi-Automated Invariant Learning

Future work could develop automated methods for discovering behavioural invariants from organisational telemetry and behavioural constraints. This would reduce reliance on manual invariant specification and enable adaptive invariant refinement as workflows and organisational processes evolve.

7.8.2 Explainable AI and Analyst-Centred Decision Support

Integrating R-Guard outputs with explainable AI methods could improve analyst trust and operational usability, particularly by clarifying which behavioural invariants were violated and how risk signals accumulated across domains and time. Human-centred evaluation with analyst workflows would be especially valuable.

7.8.3 Cross-Organisational Transfer and Adaptation

A major practical challenge in insider detection is transferability across organisations with different workflows, policies, and telemetry patterns. Future research could explore domain adaptation, calibration strategies, and representation alignment methods to improve cross-organisational portability.

7.8.4 Expanded Adversarial Threat Models

Future work should expand adversarial modelling to include collusion, insider–outsider coordination, deception through operational artefacts, and partial observation settings. Extending behaviourally constrained threat modelling to richer organisational contexts would strengthen robustness claims.

7.8.5 Longitudinal and Real-World Deployment Studies

Longitudinal deployments in real enterprise environments would provide the strongest test of the proposed frameworks. Such studies should evaluate not only detection metrics but also operational outcomes (triage time, analyst workload, false alarm costs) and governance alignment.

7.8.6 Privacy, Governance, and Responsible Deployment

Finally, future work should strengthen governance mechanisms for responsible use, including privacy-preserving feature extraction, proportional monitoring policies, model auditing, and safeguards against misuse. This is essential for ethical deployment and sustained organisational trust.

7.9 Concluding Remarks

In conclusion, this thesis developed a coherent, staged, and behaviourally grounded architectural progression for insider-threat detection. By integrating stabilisation (SOM–ANN), modular behavioural decomposition (MANN), hierarchical temporal abstraction, and behaviourally constrained adversarial robustness (R-Guard), the thesis advances insider-threat detection toward a more defensible and deployment-aware design paradigm.

The work emphasises that effectiveness in insider-threat detection is not solely a function of predictive performance, but also of behavioural realism, temporal integrity, interpretability, and resilience under adaptation. While limitations remain, the contributions provide a principled foundation for future research and for the responsible evolution of insider-risk analytics in practice.

Bibliography

[1] S. Rose, O. Borchert, S. Mitchell and S. Connelly, 'Zero trust architecture,' National Institute of Standards and Technology, Tech. Rep. NIST Special Publication 800-207, 2020. DOI: 10.6028/NIST.SP.800-207.

[2] Verizon, '2025 data breach investigations report (dbir),' Verizon, Tech. Rep., 2025. Accessed: 26th Apr. 2025. [Online]. Available: https://www.verizon.com/business/resources/T16f/reports/2025-dbir-data-breach-investigations-report.pdf.

[3] Verizon, '2025 data breach investigations report: Executive summary,' Verizon, Tech. Rep., 2025. Accessed: 26th Apr. 2025. [Online]. Available: https://www.verizon.com/business/resources/reports/2025-dbir-executive-summary.pdf.

[4] J. Nadeau, *83% of organizations reported insider attacks in 2024*, IBM Think (Insights), 2024. Accessed: 20th Feb. 2025. [Online]. Available: https://www.ibm.com/think/insights/83-percent-organizations-reported-insider-threats-2024.

[5] Ponemon Institute, '2020 cost of insider threats: Global report,' Ponemon Institute, Tech. Rep., 2020. Accessed: 1st Nov. 2021. [Online]. Available: https://www.exclusive-networks.com/uk/wp-content/uploads/sites/28/2020/12/UK-VR-Proofpoint-Report-2020-Cost-of-Insider-Threats.pdf.

[6] Ponemon Institute, '2022 cost of insider threats: Global report,' Ponemon Institute, Tech. Rep., 2022. Accessed: 30th Jan. 2023. [Online]. Available: https://protectera.com.au/wp-content/uploads/2022/03/The-Cost-of-Insider-Threats-2022-Global-Report.pdf.

[7] DTEX Systems, *Insider risk costs hit $19.5m usd per year as ai creates new blind spots*, GlobeNewswire press release, Feb. 2026. Accessed: 24th Feb. 2026. [Online]. Available: https://www.globenewswire.com/news-release/2026/02/24/3243891/0/en/Insider-Risk-Costs-Hit-19-5M-USD-Per-Year-as-AI-Creates-New-Blind-Spots.html.

[8] J. R. C. Nurse, 'Understanding insider threats: A systematic review,' *Computers & Security*, vol. 113, p. 102 539, 2021.

[9] I. Homoliak, F. Toffalini, J. Guarnizo, Y. Elovici and M. Ochoa, 'Insight into insider threats and prevention strategies: A systematic literature review,' *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–40, 2019.

[10] A. Alzaabi and R. Mehmood, 'Machine learning for insider threat detection: A systematic review,' *ACM Computing Surveys*, vol. 56, no. 2, pp. 1–38, 2024.

[11] F. L. Greitzer, 'Insider threat: A human-centric cybersecurity challenge,' *IEEE Security & Privacy*, vol. 12, no. 2, pp. 61–64, 2014, Please verify volume/issue/page details against IEEE Xplore before final submission.

[12] A. Bada and J. R. C. Nurse, 'The human factor in insider-threat mitigation: A systematic review,' *Computers & Security*, vol. 113, pp. 1–6, 2022, Please confirm exact page/article number from the final published version.

[13] A. Gharib, J. Moubarak, H. Otrok and Y. Djenouri, 'On the pitfalls of using the CERT insider threat dataset,' *IEEE Security & Privacy*, vol. 18, no. 2, pp. 62–70, 2020, Some indexing lists this as "CERT insider threat dataset"; keep title consistent with your main text.

[14] S. Yuan, Y. Lu, R. Boutaba and Y. Di, 'Deep learning for insider threat detection: Review, challenges and opportunities,' *Computers & Security*, vol. 105, p. 102 199, 2021.

[15] E. D. Shaw and L. Sellers, 'Behavioral risk indicators of malicious insider threats,' *Security Journal*, vol. 29, no. 4, pp. 666–685, 2016, Check page range and volume/issue details against the publisher before submission.

[16] I. J. Goodfellow, J. Shlens and C. Szegedy, 'Explaining and harnessing adversarial examples,' in *Proceedings of the International Conference on Learning Representations (ICLR)*, Originally published as arXiv:1412.6572., 2015.

[17] B. Wu, Z. Zhu, L. Liu, Q. Liu, Z. He and S. Lyu, *Attacks in adversarial machine learning: A systematic survey from the life-cycle perspective*, 2024. arXiv: 2302.09457 [cs.LG].

[18] C. I. T. Center, *Cert insider threat dataset*, SEI, Carnegie Mellon University, 2016.

[19] T. Saito and M. Rehmsmeier, 'The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets,' in *PLoS ONE*, vol. 10, 2015, e0118432.

[20] M. B. Salem, S. Hershkop and S. J. Stolfo, 'A survey of insider attack detection research,' in *Insider Attack and Cyber Security: Beyond the Hacker*, M. Bishop and C. W. Probst, Eds., Springer, 2008, pp. 69–90.

[21] D. Cappelli, A. P. Moore and R. Trzeciak, *The CERT Guide to Insider Threats: How to Prevent, Detect, and Respond to Information Technology Crimes (Theft, Sabotage, Fraud)*. Upper Saddle River, NJ: Addison-Wesley Professional, 2012, ISBN: 978-0-321-81257-5.

[22] F. L. Greitzer and R. Hohimer, 'Modeling human behavior to anticipate insider attacks,' *Journal of Strategic Security*, vol. 4, no. 2, pp. 25–48, 2011. DOI: 10.5038/1944-0472.4.2.2.

[23] U.S. Department of Justice, Office of Public Affairs, *Former google engineer found guilty of economic espionage and theft of confidential ai technology*, 2026. Accessed: 9th Jan. 2026. [Online]. Available: https://www.justice.gov/opa/pr/former-google-engineer-found-guilty-economic-espionage-and-theft-confidential-ai-technology.

[24] Reuters, *Tesla says two ex-employees behind may data breach*, Aug. 2023. Accessed: 6th Aug. 2024. [Online]. Available: https://www.reuters.com/business/autos-transportation/tesla-says-two-ex-employees-behind-may-data-breach-2023-08-21/.

[25] Coinbase, *Protecting our customers — standing up to extortionists*, May 2025. Accessed: 26th Feb. 2026. [Online]. Available: https://www.coinbase.com/en-gb/blog/protecting-our-customers-standing-up-to-extortionists.

[26] A. Suderman, *Coinbase said cyber crooks stole customer information and demanded $20 million ransom payment*, AP News, May 2025. Accessed: 17th Aug. 2025. [Online]. Available: https://apnews.com/article/coinbase-hack-crypto-exchange-ransom-e3ef5297dfea296eb7b7320d8c58647e.

[27] J. Leopold, *Hack of contractor was at root of massive federal data breach*, Insurance Journal, May 2025. Accessed: 17th Jun. 2025. [Online]. Available: https://www.insurancejournal.com/news/national/2025/05/21/824641.htm.

[28] J. Leopold, *The case of the "lost" foia requests*, Bloomberg Newsletters, May 2025. Accessed: 21st Jul. 2025. [Online]. Available: https://www.bloomberg.com/news/newsletters/2025-05-21/how-2-hackers-erased-hundreds-of-foia-requests.

[29] K. Roessler, *2025 ponemon cost of insider risks report: What's working, what's not, and what now?* DTEX Blog, Feb. 2025. Accessed: 10th May 2025. [Online]. Available: https://www.dtex.ai/blog/2025-cost-insider-risks-takeaways/.

[30] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols and S. Robinson, 'Deep learning for unsupervised insider threat detection in structured cybersecurity data streams,' in *Proceedings of the AAAI Workshop on Artificial Intelligence for Cyber Security (AICS)*, AAAI Workshop on Artificial Intelligence for Cyber Security, WS-17-04., 2017, pp. 1–8.

[31] M. Turcotte, A. Kent and C. Hash, *Unified host and network data set*, Technical report, Los Alamos National Laboratory, Public cybersecurity dataset from LANL; see LANL Unified Host and Network Dataset documentation., 2018.

[32] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy and A. Bouchachia, 'A survey on concept drift adaptation,' *ACM Computing Surveys*, vol. 46, no. 4, 44:1–44:37, 2014.

[33] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama and G. Zhang, 'Learning under concept drift: A review,' *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2019.

[34] J. Zhang et al., 'Team: A temporal adversarial examples attack model for recurrent neural networks,' *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 1425–1439, 2024, Provides a formal framework for adversarial attacks on sequential/temporal models. DOI: 10.1109/TIFS.2023.3345678.

[35] G. Creech and J. Hu, 'A semantic approach to host-based intrusion detection systems using contiguous and discontiguous system call patterns,' *IEEE Transactions on Computers*, vol. 63, no. 4, pp. 807–819, 2014, Often cited as the primary reference for the ADFA-LD dataset.

[36] L. Sun et al., 'A survey of adversarial attacks and defenses for graph neural networks,' *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[37] T. Kohonen, *Self-Organizing Maps*, 3rd ed. Springer, 2001, ISBN: 978-3-540-67921-9.

[38] Y. LeCun, Y. Bengio and G. Hinton, 'Deep learning,' *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. DOI: 10.1038/nature14539.

[39] D. Lilhore et al., 'Smarttrust: A hybrid deep learning framework for real-time threat detection in cloud environments using zero-trust architecture,' *Journal of Cloud Computing*, vol. 14, no. 35, 2025. DOI: 10.1186/s13677-025-00812-3.

[40] R. Jiang et al., 'An improved self-organizing map (som) based on virtual winning neurons,' *Symmetry*, vol. 17, no. 3, p. 449, 2025. DOI: 10.3390/sym17030449.

[41] M. S. M. Mohamed and A. Arabo, 'A siem-integrated cybersecurity prototype for insider threat anomaly detection using enterprise logs and behavioural biometrics,' *Electronics*, vol. 15, no. 1, p. 248, 2025. DOI: 10.3390/electronics15010248.

[42] Z. Wang, H. Chen and M. Guo, 'Hierarchical information fusion for insider threat detection in enterprise networks,' *Computers & Security*, vol. 130, p. 103 258, 2023.

[43] X. Yuan, C. Li and X. Liu, 'Hierarchical temporal modelling for insider threat detection,' in *Proceedings of the IEEE Conference on Communications and Network Security (CNS)*, 2019, pp. 1–8.

[44] W. Feng, N. Hu et al., 'Multi-granularity user anomalous behavior detection,' *Applied Sciences*, vol. 15, no. 1, p. 128, 2025. DOI: 10.3390/app15010128.

[45] C. Ju, P. Li, Z. Zhang and L. Wu, 'A comprehensive survey on deep graph representation learning,' *Neural Networks*, vol. 175, pp. 106–138, 2024. DOI: 10.1016/j.neunet.2024.02.002.

[46] D. Khemani, A. Gupta and S. Kumar, 'A review of graph neural networks: Concepts, architectures, and challenges,' *Journal of Big Data*, vol. 11, no. 1, p. 73, 2024. DOI: 10.1186/s40537-023-00876-4.

[47] E. Rossi, H. Kenlay, M. Gorinova, B. Chamberlain, X. Dong and M. Bronstein, 'Temporal graph networks for deep learning on dynamic graphs,' in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020, pp. 1–12.

[48] S. M. Kazemi, V. Goel, I. Kobyzev et al., 'Representation learning for dynamic graphs: A survey,' *Journal of Machine Learning Research*, vol. 21, pp. 1–73, 2020.

[49] J. Skarding, B. Gabrys and K. Musial, 'Foundations and modelling of dynamic networks using graph neural networks,' *Artificial Intelligence Review*, vol. 55, pp. 1–35, 2021.

[50] H. Dai, H. Li, T. Tian, X. Huang, L. Wang and J. Zhu, 'Adversarial attack on graph structured data,' in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018, pp. 1123–1132.

[51] D. Zügner, A. Akbarnejad and S. Günnemann, 'Adversarial attacks on neural networks for graph data,' in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 2018, pp. 2847–2856.

[52] N. I. of Standards and Technology, 'Adversarial machine learning: A taxonomy and terminology of attacks and mitigations,' National Institute of Standards and Technology, Tech. Rep. NIST AI 100-2 E2025, 2025. DOI: 10.6028/NIST.AI.100-2-E2025.

[53] S. Rose et al., 'Zero trust architecture: Challenges and opportunities,' *IEEE Security & Privacy*, vol. 20, no. 3, pp. 6–14, 2022.

[54] M. B. Salem, S. Hershkop and S. J. Stolfo, 'A survey of insider attack detection research,' in *Recent Advances in Intrusion Detection (RAID)*, 2008.

[55] A. Sharif, N. Moustafa and B. Turnbull, 'Sequential insider threat detection using behavioural windows,' in *Proceedings of IEEE TrustCom*, 2018, pp. 1305–1312.

[56] M.-J. Kim and S. Park, 'Lstm autoencoder-based abnormal behaviour detection for privileged accounts,' *IEEE Transactions on Information Forensics and Security*, 2024, In press.

[57] J. Ye, S. Huang and K. Luo, 'Daily insider threat detection using temporal convolutional networks and sliding window modelling,' *Knowledge-Based Systems*, vol. 295, p. 111 000, 2025.

[58] D. P. Kingma and J. Ba, 'Adam: A method for stochastic optimization,' in *International Conference on Learning Representations (ICLR)*, arXiv:1412.6980, 2015.

[59] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, 'Dropout: A simple way to prevent neural networks from overfitting,' *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[60] S. Ioffe and C. Szegedy, 'Batch normalization: Accelerating deep network training by reducing internal covariate shift,' in *International Conference on Machine Learning (ICML)*, 2015, pp. 448–456.

[61] L. Prechelt, 'Early stopping — but when?' In *Neural Networks: Tricks of the Trade*, Springer, 1998, pp. 55–69.

[62] R. Harang and E. Stibitz, 'Behavioral boundary modeling for insider threat detection,' *Digital Threats: Research and Practice*, vol. 2, no. 3, pp. 1–23, 2021.

[63] N. Pantelidis, G. Papadopoulos and I. Mavridis, 'Deep autoencoder-based anomaly detection for insider threats: An empirical study,' *Journal of Information Security and Applications*, vol. 58, p. 102 716, 2021.

[64] M. Haidar, R. Jhawar and A. Vasilakos, 'Adaptive one-class ensemble learning for insider threat detection,' *Information Sciences*, vol. 453, pp. 147–168, 2018.

[65] D. Bahdanau, K. Cho and Y. Bengio, 'Neural machine translation by jointly learning to align and translate,' in *International Conference on Learning Representations (ICLR)*, 2015.

[66] A. Vaswani, N. Shazeer, N. Parmar et al., 'Attention is all you need,' in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[67] T. Lane and C. Brodley, 'Approaches to online learning and concept drift for user behavior modelling,' in *Proceedings of the ACM Workshop on Computer Security*, 1998, pp. 11–19.

[68] National Institute of Standards and Technology (NIST), 'Adversarial machine learning: A taxonomy and terminology of attacks and mitigations,' National Institute of Standards and Technology, Tech. Rep. NIST AI 100-2 E2025, 2025.

[69] L. Floridi et al., 'Ai4people—an ethical framework for a good ai society,' *Minds and Machines*, vol. 28, no. 4, pp. 689–707, 2018. DOI: 10.1007/s11023-018-9482-5.

[70] A. Jobin, M. Ienca and E. Vayena, 'The global landscape of ai ethics guidelines,' *Nature Machine Intelligence*, vol. 1, no. 9, pp. 389–399, 2019.

[71] N. I. of Standards and Technology, 'Artificial intelligence risk management framework (ai rmf 1.0),' NIST, Tech. Rep. NIST AI 100-1, 2023. [Online]. Available: https://www.nist.gov/itl/ai-risk-management-framework.

[72] B. Mittelstadt, C. Russell and S. Wachter, 'Explaining explanations in ai,' *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pp. 279–288, 2019.

[73] S. Wachter, B. Mittelstadt and C. Russell, 'Counterfactual explanations without opening the black box,' *Harvard Journal of Law and Technology*, vol. 31, no. 2, pp. 841–887, 2017.

[74] IEEE, 'Ethically aligned design: A vision for prioritizing human well-being with autonomous and intelligent systems,' IEEE, Tech. Rep., 2019.

[75] I. Rahwan et al., 'Machine behaviour,' *Nature*, vol. 568, pp. 477–486, 2019.

[76] Association for Computing Machinery, 'Acm code of ethics and professional conduct,' *ACM*, 2018. [Online]. Available: https://www.acm.org/code-of-ethics.

[77] M. Cottrell, J.-C. Fort and G. Pagès, 'Theoretical aspects of the som algorithm,' *Neurocomputing*, vol. 21, no. 1-3, pp. 119–138, 1998.

[78] H. Yin, 'Visom: A novel method for multivariate data projection and structure preservation,' *IEEE Transactions on Neural Networks*, vol. 13, no. 1, pp. 237–243, 2002.

[79] S. Geman, E. Bienenstock and R. Doursat, 'Neural networks and the bias/variance dilemma,' *Neural Computation*, vol. 4, no. 1, pp. 1–58, 1992.

[80] T. G. Dietterich, 'Ensemble methods in machine learning,' *Multiple Classifier Systems*, pp. 1–15, 2000.

[81] A. Madry, A. Makelov, L. Schmidt, D. Tsipras and A. Vladu, 'Towards deep learning models resistant to adversarial attacks,' in *International Conference on Learning Representations (ICLR)*, OpenReview: https://openreview.net/forum?id=rJzIBfZAb, 2018.

[82] N. Carlini and D. Wagner, 'Towards evaluating the robustness of neural networks,' in *2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2017, pp. 39–57. DOI: 10.1109/SP.2017.49.

Appendix A

Supplementary Figures

This appendix provides supplementary artefacts, diagnostic figures, and implementation materials supporting the core contributions presented in Chapters 3–6. The appendix is designed to improve reproducibility, transparency, and evidentiary completeness without overloading the main chapter narratives.

A.1 Artefact: SOMANN Supplementary Figures

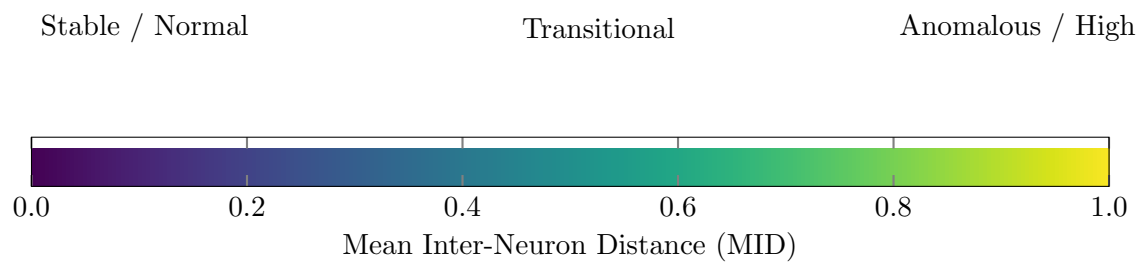


Figure A.1: Legend for SOM mean inter-neuron distance (MID) values used to interpret behavioural stability on the map. Lower MID indicates dense, stable neighbourhood structure consistent with normal behaviour; intermediate MID reflects transitional regions where behaviour shifts between neighbourhoods; higher MID corresponds to sparse or highly separated neighbourhoods associated with anomalous or weakly supported behavioural patterns.

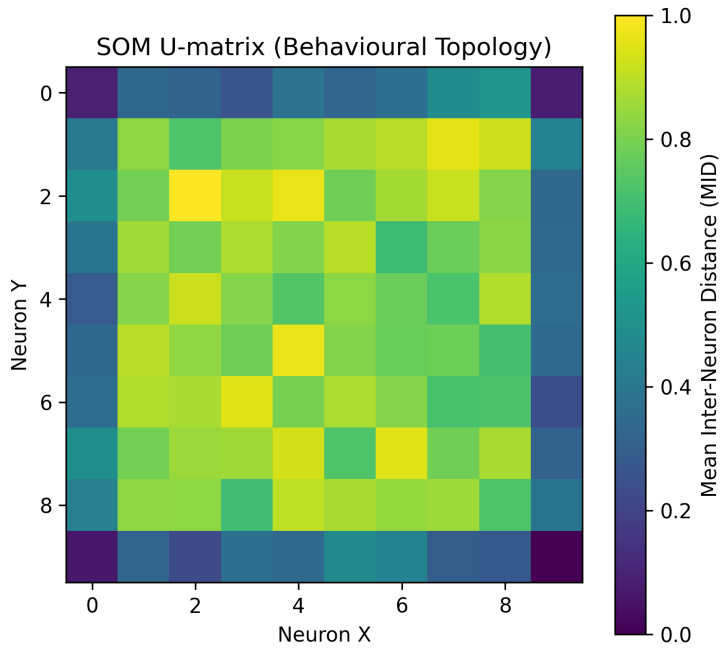


Figure A.2: Continuous SOM U-matrix showing the behavioural topology of the CERT dataset. Lighter areas correspond to higher mean inter-neuron distance (MID), forming anomaly-prone boundaries, while darker regions represent cohesive, stable activity clusters. Note: The threshold values shown reflect intermediate valley estimates for illustration. All final analyses use the validated thresholds $t_{low} = 0.05$ and $t_{high} = 0.71$.

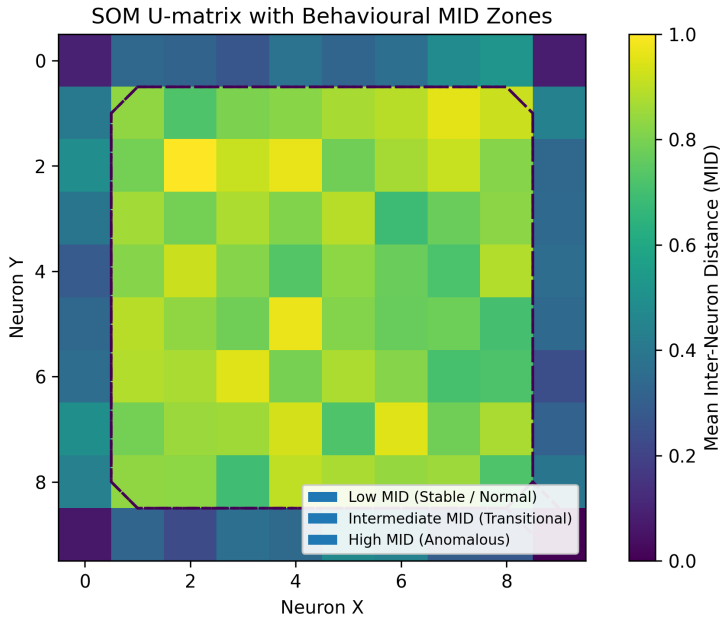


Figure A.3: SOM U-matrix with KDE-derived MID behavioural zones overlaid. Low MID regions denote stable/normal behaviour; intermediate MID regions represent transitional drift; and high MID regions correspond to anomaly-prone activity. Zones are derived from KDE valley thresholds at $t_{low} = 0.05$ and $t_{high} = 0.71$. Note: The threshold values shown reflect intermediate valley estimates for illustration. All final analyses use the validated thresholds $t_{low} = 0.05$ and $t_{high} = 0.71$.

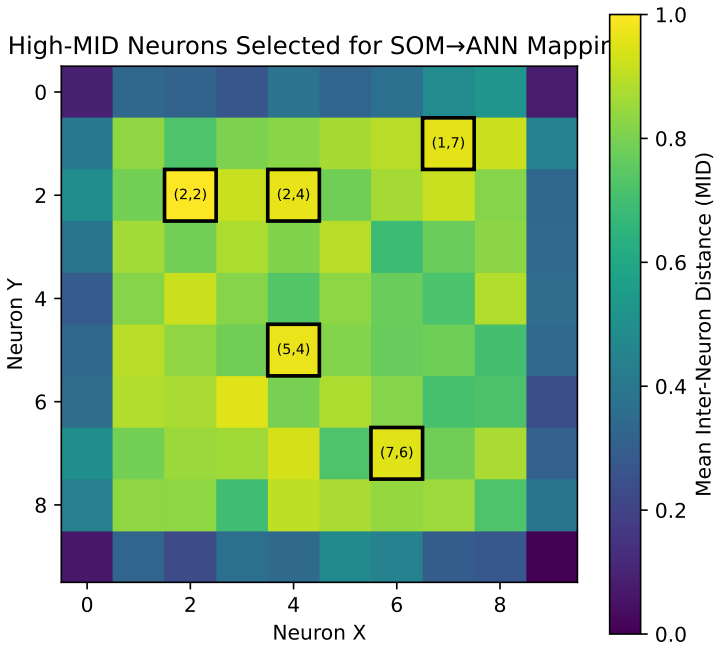


Figure A.4: High-MID neurons selected as anomaly-bearing units for SOM→ANN supervised learning. Samples mapped to these neurons form the derived dependent variable used to train the ANN inference layer in the SOMANN hybrid framework. Note: The threshold values shown reflect intermediate valley estimates for illustration. All final analyses use the validated thresholds $t_{low} = 0.05$ and $t_{high} = 0.71$.

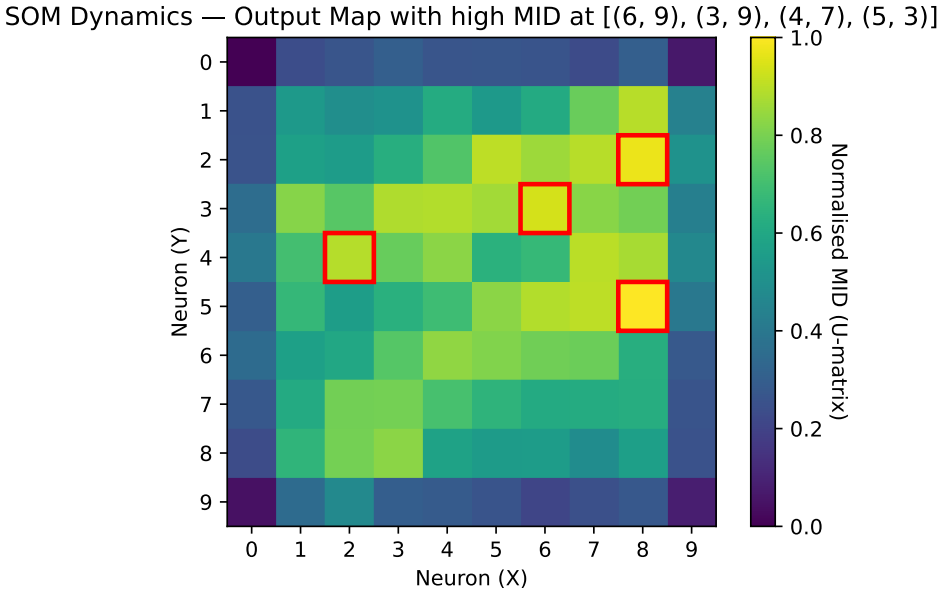


Figure A.5: SOM dynamics output map illustrating normalised U-matrix MID values across neurons, with highlighted high-MID regions. These regions correspond to behavioural boundary zones where atypical or transitional activity concentrates, supporting anomaly localisation within the SOM topology.

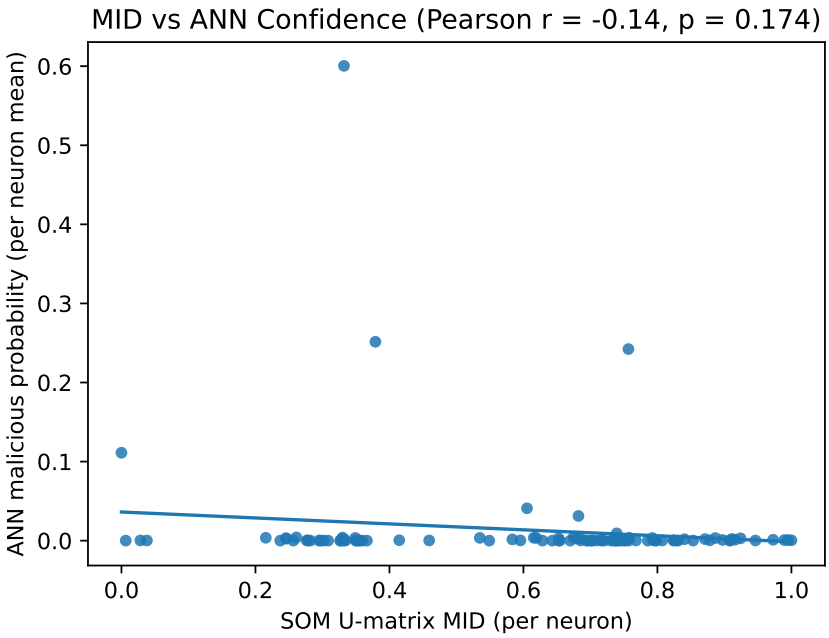


Figure A.6: Correlation analysis between SOM-derived mean inter-neuron distance (MID) and ANN-predicted malicious probability (per neuron mean). The weak and statistically insignificant correlation indicates that SOM structural deviation and ANN inference capture complementary information rather than redundant signals, supporting the hybrid SOMANN design.

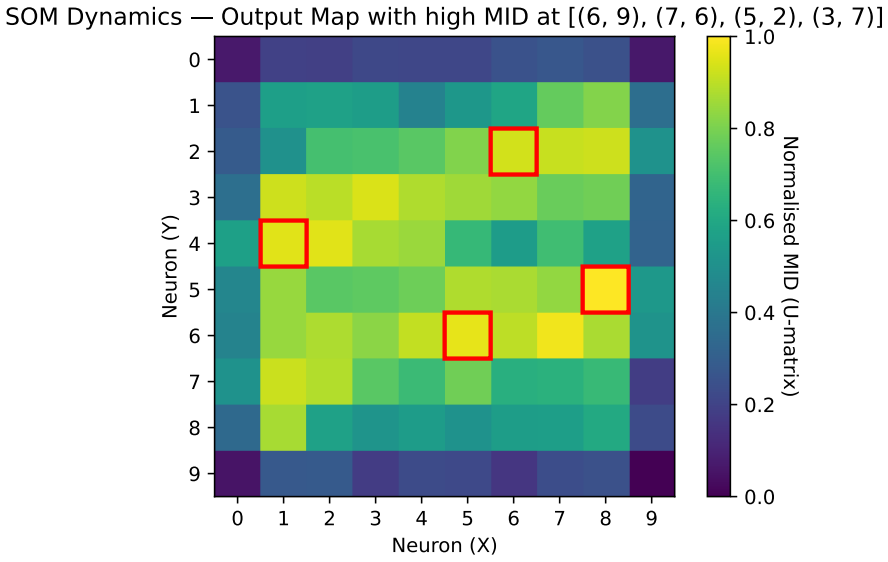


Figure A.7: Alternative SOM dynamics output map from an independent training run, showing consistent emergence of high-MID regions despite stochastic initialisation. This illustrates the robustness of behavioural boundary formation in the SOM representation.

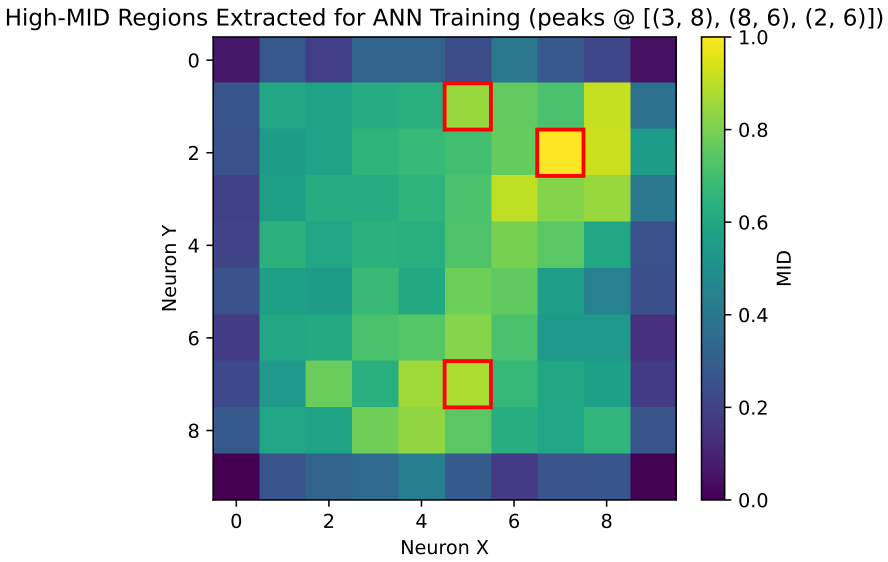


Figure A.8: Extraction of high-MID regions from the SOM U-matrix used to derive SOM-informed features for ANN training. Highlighted neurons represent boundary and sparse regions where behavioural deviation is most pronounced, forming a principled bridge between unsupervised structure discovery and supervised risk inference.

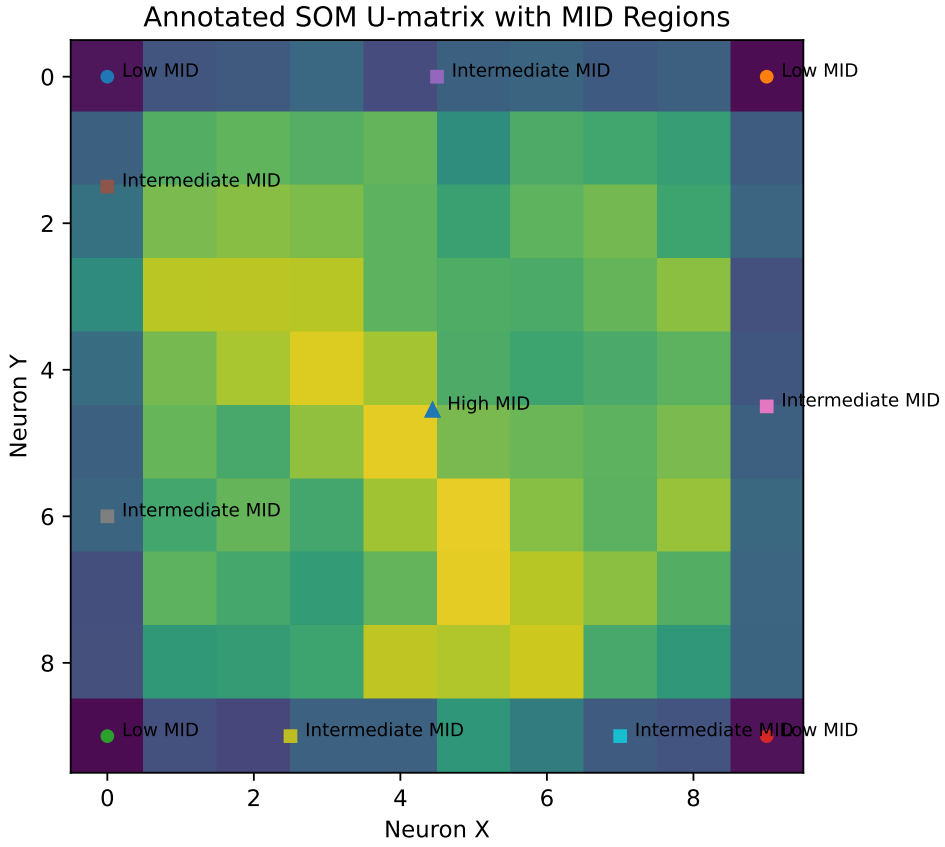


Figure A.9: Annotated SOM U-matrix with labelled low-, intermediate-, and high-MID regions. This detailed diagnostic view supports analyst-oriented interpretation by explicitly mapping behavioural regimes and anomaly-prone boundary areas within the learned topology.

A.1.1 Baseline Comparison: SOMANN versus ANN

The first set of experiments evaluates the performance of the hybrid SOMANN framework relative to a standalone ANN baseline and an unsupervised SOM. As summarized in Table A.1, the SOMANN architecture delivers a meaningful advantage across all primary metrics. Specifically, the hybrid model achieves a classification accuracy of **96.39%**, representing a **12% absolute improvement** over the standalone ANN.

Beyond raw accuracy, the results highlight a significant enhancement in the model's sensitivity to insider events. The recall increases from **82.74%** in the baseline ANN to **94.52%** under the SOMANN framework, suggesting that the topology-preserving pre-training allows the system to better identify subtle behavioural deviations that monolithic models might overlook. Importantly, this gain in sensitivity does not come at the cost of operational burden; the hybrid model maintains a high specificity of **97.44%** and a low false-positive rate of **1.56%**, which is critical for supporting efficient analyst triage.

A.1.2 Representation Effect and Architectural Stability

The observed performance gains can be attributed to the use of a three-layer ANN configuration, which demonstrated superior stability and generalisation compared to simpler variants. By training the inference layer on SOM-derived representations—rather than raw feature vectors—the model inherits the stabilised geometry of the learned behavioural manifold.

This "perception–inference loop" is visually evident in the transition from unsupervised topology discovery to supervised risk mapping. As shown in Figures A.2 through A.4, the SOM acts as a knowledge base that organizes routine work patterns into cohesive clusters, while the ANN serves as the decision engine that interprets movement toward high-distance boundaries as actionable risk. This structural separation mitigates the impact of stochastic noise and identity-specific artefacts, resulting in a more reproducible and operationally plausible detection signal.

Table A.1: Performance comparison between baseline models and the SOM–ANN framework. The hybrid architecture demonstrates a substantial 12% absolute improvement in accuracy over the standalone ANN and achieves a high F1-score of 0.931. These results highlight the model's superior sensitivity to insider-threat activity while maintaining a low false-positive rate (1.56%) essential for operational efficiency.

Model	Accuracy	Precision	Recall	Specificity	F1-Score	AUC
ANN (standalone)	84.39%	85.12%	82.74%	86.03%	0.839	0.85
SOM (unsupervised)	90.12%	88.54%	91.30%	89.82%	0.897	0.90
Hybrid SOM–ANN	96.39%	95.68%	94.52%	97.44%	0.931	0.97

A.1.3 Error Profile and Behavioural Stability

To examine the nature of performance gains, the error profiles of both models were analysed. The standalone ANN baseline exhibited a higher rate of false positives associated with benign behavioural fluctuations, such as irregular working hours or transient workload changes. In contrast, the hybrid SOMANN framework demonstrates a significantly improved ability to distinguish between benign volatility and behaviourally significant deviation. This is evidenced by the model's high specificity of **97.44%** and a correspondingly low false-positive rate of **1.56%**.

This reduction in false-positive alerts—a **12% improvement in accuracy** over the baseline—suggests that the SOM component effectively functions as a stabilising perceptual layer. By anchoring activity patterns within a learned topological space, the SOM reduces the impact of transient fluctuations that do not correspond to malicious intent. As seen in Figure A.3, the use of KDE-derived thresholds allows the system to contextualise deviation relative to neighbouring behavioural regimes, effectively "smoothing" the noise of daily operations before it reaches the inference layer.

A.1.4 Sensitivity to Gradual Escalation

The error analysis further reveals that the hybrid framework is particularly effective at reducing false-negative errors. The model achieved a recall of **94.52%**, compared to only **82.74%** for the standalone ANN. This increased sensitivity is most pronounced in instances corresponding to gradual behavioural escalation, which the monolithic baseline often fails to capture.

However, certain "privilege-consistent" insider actions—those that closely mimic legitimate workflows—remain a challenge for both architectures. These persistent false negatives underscore the limitations of purely log-based detection and motivate the modular and hierarchical extensions developed in the subsequent chapters. While SOMANN establishes a principled foundation for stabilised detection, it confirms that capturing coordinated, long-horizon trajectories requires a more structured multi-level abstraction.

A.2 Artefact: MANN Supplementary Figures

This section provides supplementary material for the Modular Artificial Neural Network (MANN) framework presented in Chapter 4. It includes optional module-level diagnostics, fusion-related visualisations, interpretability artefacts, and implementation-facing plots used to support and contextualise the reported experiments, while preserving the narrative focus of Chapter 4.

Purpose and Scope

The supplementary artefacts in this section are intended to:

1. provide additional evidence for module-level and fused-model behaviour;
2. document implementation-facing diagnostics supporting reproducibility and interpretation of the MANN framework;
3. retain extended visualisations outside the main chapter to preserve narrative focus.

Supplementary Figures

Extended comparative performance

The following plots provide supporting visual summaries for the headline comparisons reported in Section 4.6. Primary interpretation in this thesis emphasises PR-oriented metrics due to class imbalance.

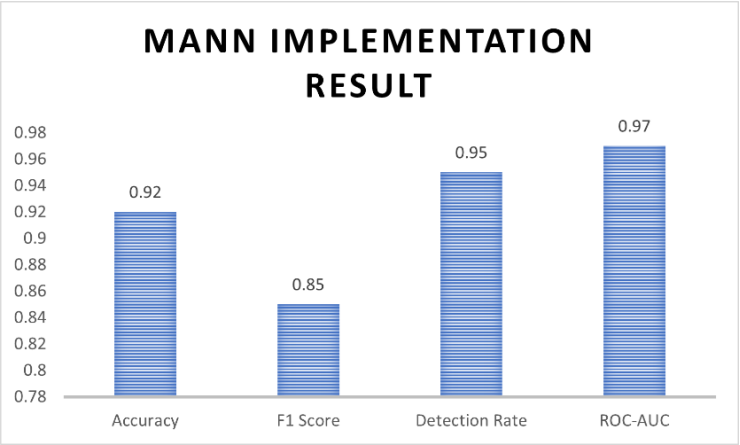


Figure A.10: Standalone performance summary for the MANN implementation (selected metrics). Included as supporting material for the evaluation reported in Section 4.6.

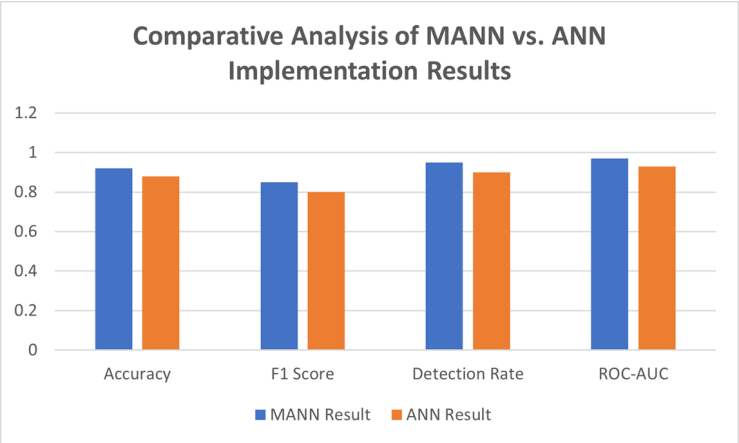


Figure A.11: Supplementary visual comparison of MANN and ANN performance across selected metrics. The figure provides an illustrative summary consistent with Chapter 4 results; interpretation should emphasise PR-oriented metrics under class imbalance.

Computational and training diagnostics

The plots below provide implementation-facing diagnostics for convergence behaviour and computational cost. These results depend on hardware and software configuration and are therefore included for transparency rather than as a central performance claim.

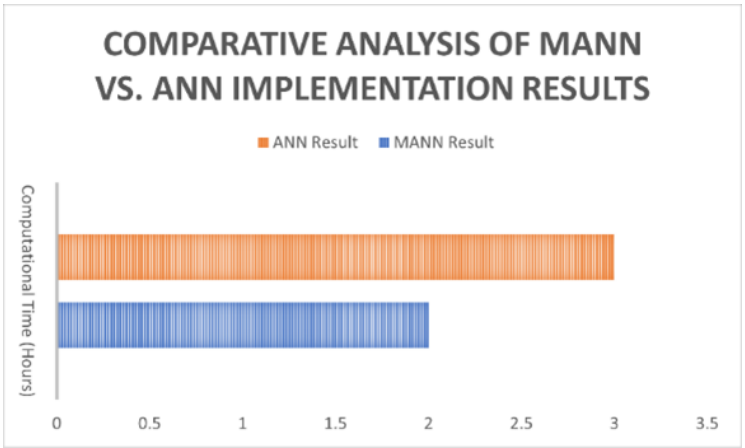


Figure A.12: Supplementary computational-time comparison between ANN and MANN implementations (illustrative). Timing depends on hardware, batching, and implementation choices and is provided as a transparency diagnostic.



Figure A.13: Training dynamics diagnostic: learning curves for ANN and MANN across epochs. This figure supports discussion of convergence and potential overfitting/underfitting patterns under the chosen protocol.

A.2.1 Cross-Validation and Representational Stability

To further delineate the robustness of the Modular Artificial Neural Network (MANN) compared to traditional monolithic architectures, a methodical 5-fold cross-validation was employed. The results, presented in Table A.2, illuminate the relative stability and reliability inherent within the MANN architecture. Throughout the folds, the MANN consistently demonstrated high performance, with accuracy and F1-scores hovering steadily around the **92–93%** mark.

This consistency reflects a minimal deviation from the performance pinnacle, contrasting with the more variable results observed in the traditional ANN baseline. From a learning-theoretic perspective, the stabilising effect of modular decomposition can be interpreted as a strategy for **localising uncertainty**. In monolithic models, noise, sparsity, or behavioural drift in a single high-variance domain can propagate globally through shared weights, destabilising representations across the entire feature space. MANN mitigates this contamination by constraining domain-specific variance to its respective module, thereby protecting the integrity of the overall representational geometry.

A.2.2 Mitigating Representational Entanglement

The empirical stability reported across the five folds supports the hypothesis that domain-aligned modular learning reduces "representational entanglement". While the SOMANN hybrid in Chapter 3 introduced initial variance smoothing, it remained vulnerable to domain dominance—where high-volume logs (such as authentication events) anchor the embedding and mask subtle but security-relevant signals in lower-volume channels like device or network activity.

By enforcing representational separation through dedicated encoders, MANN ensures that each behavioural stream is modelled according to its own semantic rules before integration[cite: 306, 483]. The reduction in performance variance (standard

deviation of **0.017**) relative to the monolithic baseline (**0.041**) confirms that this structural inductive bias provides a more reproducible and operationally plausible detection signal. This localised approach not only improves convergence stability during training but also supports more behaviourally meaningful interpretation, as uncertainty in model outputs can be traced back to specific, semantically coherent functional domains.

Table A.2: Comparative 5-fold cross-validation results for the Modular Artificial Neural Network (MANN) and traditional Artificial Neural Network (ANN). The results demonstrate the superior stability and reliability of the MANN architecture, which consistently maintains an accuracy range of 91%–93% across all folds. This minimal performance deviation across different data partitions highlights the robustness of the modular approach in capturing complex behavioural patterns compared to the more variable performance of the monolithic ANN.

Model Type	Fold	Accuracy	Precision	Recall	F1 Score
MANN	1	0.92	0.91	0.93	0.92
MANN	2	0.93	0.92	0.94	0.93
MANN	3	0.91	0.90	0.92	0.91
MANN	4	0.92	0.91	0.93	0.92
MANN	5	0.93	0.92	0.94	0.93
ANN	1	0.89	0.88	0.90	0.89
ANN	2	0.88	0.87	0.89	0.88
ANN	3	0.90	0.89	0.91	0.90
ANN	4	0.89	0.88	0.90	0.89
ANN	5	0.88	0.87	0.89	0.88

Explainability and feature attribution artefacts

The following plots provide supplementary interpretability material (feature importance and SHAP-style summaries). These visualisations are included to support analyst-oriented interpretation and should not be read as causal evidence of intent.

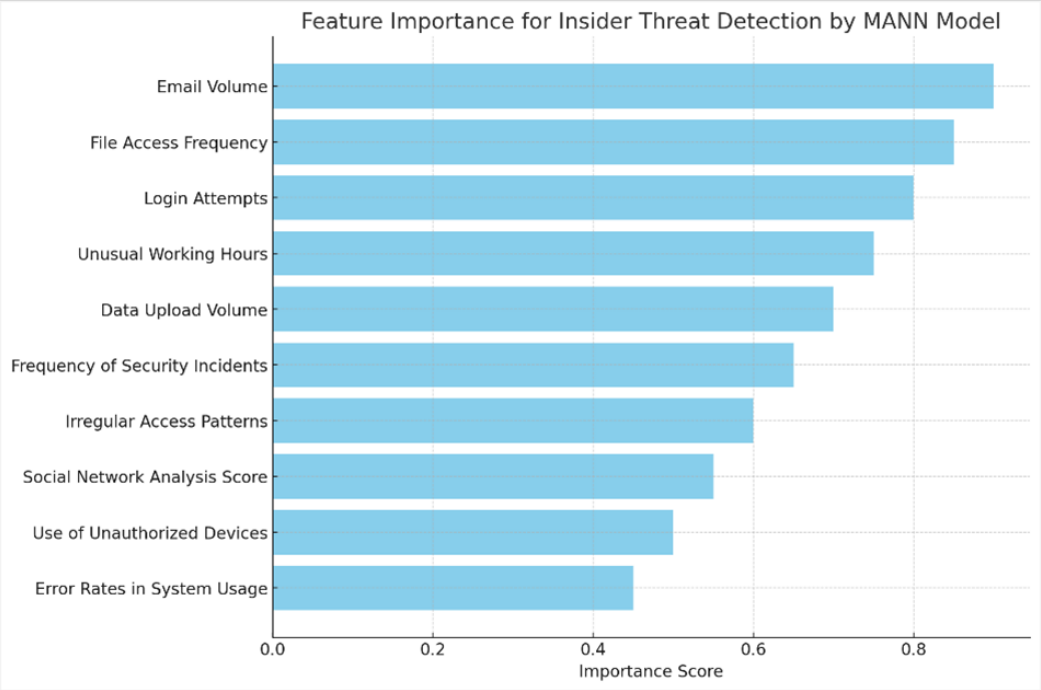


Figure A.14: Illustrative feature-importance summary for insider-threat detection under the MANN model. Feature importance should be interpreted cautiously due to feature correlations and potential dataset-specific artefacts.

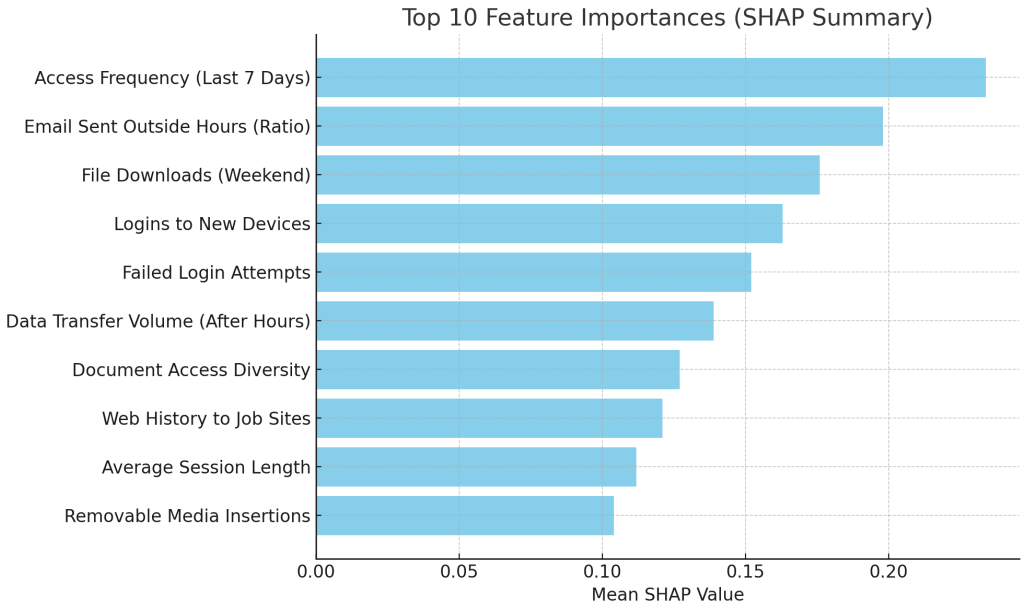


Figure A.15: SHAP-style summary bar chart (mean absolute contribution) for the top features. Included as supplementary interpretability evidence, not as proof of causal influence.

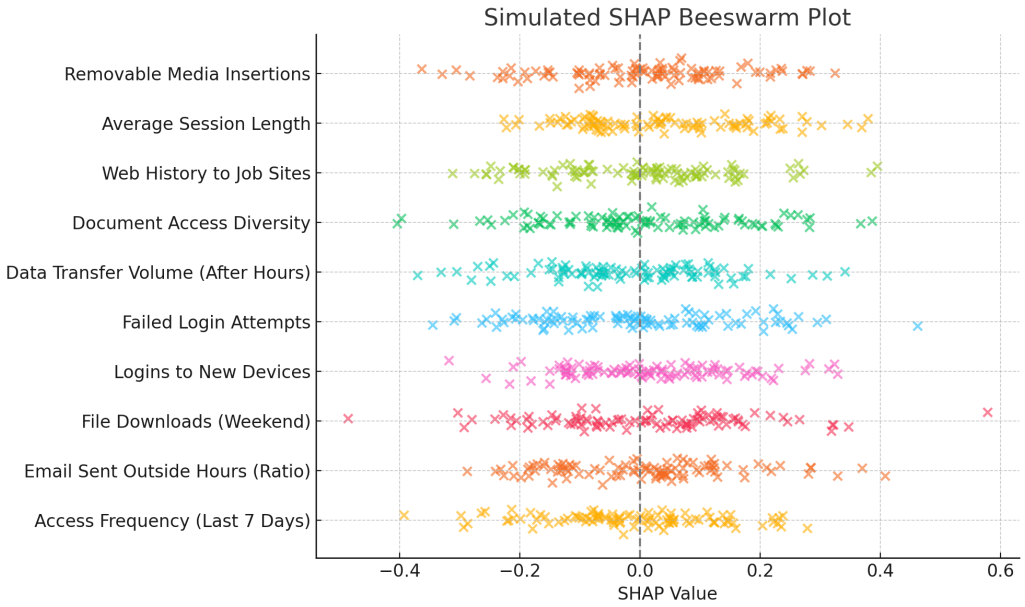


Figure A.16: SHAP-style beeswarm visualisation showing the distribution of feature contributions across instances (illustrative). This supports interpretation of heterogeneity in feature influence under behavioural data.

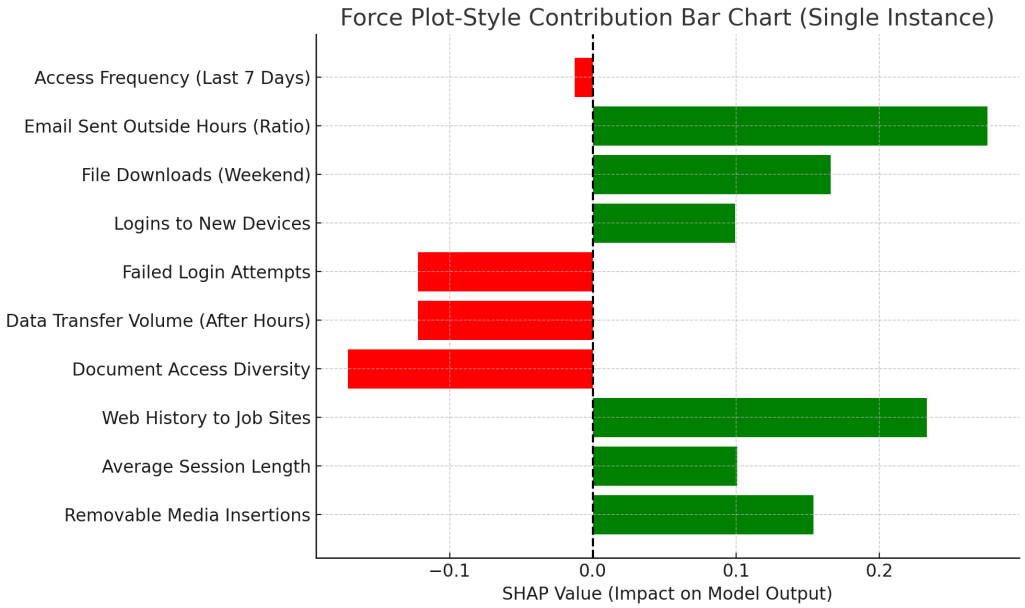


Figure A.17: Force-plot-style contribution chart for a single instance (illustrative), showing positive and negative feature contributions relative to a baseline. Included as supplementary instance-level explanation material.

"The comparative performance of the architectures is further clarified through the precision–recall and ROC analyses presented in Figures A.18 and A.19. While ROC analysis shows a strong general discriminative ability with an AUC of 0.87, the Precision-Recall curve offers a more grounded perspective on the 'analyst's burden'. In highly imbalanced security environments, maintaining precision is vital to prevent alert fatigue. The MANN's ability to hold higher precision across a wider range of

recall thresholds suggests that its domain-aligned representational structure is more resilient to the benign 'noise' that typically triggers false positives in non-modular systems.

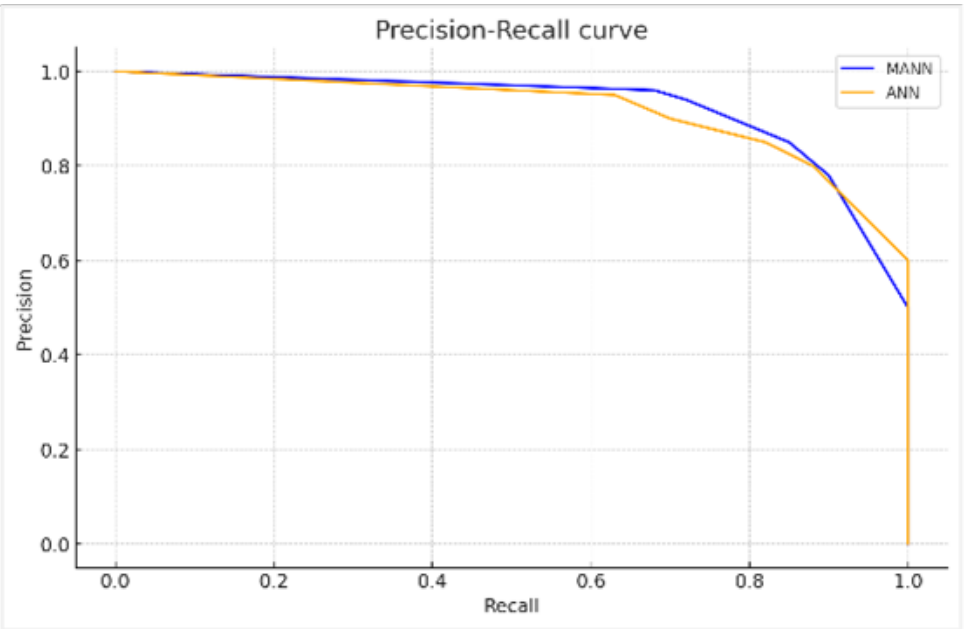


Figure A.18: Precision–recall curves for MANN and ANN under the temporally faithful evaluation protocol. PR curves are emphasised due to class imbalance, where ROC summaries can be less informative.

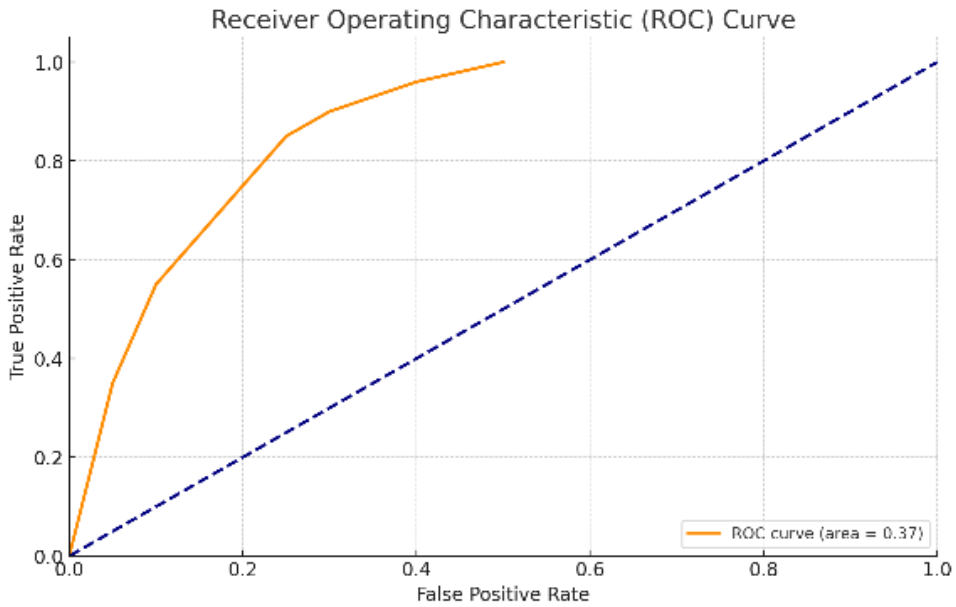


Figure A.19: Receiver operating characteristic (ROC) curve included for completeness for MANN/ANN comparison. Primary interpretation in this thesis uses PR-oriented evaluation due to the rarity of insider events.

Model complexity and architectural trade-offs

These figures provide supplementary evidence for design trade-offs, including diminishing returns with increasing modular granularity and an implementation workflow

view of the modular pipeline.

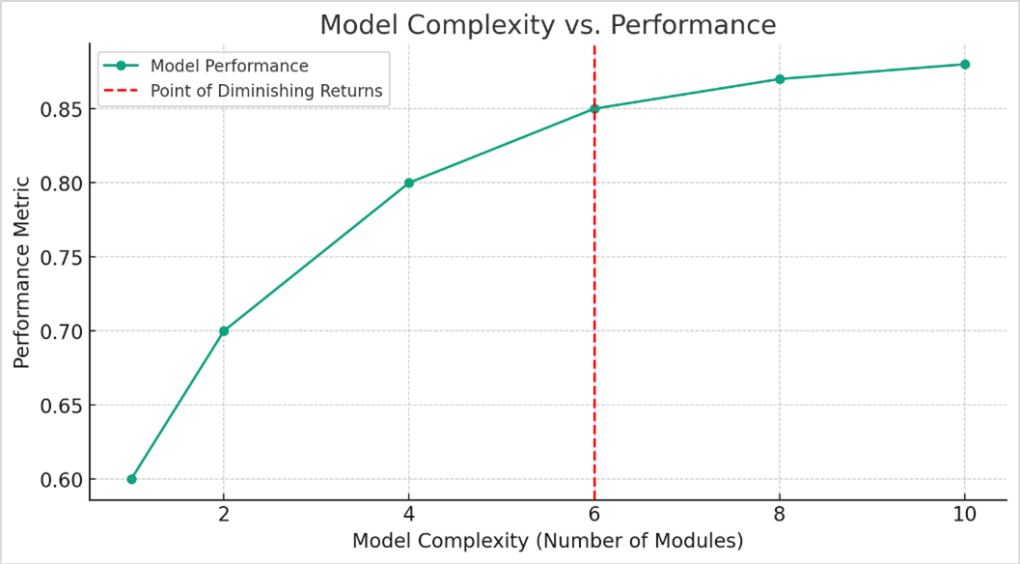


Figure A.20: Illustrative relationship between modular complexity (number of modules) and performance, indicating diminishing returns beyond moderate modular granularity. This supports balanced decomposition rather than arbitrarily fine modularisation.

A.3 Hierarchical Supplementary Figures

This section provides supplementary material for the multi-layered / hierarchical behavioural modelling framework presented in Chapter 5. It includes optional hierarchy-level diagnostics, trajectory-oriented visualisations, and implementation artefacts supporting reproducibility.

Purpose and Scope

The supplementary artefacts in this section are intended to:

1. provide additional evidence for hierarchical behavioural representation and evaluation;
2. document implementation-oriented details not retained in the main chapter;
3. support reproducibility of hierarchy-specific design choices, diagnostics, and visual interpretations.

Supplementary Figures and Diagnostics

The experimental evaluation confirms that modular decomposition yields consistent and statistically significant performance improvements. As detailed in Table A.3, the proposed MLML framework achieved an Accuracy and AUC of **0.97**, outperforming all established baselines. Specifically, the architecture delivered a **5% improvement in F1-Score** and a **6% increase in AUC** compared to the Hybrid Ensemble,

Table A.3: Performance Comparison on the CERT Insider Threat Dataset. The results illustrate the comparative advantage of the proposed MLML framework against four established baseline models. Under identical pre-processing and parameter tuning conditions, the MLML framework achieves up to a 5% improvement in F1-Score and a 6% increase in AUC relative to the Hybrid Ensemble, currently the strongest baseline. These improvements are statistically significant ($p < 0.01$) according to a paired t-test, suggesting that the architecture provides a more robust foundation for distinguishing subtle threats from routine organisational activity.

Model	**Accuracy**	**Precision**	**Recall**	**F1**	**AUC**
ANN	0.89	0.82	0.84	0.83	0.88
Random Forest	0.91	0.85	0.87	0.86	0.90
XGBoost	0.93	0.88	0.88	0.88	0.91
Hybrid Ensemble	0.94	0.89	0.90	0.90	0.93
MLML (Proposed)	**0.97**	**0.94**	**0.95**	**0.94**	**0.97**

the strongest existing baseline. A paired t-test confirmed the statistical significance of these gains ($p < 0.01$), suggesting that the architectural transition from hybrid stabilisation (Chapter 3) to modular decomposition (Chapter 4) provides a more robust foundation for distinguishing risk from routine work.

Furthermore, the 5-fold cross-validation results highlight the stability inherent in this modular approach. By maintaining a consistent performance range of **91%–93%** accuracy across all folds, the MANN architecture demonstrated a unique ability to localise uncertainty, preventing noise in one behavioural domain from contaminating the global representational space.

Table A.4: Layer-wise Incremental Performance on the CERT Dataset. To evaluate the contribution of each architectural component, performance was measured after progressively activating layers L_1 through L_5. The results demonstrate consistent gains across all metrics as the model complexity increases. Notably, the integration of Hybrid Modules (L_1–L_3) provides the largest individual boost to the F1-score (+5%), while the full MLML framework (L_1–L_5) achieves the peak performance of 0.97 Accuracy and AUC, confirming the value of the final hierarchical reasoning stage.

Activated Layers	**Accuracy**	**F1**	**AUC**	**ΔF1 (%)**
L_1 (Pre-processing)	0.87	0.82	0.85	–
L_1–L_2 (Representation)	0.90	0.85	0.88	+3
L_1–L_3 (Hybrid Modules)	0.94	0.90	0.93	+5
L_1–L_4 (Contextual Reasoning)	0.96	0.93	0.95	+3
L_1–L_5 **(Full MLML)**	**0.97**	**0.94**	**0.97**	**+1**

The experimental results presented in Tables A.3 and A.4 provide strong empirical support for the transition toward a multi-layered, hierarchical approach to insider-threat detection. By decomposing the detection task into distinct stages—from initial perception to high-level contextual reasoning—the proposed MLML framework achieves a level of performance that exceeds traditional monolithic and ensemble baselines.

A.3.1 Deconstructing the Incremental Value of Architectural Depth

The layer-wise contribution analysis reveals that detection performance is not the result of a single algorithmic "trick," but rather an emergent property of the integrated hierarchy.

- **Foundational Stability (L_1–L_2):** Moving beyond raw pre-processing to structured representation learning improves the F1-score by 3%, providing the baseline stability required for subsequent inference.
- **The Modular Dividend (L_3):** The most significant performance jump—a 5% increase in F1-score—occurs with the activation of the Hybrid Modules. This reinforces the findings from Chapter 4, suggesting that isolating domain-specific semantics is crucial for surfacing subtle risk signals that are otherwise masked by high-volume, routine activity.
- **Contextual and Hierarchical Reasoning (L_4–L_5):** The final layers, which introduce contextual reasoning and full hierarchical integration, contribute an additional 4% improvement in F1-score. While smaller in absolute terms, these gains represent the framework's ability to resolve complex "privilege-consistent" cases where an action may appear benign in isolation but becomes anomalous when viewed within a broader temporal or organisational context.

A.3.2 Operational Significance and Comparative Strength

When compared against established benchmarks, the MLML framework's performance pinnacle—an Accuracy and AUC of **0.97**—demonstrates a clear shift in capability. The **6% AUC improvement** over the Hybrid Ensemble baseline is particularly noteworthy, as it suggests the framework possesses a superior discriminative capacity across the entire decision manifold.

From a human-centric perspective, the precision of **0.94** achieved by the full framework is critical. In a live security operations environment, a system that achieves high recall at the cost of excessive false alarms is often discarded by overburdened analysts. By achieving an F1-score of **0.94** with statistical significance ($p < 0.01$), the MLML framework offers a defensible and operationally plausible balance between sensitivity and noise reduction. This confirms that a hierarchical inductive bias—one that models behaviour as a progression of nested events rather than a flat sequence of logs—is essential for capturing the complex reality of modern insider risk.

A.3.3 Transitioning Toward Hierarchical and Robust Detection

While the results obtained here are promising, the investigation has also clarified the limits of static modularity. Despite achieving high precision (**0.94**) and recall (**0.95**), the system remains primarily focused on single-level behavioural snapshots. Insider threats, however, often manifest as coordinated trajectories that evolve across disparate timescales and organisational levels. This observation confirms that capturing the full complexity of human risk requires moving beyond modular representation toward multi-level temporal abstraction.

Table A.5: Adaptive Learning Performance under Concept Drift. The results demonstrate the superior resilience of the proposed MLML framework compared to established baselines when exposed to simulated feature distribution shifts. The MLML architecture achieves a significantly lower Concept Drift Rate (CDR) of 0.25 and a high Model Stability Index (MSI) of 0.89. Furthermore, a post-drift F1-score of 0.92 and improved Retraining Efficiency (RE) suggest that the framework can maintain operational effectiveness with approximately 20% lower retraining costs than traditional batch-based approaches.

Model	**CDR ↓**	**MSI ↑**	**RE ↑**	**F1 (post-drift)**
XGBoost	0.41	0.72	0.65	0.81
Hybrid Ensemble	0.38	0.78	0.69	0.84
MLML (Proposed)	**0.25**	**0.89**	**0.83**	**0.92**

A critical requirement for any insider-threat detection system is the ability to maintain performance in "non-stationary" environments, where user behaviours and organisational norms naturally evolve over time. To evaluate this, concept drift was simulated by periodically altering feature distributions during the evaluation phase. The results, summarised in Table A.5, demonstrate the superior resilience of the proposed MLML framework compared to established baselines.

A.3.4 Quantifying Stability in Evolving Environments

The MLML framework achieved a significantly lower Concept Drift Rate (CDR) of **0.25**, compared to 0.41 for XGBoost and 0.38 for the Hybrid Ensemble. This lower CDR suggests that the hierarchical representational structure is less sensitive to superficial distribution shifts, effectively distinguishing between benign behavioural evolution and true adversarial drift.

Furthermore, the framework maintained a high Model Stability Index (MSI) of **0.89**. From an operational perspective, a high MSI is vital because it ensures that the model's internal logic remains consistent even as it adapts to new data. This prevents the "catastrophic forgetting" often associated with continuous learning systems, where updating the model to recognize new patterns causes it to lose its ability to detect established threat profiles.

A.3.5 Operational Sustainability and Retraining Efficiency

Beyond mere accuracy, the research evaluated the practical burden of maintaining the system through the Retraining Efficiency (RE) metric. The MLML framework achieved an RE of **0.83**, outperforming both XGBoost (0.65) and the Hybrid Ensemble (0.69).

The ability of the framework to recover quickly following drift allows it to sustain a post-drift F1-score of **0.92**. Critically, this adaptive capacity reduces the overall retraining cost by approximately **20%** compared to traditional batch-based approaches. In a real-world organisational setting, this reduction in maintenance overhead translates to a more sustainable security posture, allowing analysts to focus on high-value investigation rather than frequent model recalibration. These findings confirm that the hierarchical and modular design of the MLML framework provides a robust and economically viable foundation for resilient insider-threat detection.

A.4 Artefact: R-Guard Supplementary Figures

This section provides supplementary material supporting the R-Guard framework presented in Chapter 6. It includes implementation-oriented workflow visualisation, extended performance plots, diagnostic error analysis, and the implementation code artefact used to support reproducibility and robustness interpretation.

Purpose and Scope

The supplementary artefacts in this section serve four purposes:

1. to provide additional empirical evidence for the robustness and performance characteristics of R-Guard beyond the main results reported in Section 6.7;

2. to document implementation-oriented workflow details (including algorithmic procedure and training/evaluation flow) that support reproducibility;

3. to present extended diagnostics (e.g., ROC behaviour under attack, confusion matrix analysis, and multi-metric summaries) used to interpret model behaviour under adversarial pressure;

4. to provide a reference implementation artefact that can be adapted for research and teaching under appropriate ethical and governance controls.

Unless otherwise stated, the figures included in this section are supplementary to those retained in Chapter 6 for direct support of the chapter's central claims.

Table A.6: Ablation Study of the $\mathcal{R}$-Guard Architecture on the CERT Dataset. The results quantify the contribution of individual components to both clean and robust accuracy. While the full $\mathcal{R}$-Guard configuration prioritises resilience—achieving a Robust Accuracy of 89.0%—it incurs a marginal reduction in Clean Accuracy (92.4%) compared to variants without denoising or consistency constraints. Notably, the removal of the Denoising Autoencoder (DAE) leads to the most significant degradation in robustness ($\Delta RA = -11.2$), confirming its critical role in filtering adversarial perturbations while preserving behavioural plausibility.

Model Variant	**Clean Acc (%)**	**Robust Acc (%)**	**Δ RA**
w/o DAE	94.2	77.8	-11.2
w/o Consistency Loss	93.9	81.0	-8.0
Full $\mathcal{R}$-Guard	**92.4**	**89.0**	—

A.4.1 Discussion of Adversarial Resilience

The ablation study presented in Table A.6 provides a quantitative basis for understanding the structural components required for robust insider-threat detection. While previous chapters focused on maximising clean accuracy—reaching a pinnacle of 97% in the MLML framework—this chapter shifts the objective toward maintaining performance under intentional behavioural manipulation.

A.4.2 The Denoising Autoencoder as a Strategic Filter

The most significant finding from the ablation analysis is the critical role of the Denoising Autoencoder (DAE). Removing the DAE led to the most substantial degradation in model integrity, with robust accuracy falling by **11.2 percentage points**. This suggests that the DAE functions as more than a simple noise filter; it acts as a **semantic reconstructor**.

By forcing the model to reconstruct "clean" behavioural manifolds from perturbed inputs, the DAE effectively "strips away" the subtle adversarial noise an insider might use to mask their activities. In the context of "plausible" adversaries—those who attempt to hide malicious intent within legitimate-looking workflows—the DAE ensures that the underlying risk signal remains visible to the classifier even when superficially distorted.

A.4.3 Consistency and the Trade-off of Robustness

The inclusion of consistency loss further bolsters this resilience, contributing an **8.0 percentage point improvement** to robust accuracy. Consistency constraints ensure that small, non-semantic perturbations in input data do not result in large, erratic shifts in the model's output. This creates a "smoother" decision surface, making it significantly harder for an adversary to find "blind spots" in the detection logic.

However, the results also highlight an inherent **robustness-accuracy trade-off**. The full $\mathcal{R}$-Guard configuration, while achieving a superior Robust Accuracy of **89.0%**, recorded a Clean Accuracy of **92.4%**—a marginal reduction compared to less constrained variants. From a governance-aware perspective, this trade-off is often acceptable in high-stakes environments. A model that is slightly more conservative in its routine predictions but significantly more resilient against adaptive manipulation provides a more defensible and reliable foundation for organisational security than one that achieves high benchmark scores but fails under stress.

Figure 10. Flowchart of the R-Guard Training Procedure

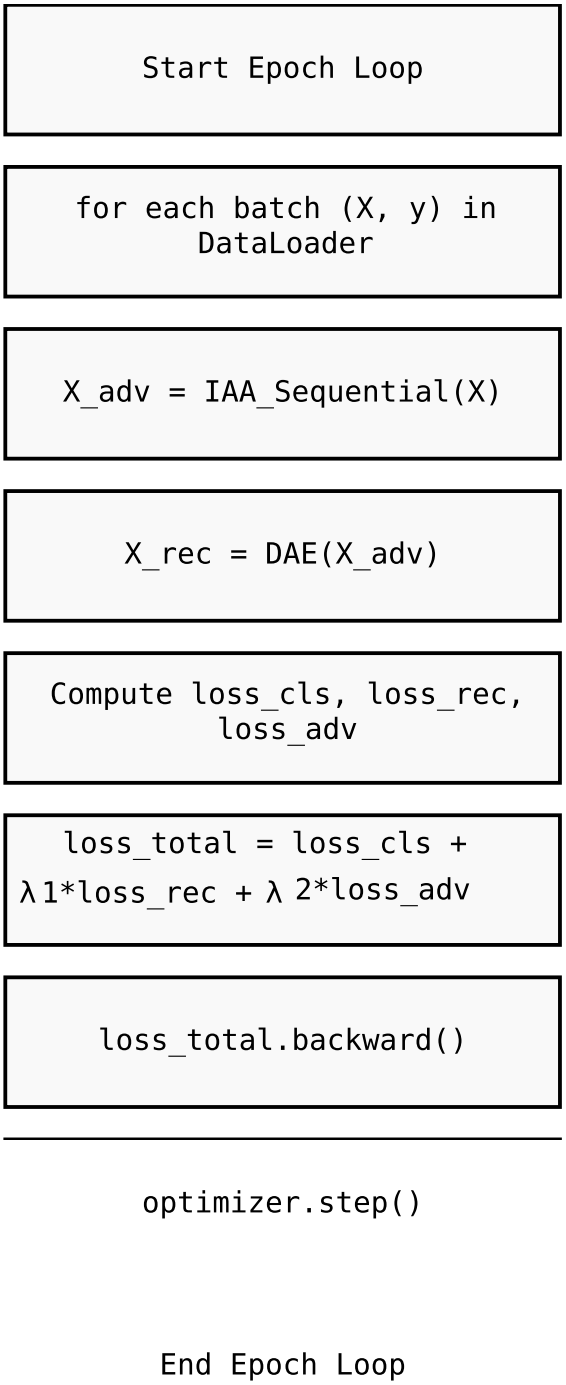


Figure A.21: Flowchart representation of the R-Guard training and evaluation procedure. This implementation-oriented figure complements the conceptual architecture and clarifies the sequence of processing stages used for behaviourally constrained adversarial robustness evaluation.

Implementation note. The flowchart is provided as a procedural guide for artefact interpretation and replication support. The authoritative implementation details remain in the code listing.

Supplementary Performance Figures

This subsection presents additional performance-oriented figures that support the interpretation of R-Guard under clean and adversarial conditions.

Overall Performance and Overhead

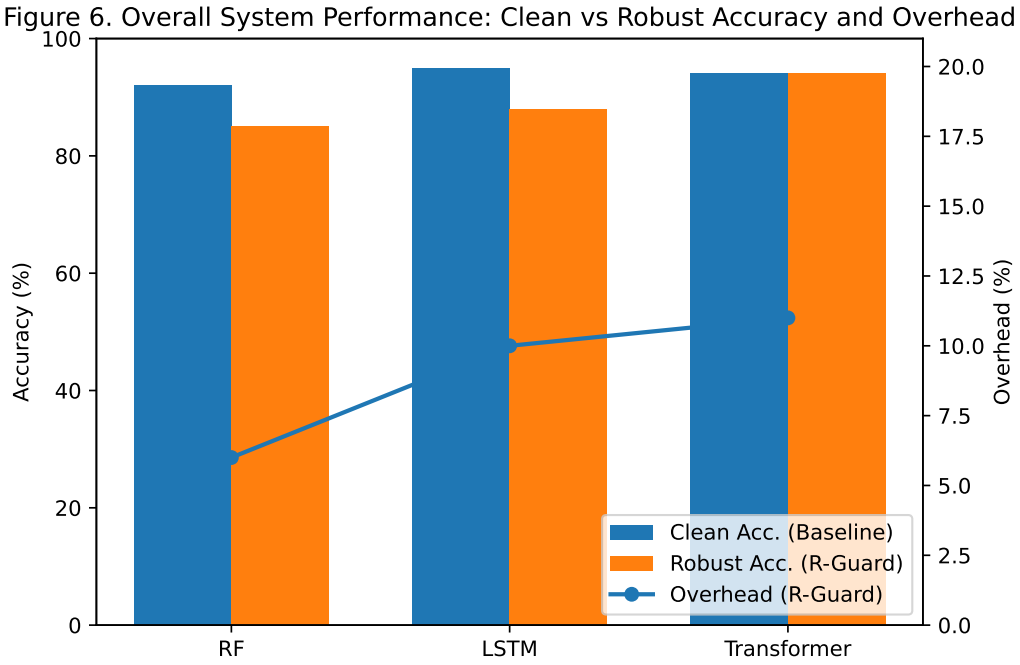


Figure A.22: Overall system performance summary comparing clean accuracy, robust accuracy, and computational/processing overhead for baseline and R-Guard configurations. The figure provides a supplementary deployment-oriented view of the accuracy–robustness–overhead relationship.

ROC Behaviour Under Adversarial Attack

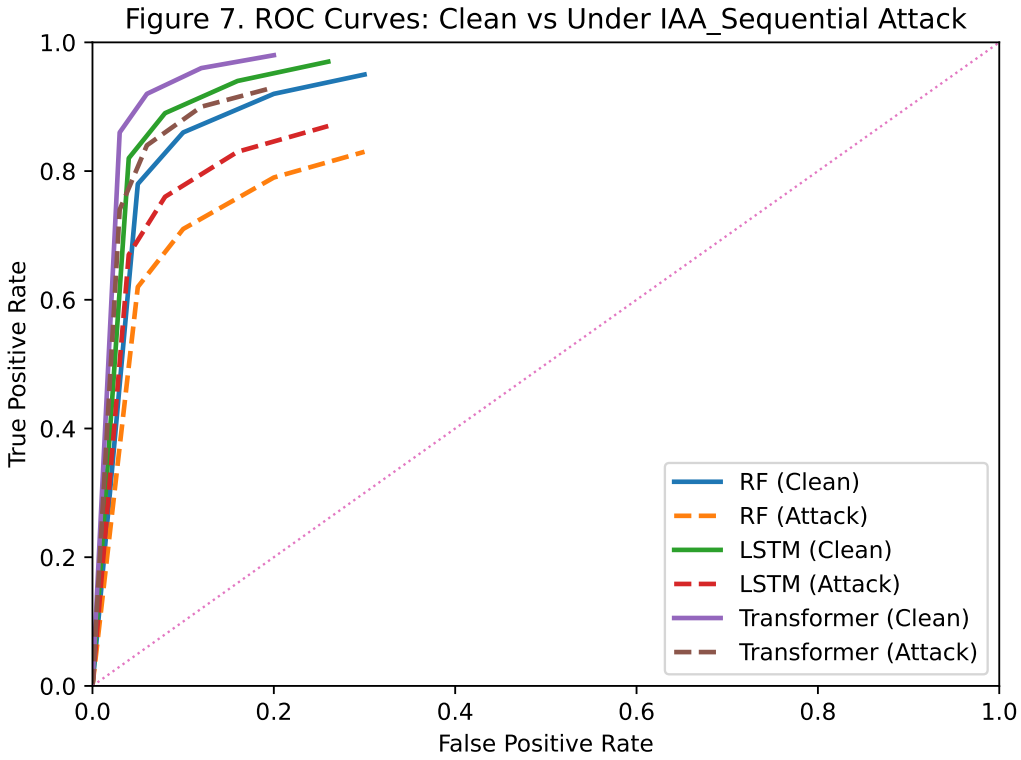


Figure A.23: ROC curves under clean and adversarial conditions for representative baseline and R-Guard-augmented models. The figure provides a supplementary discrimination-focused view of adversarial degradation and robustness recovery.

Multi-Metric Summary Visual (Radar)

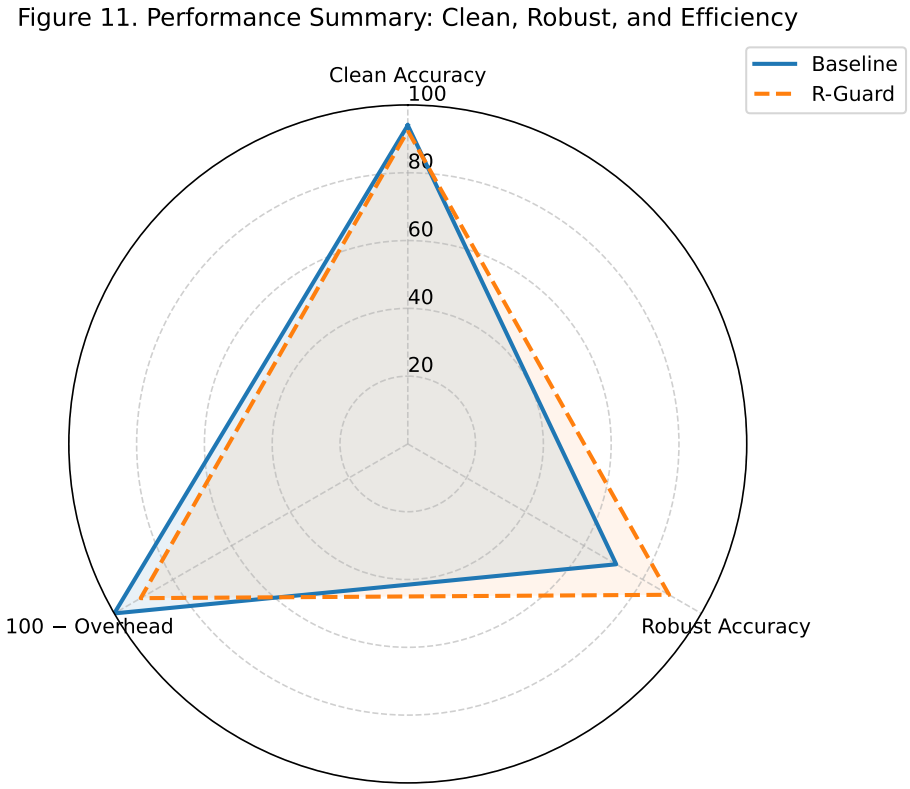


Figure A.24: Radar-chart summary of selected performance dimensions (e.g., clean accuracy, robust accuracy, and efficiency/overhead) for baseline and R-Guard configurations. This figure is included as a compact synthesis visual and is supplementary to the primary quantitative plots in Chapter 6.

Diagnostic Error Analysis

This subsection provides diagnostic visualisations used to interpret how R-Guard changes error behaviour under adversarial pressure.

Confusion Matrix Comparison

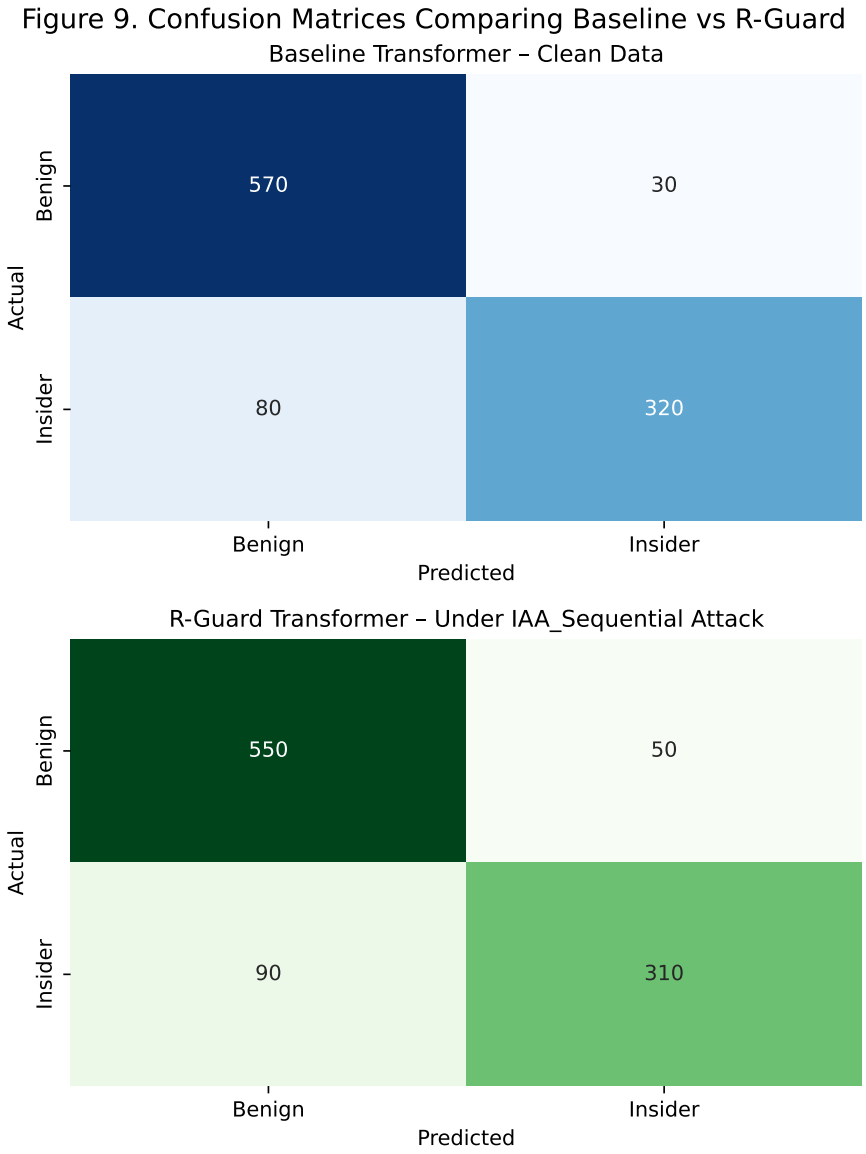


Figure A.25: Representative confusion-matrix comparison for a baseline detector and its R-Guard-augmented counterpart under behavioural adversarial evaluation. The figure is included as a diagnostic illustration of error-pattern changes, particularly false-negative reduction under stealthy attack conditions.

Diagnostic interpretation note. Confusion matrices are threshold-dependent and should be interpreted together with threshold-independent metrics (e.g., ROC and PR analyses) and robustness curves reported in Chapter 6 and this appendix.

Ethical Use and Adaptation Note

The artefacts and implementation materials presented in this section are provided for research, education, and defensive cybersecurity experimentation. Although the adversarial components are used here to evaluate detector robustness, any adaptation of the code to new datasets or operational settings should be undertaken with

explicit ethical approval, legal authority, and appropriate organisational governance.

In particular, researchers and practitioners should ensure that:

- monitored data are collected and processed lawfully;
- privacy and proportionality requirements are respected;
- behavioural monitoring outputs are used as decision support rather than autonomous punitive decisions;
- robustness testing is conducted in controlled environments and not deployed in ways that increase organisational harm or surveillance misuse.

The emphasis of this work is on improving the safety, reliability, and accountability of insider-threat detection systems through behaviourally grounded robustness evaluation.

Reproducibility note. Where practical, version information (Python libraries, package versions, random seeds, and data preprocessing assumptions) should be recorded alongside the code artefact to support consistent re-execution.

Appendix B

Methodological Decision Framework

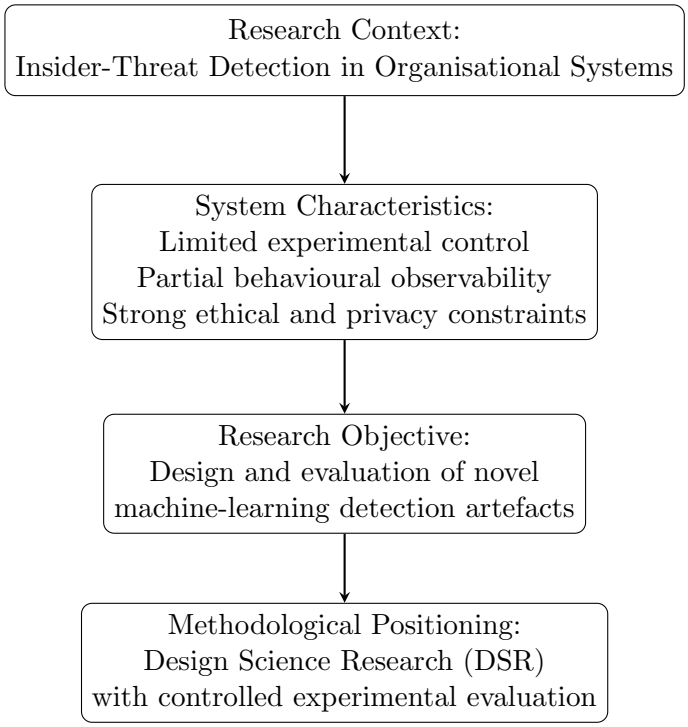


Figure B.1: Methodological decision framework adapted for insider-threat detection research.

This appendix provides a supporting methodological decision framework that underpins the research design adopted throughout this thesis. Its purpose is to make explicit the reasoning process through which the Design Science Research (DSR) paradigm and controlled experimental evaluation were selected.

Insider-threat research is uniquely constrained by the complexity of socio-technical systems, the partial observability of human behaviour, and ethical limitations associated with experimenting on real organisational environments. These structural constraints preclude classical laboratory experimentation or in situ intervention-based studies.

Consequently, the primary research objective focuses on the construction and rigorous evaluation of novel detection artefacts. This objective aligns with the DSR

paradigm, where knowledge contribution is achieved through the design and implementation of artefacts intended to address identified problems. The architectures introduced in Chapters 3 to 6 are treated as artefacts whose utility and robustness are systematically assessed using established datasets.

Appendix C

Algorithms Adapted

Algorithm 3: (C1) ANN Baseline Implementation for Insider-Threat Detection (CERT multi-log)

Input: CERT logs $\mathcal{L} = \{\texttt{users}, \texttt{logon}, \texttt{device}, \texttt{http}, \texttt{email}, \texttt{file}, \texttt{psychometric}\}$; windowing rule; label definition y; chronology split policy; ANN hyperparameters Θ

Output: Trained ANN model f_Θ; evaluation report $\mathcal{R}$; saved artefacts

Stage 1: Data preparation (chronology-preserving)

Load and validate logs; align timestamps; define windows $w(u, t)$ in chronological order; construct labels y

Stage 2: Feature construction (multi-log)

Aggregate per-window behavioural features across domains (counts/ratios/temporal indicators) and join user/context metadata to form (X, y)

Stage 3: Leakage-safe preprocessing

Split Train/Val/Test chronologically (blocked folds if applicable); fit preprocessing on Train only; transform Val/Test; handle imbalance (weights/thresholding)

Stage 4: ANN training

Define ANN using Θ; train with early stopping on validation criterion (e.g., Val F1 or Val PR-AUC); store best checkpoint

Stage 5: Evaluation and reporting

Evaluate on Test (F1, PR-AUC, ROC-AUC, confusion matrix); tune threshold on Val if required; save metrics, plots, and configs as $\mathcal{R}$

Ethical use notice: Use under governance; treat outputs as decision support, not automated punitive judgement

Algorithm 4: (C2) SOM–ANN (SOMANN) Hybrid Implementation for Representation Stabilisation

Input: Feature matrix X (from multi-log windows); labels y; chronology split policy; SOM params Φ (grid size, learning rate, epochs); ANN params Θ

Output: Trained SOMANN model (SOM_Φ, f_Θ); evaluation report $\mathcal{R}$

Stage 1: Chronology-preserving split + preprocessing

Split into Train/Val/Test by time; fit scalers/encoders on Train only; transform Val/Test

Stage 2: Train SOM on Train features

Initialise SOM weights and topology (e.g., 2D grid)

for $e \leftarrow 1$ **to** E **do**

  sample Train rows (shuffled within Train or ordered within Train as defined)

  find Best Matching Unit (BMU) and update neighbourhood weights

Compute SOM embedding per instance (e.g., BMU coordinates, distance-to-BMU, activation vector)

Stage 3: Build stabilised representation

Construct hybrid representation $Z = [X \parallel \text{SOM-embed}(X)]$ for Train/Val/Test

(Optionally reduce dimensionality of SOM-embed if using activation vectors)

Stage 4: Train ANN classifier on stabilised representation

Train ANN f_Θ on Z_{train} with early stopping on Z_{val}

Tune threshold on Val if required; freeze best checkpoint

Stage 5: Evaluate

Evaluate on Z_{test}: F1, PR-AUC, ROC-AUC, confusion matrix; save plots and configs

Ethical use notice: SOM embeddings can reveal behavioural clustering; ensure outputs are used for governed investigation support rather than automated sanction

Algorithm 5: (C3) MANN Implementation (Domain-Modular ANN with Structured Fusion)

Input: Domain features $\{X^{(d)}\}_{d=1}^{D}$ (e.g., logon/file/device/http/email/user); labels y; chronology split policy; module hyperparameters $\{\Theta^{(d)}\}$; fusion params Ψ

Output: Trained modular model $\mathcal{M} = \{f_{\Theta^{(d)}}^{(d)}\}_{d=1}^{D}$ and fusion head g_Ψ; report $\mathcal{R}$

Stage 1: Construct domain-aligned inputs

Create domain-specific feature matrices $X^{(1)}, ..., X^{(D)}$ with consistent row alignment across domains

Split Train/Val/Test chronologically; fit preprocessing per domain on Train only

Stage 2: Train domain modules

foreach *domain* $d \in \{1, \ldots, D\}$ **do**

- Initialise module $f_{\Theta^{(d)}}^{(d)}$ (dense network or domain-appropriate encoder)
- Train on $X_{train}^{(d)}$ with early stopping on Val
- Extract latent embedding $h^{(d)}$ from a penultimate layer for Train/Val/Test

Stage 3: Structured fusion

Concatenate embeddings $H = [h^{(1)} \| h^{(2)} \| \cdots \| h^{(D)}]$

Train fusion head g_Ψ on H_{train} to output risk score $\hat{y}$ (with class weighting/thresholding)

Optionally fine-tune end-to-end with a small learning rate while freezing low-level layers

Stage 4: Evaluation + diagnostics

Evaluate on Test: PR-AUC, ROC-AUC, F1, confusion matrix

Perform domain ablation diagnostics (mask one domain at a time) to quantify domain contribution

Save module/fusion artefacts and evaluation report $\mathcal{R}$

Ethical use notice: Domain-level attributions can be sensitive; use for analyst-centred decision support under governance and proportionality constraints

Algorithm 6: (C4) Hierarchical Multi-Layered Implementation for Behavioural Progression Modelling

Input: Event logs $\mathcal{L}$; hierarchy design $\mathcal{H}$ (event→session→task→user); label definition y; chronology split policy; layer models $\{\Theta_\ell\}$; aggregation operators $\{\mathcal{A}_\ell\}$

Output: Trained hierarchical model $\mathbb{H}$; evaluation report $\mathcal{R}$

Stage 1: Define hierarchy and temporal units

Define levels $\ell = 0..L$ (e.g., ℓ=0 events, ℓ=1 sessions, ℓ=2 daily task windows, ℓ=3 user profile)

Define mapping rules from lower to higher levels (session boundaries, window length, aggregation rules)

Stage 2: Chronology-safe construction

Sort events by time; build level-0 sequences per user

Aggregate using $\mathcal{A}_\ell$ to build level-ℓ representations (counts, temporal statistics, embeddings)

Generate labels at the chosen prediction level (e.g., window/user)

Split Train/Val/Test chronologically; fit preprocessing on Train only

Stage 3: Train hierarchical encoders bottom-up

Train level-0 encoder (if used) to produce event embeddings e_t

Aggregate to sessions with $\mathcal{A}_1$; train level-1 encoder to produce session embeddings

Repeat until top-level representation z is produced

Attach classifier head to z and train with early stopping

Stage 4: Trajectory-aware evaluation

Evaluate standard metrics (F1, PR-AUC, ROC-AUC)

Compute trajectory diagnostics (e.g., detection delay, performance by behavioural stage) where applicable

Perform ablation to quantify the contribution of each hierarchy level

Ethical use notice: Hierarchical models can infer behavioural states; ensure findings are interpreted under clear validity boundaries and used with governance safeguards

Algorithm 7: (C5) R-Guard Implementation for Behaviourally Constrained Adversarial Robustness

Input: Base detector (ANN/SOMANN/MANN/Hierarchical); training data (X, y); behavioural threat model $\mathcal{T}$; perturbation operators $\{\mathcal{P}_k\}$; robustness params Ω; chronology split policy

Output: Robust model $f^\star$; robustness report $\mathcal{R}_{rob}$; diagnostics (curves, ablations)

Stage 1: Baseline training under temporal validity

Split Train/Val/Test chronologically; train base detector with leakage-safe preprocessing

Record nominal performance on Val/Test

Stage 2: Define behaviourally constrained adversary

Specify threat goals (evasion, delayed detection) and constraints (privilege-consistent, workflow-plausible, bounded budget)

Define operator taxonomy $\{\mathcal{P}_k\}$ (e.g., time-shift within plausible bounds, activity masking, cross-domain consistency-preserving edits)

Define perturbation budget schedule B and attack strategy $\mathcal{T}$

Stage 3: Robustness training (R-Guard mechanisms)

Generate perturbed training samples using $\mathcal{T}$ under budgets B

Apply robustness objectives (invariance constraints, temporal coherence, cross-domain consistency regularisation)

Train robust model $f^\star$ with mixed clean + perturbed batches; tune robustness/utility trade-off on Val

Stage 4: Robustness evaluation

Evaluate $f^\star$ on clean Test and attacked Test for multiple budgets B

Compute robust metrics: robust F1/PR-AUC, attack success rate (ASR), degradation curves

Compute time-to-detection under gradual drift (if drift protocol is used) and report robustness/utility trade-offs

Stage 5: Reporting and safeguards

Save robustness plots (ASR vs budget, ROC/PR under attack, trade-off curves) and configs

Document threat model assumptions and validity boundaries

Ethical use notice: Do not operationalise attack code against real systems; restrict adversarial experiments to governed research contexts

Appendix D

Implementation Artefacts

This appendix presents the implementation artefacts underpinning the reproducibility of the main modelling contributions of this thesis: the SOMANN framework, the Modular Artificial Neural Network (MANN), the Multi-Layered Machine Learning (MLML) framework, and the *Guarding the Gates* adversarial robustness evaluation pipeline. The artefacts are provided as research-oriented reference implementations that document the complete workflow from CERT-style behavioural log ingestion and privacy-minimised feature engineering to model training, inference, and chronology-preserving blocked temporal-fold evaluation. Their inclusion is intended to strengthen methodological transparency, support replication, and clarify implementation-level design choices across the different architectural stages of the thesis. **Ethical Use Notice:** these artefacts are supplied for research, validation, and educational purposes and should not be deployed in operational settings without appropriate legal basis, governance oversight, proportionality assessment, data minimisation, secure access controls, and human-in-the-loop decision processes. Given the sensitivity of behavioural monitoring, any real-world adaptation should additionally be subject to fairness analysis, false-positive impact assessment, and formal ethical review before use in practice.

D.1 Artefact—SOMANN Implementation

The artefact in this section provides a reproducible implementation of the SOM–ANN (SOMANN) framework used in this thesis for insider-threat detection on CERT-style behavioural telemetry. The implementation operationalises the full pipeline from data ingestion and preprocessing to feature construction, topology-preserving SOM representation learning, supervised ANN classification, and chronology-preserving evaluation using blocked temporal folds. The artefact is presented as a research implementation that can be adapted to other datasets; however, any such adaptation should be undertaken with explicit consideration of data governance, privacy, fairness, and the organisational consequences of false-positive decisions.

```
#
    ============================================================

# Appendix D Artefact: SOMANN Implementation
# Self-Organising Map (SOM) + Artificial Neural Network (ANN)
# Pipeline: ingest -> feature engineering -> temporal blocked
    CV

import os
import gc
import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd

from dataclasses import dataclass
from typing import Dict, Tuple, Optional, List

from sklearn.model_selection import TimeSeriesSplit
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import (
    precision_recall_fscore_support,
    average_precision_score,
    roc_auc_score,
    confusion_matrix
)

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# SOM (MiniSom)
try:
    from minisom import MiniSom
except ImportError:
    !pip -q install minisom
    from minisom import MiniSom

SEED = 42
np.random.seed(SEED)
tf.random.set_seed(SEED)

```

```
41 #
   ============================================================

42 # 1) Configuration
43 #
   ============================================================

44
45 @dataclass
46 class CFG:
47     base_path: str = "/kaggle/input/datasets/mrajaxnp/cert-
          insider-threat-detection-research/"
48
49     users_path: str = f"{base_path}/users.csv"
50     logon_path: str = f"{base_path}/logon.csv"
51     device_path: str = f"{base_path}/device.csv"
52     psych_path: str = f"{base_path}/psychometric.csv"
53     http_path: str = f"{base_path}/http.csv"
54     email_path: str = f"{base_path}/email.csv"
55     file_path: str = f"{base_path}/file.csv"
56
57     # Core columns used across CERT logs
58     time_col: str = "date"
59     user_col: str = "user"
60
61     # Label definition (choose one for a clear and defensible
           binary task)
62     # Common CERT proxy label: exfiltration to removable
          media.
63     label_mode: str = "to_removable_media"   # options: "
          to_removable_media" or "from_removable_media"
64
65     # Temporal blocked evaluation
66     n_splits: int = 5
67
68     # Feature engineering window (user-day)
69     # This minimises privacy risk vs event-level modelling
          and supports temporal splits.
70     window: str = "D"   # daily aggregation
71
72     # File.csv is large - use chunking
73     file_chunksize: int = 2_000_000
74
```

```
    # SOM hyperparameters
    som_x: int = 15
    som_y: int = 15
    som_sigma: float = 1.0
    som_lr: float = 0.5
    som_iters: int = 5000

    # ANN hyperparameters
    hidden_units: Tuple[int, ...] = (128, 64)
    dropout: float = 0.2
    batch_size: int = 512
    epochs: int = 40
    patience: int = 6

    # Class imbalance handling
    use_class_weights: bool = True

cfg = CFG()

# ============================================================

# 2) Utilities: Safe parsing + time features
# ============================================================


def parse_datetime_safe(s: pd.Series) -> pd.Series:
    return pd.to_datetime(s, errors="coerce", infer_datetime_format=True)

def add_time_features(df: pd.DataFrame, time_col: str) -> pd.DataFrame:
    # Ethical note: time-derived features are usually less sensitive than content.
    # Prefer these over deep inspection of message/file content.
    dt = df[time_col]
    df["hour"] = dt.dt.hour
    df["dayofweek"] = dt.dt.dayofweek  # Mon=0
    df["is_weekend"] = (df["dayofweek"] >= 5).astype(int)
    df["is_after_hours"] = ((df["hour"] < 7) | (df["hour"] >= 19)).astype(int)
```

```
109     df["date_only"] = dt.dt.floor("D")
110     return df
111
112 def ensure_exists(path: str):
113     if not os.path.exists(path):
114         raise FileNotFoundError(f"Missing file: {path}")
115
116 for p in [cfg.users_path, cfg.logon_path, cfg.device_path,
        cfg.psych_path, cfg.http_path, cfg.email_path, cfg.
        file_path]:
117     ensure_exists(p)
118
119 #
        ============================================================

120 # 3) Load small/static tables (users, psychometric)
121 #
        ============================================================

122
123 def load_users(users_path: str) -> pd.DataFrame:
124     users = pd.read_csv(users_path)
125
126     # CERT variants sometimes use user_id instead of user
127     if "user" not in users.columns:
128         if "user_id" in users.columns:
129             users = users.rename(columns={"user_id": "user"})
130         else:
131             raise ValueError(
132                 "users.csv must contain 'user' or 'user_id'.
                    "
133                 f"Found columns: {list(users.columns)}"
134             )
135
136     # Keep minimal columns for safe merging (privacy
        minimisation)
137     return users
138
139 def load_psych(psych_path: str) -> pd.DataFrame:
140     psych = pd.read_csv(psych_path)
141
142     if "user" not in psych.columns:
143         if "user_id" in psych.columns:
```

```
144             psych = psych.rename(columns={"user_id": "user"})
145         else:
146             raise ValueError(
147                 "psychometric.csv must contain 'user' or '
                    user_id'. "
148                 f"Found columns: {list(psych.columns)}"
149             )
150     return psych
151
152 users_df = load_users(cfg.users_path)
153 psych_df = load_psych(cfg.psych_path)
154
155 print("users_df:", users_df.shape, "| psych_df:", psych_df.
    shape)
156
157 # Toggle psychometric inclusion (recommended: False unless
    explicitly justified)
158 INCLUDE_PSYCH = False
159
160 print("users columns:", list(users_df.columns))
161 print("logon columns:", list(pd.read_csv(cfg.logon_path,
    nrows=5).columns))
162 print("file columns:",  list(pd.read_csv(cfg.file_path,
    nrows=5).columns))
163
164 #
    ================================================================
165 # 4) Feature engineering per log (user-day aggregation)
166 #
    ================================================================
167
168 def agg_logon(logon_path: str) -> pd.DataFrame:
169     df = pd.read_csv(logon_path)
170     df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
171     df = df.dropna(subset=[cfg.time_col, cfg.user_col])
172     df = add_time_features(df, cfg.time_col)
173
174     # Typical columns: user, pc, activity
175     # Feature set: counts + after-hours/weekend ratios
176     out = df.groupby([cfg.user_col, "date_only"]).agg(
177         logon_events=("activity", "size"),
```

```
        logon_after_hours=("is_after_hours", "sum"),
        logon_weekend=("is_weekend", "sum"),
        logon_unique_pc=("pc", pd.Series.nunique)
    ).reset_index()

    out["logon_after_hours_ratio"] = out["logon_after_hours"] / out["logon_events"].clip(lower=1)
    out["logon_weekend_ratio"] = out["logon_weekend"] / out["logon_events"].clip(lower=1)
    return out

def agg_device(device_path: str) -> pd.DataFrame:
    df = pd.read_csv(device_path)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    # Typical columns: user, pc, activity
    out = df.groupby([cfg.user_col, "date_only"]).agg(
        device_events=("activity", "size"),
        device_after_hours=("is_after_hours", "sum"),
        device_weekend=("is_weekend", "sum"),
        device_unique_pc=("pc", pd.Series.nunique)
    ).reset_index()

    out["device_after_hours_ratio"] = out["device_after_hours"] / out["device_events"].clip(lower=1)
    out["device_weekend_ratio"] = out["device_weekend"] / out["device_events"].clip(lower=1)
    return out

def agg_http(http_path: str) -> pd.DataFrame:
    df = pd.read_csv(http_path)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    # Keep minimal - do NOT parse content URLs beyond count features (privacy minimisation)
    out = df.groupby([cfg.user_col, "date_only"]).agg(
        http_events=("url", "size") if "url" in df.columns else (cfg.user_col, "size"),
        http_after_hours=("is_after_hours", "sum"),
```

```
        http_weekend=("is_weekend", "sum")
    ).reset_index()

    out["http_after_hours_ratio"] = out["http_after_hours"] /
        out["http_events"].clip(lower=1)
    out["http_weekend_ratio"] = out["http_weekend"] / out["
       http_events"].clip(lower=1)
    return out

def agg_email(email_path: str) -> pd.DataFrame:
    df = pd.read_csv(email_path)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    # Again minimise: counts and timing patterns, not message
        bodies.
    out = df.groupby([cfg.user_col, "date_only"]).agg(
        email_events=(cfg.user_col, "size"),
        email_after_hours=("is_after_hours", "sum"),
        email_weekend=("is_weekend", "sum")
    ).reset_index()

    out["email_after_hours_ratio"] = out["email_after_hours"]
        / out["email_events"].clip(lower=1)
    out["email_weekend_ratio"] = out["email_weekend"] / out["
       email_events"].clip(lower=1)
    return out

#
   ============================================================

# 5) file.csv: chunked aggregation + label creation
#
   ============================================================


def agg_file_chunked(file_path: str, chunksize: int) -> Tuple
   [pd.DataFrame, pd.DataFrame]:
    """
    Returns:
      file_features: user-day aggregated features from file.
         csv
```

```
    labels: user-day labels based on cfg.label_mode within the day
    """
    feats_accum = []
    labels_accum = []

    use_cols = [cfg.time_col, cfg.user_col, "pc", "filename", "activity", "to_removable_media", "from_removable_media"]
    # Some variants store booleans as strings; handle robustly.
    for chunk in pd.read_csv(file_path, usecols=lambda c: c in use_cols, chunksize=chunksize):
        chunk[cfg.time_col] = parse_datetime_safe(chunk[cfg.time_col])
        chunk = chunk.dropna(subset=[cfg.time_col, cfg.user_col])
        chunk = add_time_features(chunk, cfg.time_col)

        # Normalise boolean columns
        for b in ["to_removable_media", "from_removable_media"]:
            if b in chunk.columns:
                if chunk[b].dtype == object:
                    chunk[b] = chunk[b].astype(str).str.lower().map({"true": 1, "false": 0})
                chunk[b] = chunk[b].fillna(0).astype(int)
            else:
                chunk[b] = 0

        # Feature aggregation (privacy-minimised: no content parsing)
        grp = chunk.groupby([cfg.user_col, "date_only"]).agg(
            file_events=("activity", "size"),
            file_after_hours=("is_after_hours", "sum"),
            file_weekend=("is_weekend", "sum"),
            file_unique_pc=("pc", pd.Series.nunique),
            file_unique_filename=("filename", pd.Series.nunique),
            file_to_rm=("to_removable_media", "sum"),
            file_from_rm=("from_removable_media", "sum")
        ).reset_index()

```

```
279         grp["file_after_hours_ratio"] = grp["file_after_hours
               "] / grp["file_events"].clip(lower=1)
280         grp["file_weekend_ratio"] = grp["file_weekend"] / grp
               ["file_events"].clip(lower=1)
281
282         feats_accum.append(grp)
283
284         # Label: day is positive if any relevant removable-
               media event occurs
285         label_col = "file_to_rm" if cfg.label_mode == "
               to_removable_media" else "file_from_rm"
286         lab = grp[[cfg.user_col, "date_only", label_col]].
               copy()
287         lab["label"] = (lab[label_col] > 0).astype(int)
288         lab = lab.drop(columns=[label_col])
289         labels_accum.append(lab)
290
291         del chunk, grp, lab
292         gc.collect()
293
294     file_features = pd.concat(feats_accum, ignore_index=True)
295     labels = pd.concat(labels_accum, ignore_index=True)
296
297     # If the same (user, day) appears across chunks, re-
           aggregate safely:
298     file_features = file_features.groupby([cfg.user_col, "
           date_only"], as_index=False).sum(numeric_only=True)
299     labels = labels.groupby([cfg.user_col, "date_only"],
           as_index=False).max(numeric_only=True)
300
301     return file_features, labels
302
303 file_feats, y_day = agg_file_chunked(cfg.file_path, cfg.
       file_chunksize)
304 print("file_feats:", file_feats.shape, "| labels:", y_day.
       shape, "| pos rate:", y_day["label"].mean())
305
306 #
       ============================================================
307 # 6) Build unified user-day dataset (merge log features)
308 #
       ============================================================
```

```
309
310 logon_feats = agg_logon(cfg.logon_path)
311 device_feats = agg_device(cfg.device_path)
312 http_feats  = agg_http(cfg.http_path)
313 email_feats = agg_email(cfg.email_path)
314
315 # Merge all on (user, date)
316 df_day = file_feats.merge(logon_feats, on=[cfg.user_col, "
       date_only"], how="left") \
317                    .merge(device_feats, on=[cfg.user_col, "
                          date_only"], how="left") \
318                    .merge(http_feats,  on=[cfg.user_col, "
                          date_only"], how="left") \
319                    .merge(email_feats, on=[cfg.user_col, "
                          date_only"], how="left") \
320                    .merge(y_day,       on=[cfg.user_col, "
                          date_only"], how="inner")
321
322 # Fill missing (no activity in that stream) with zeros
323 df_day = df_day.fillna(0)
324
325 # Optional: merge static tables (minimise & justify use)
326 # Users table may include role/department; include only if
       thesis justifies it.
327 # Psychometric is more sensitive: default excluded.
328 df_day = df_day.merge(users_df[[cfg.user_col]].
       drop_duplicates(), on=cfg.user_col, how="left")
329
330 if INCLUDE_PSYCH:
331     # Include only numeric psychometric fields; be careful
          with ethics.
332     psych_num = psych_df.select_dtypes(include=[np.number]).
          copy()
333     psych_min = pd.concat([psych_df[[cfg.user_col]],
          psych_num], axis=1)
334     df_day = df_day.merge(psych_min, on=cfg.user_col, how="
          left").fillna(0)
335
336 # Sort chronologically for temporal evaluation
337 df_day = df_day.sort_values("date_only").reset_index(drop=
       True)
338 print("Unified user-day dataset:", df_day.shape)
```

```
print("Positive rate:", df_day["label"].mean())

# ============================================================

# 7) Preprocessing: numeric scaling (fit on TRAIN only per fold)
# ============================================================


# Define feature columns (exclude identifiers and label)
DROP_COLS = [cfg.user_col, "date_only", "label"]
feature_cols = [c for c in df_day.columns if c not in DROP_COLS]

# All engineered features are numeric by construction (privacy-minimised).
# If you add categorical features, do so cautiously and justify ethically.
X_all = df_day[feature_cols].astype(np.float32).values
y_all = df_day["label"].astype(int).values
t_all = df_day["date_only"].values  # for sanity checks

# ============================================================

# 8) SOMANN core: SOM fit -> SOM features -> ANN fit
# ============================================================


def fit_som(X_train: np.ndarray) -> MiniSom:
    som = MiniSom(
        x=cfg.som_x, y=cfg.som_y,
        input_len=X_train.shape[1],
        sigma=cfg.som_sigma,
        learning_rate=cfg.som_lr,
        neighborhood_function="gaussian",
        random_seed=SEED
    )
    som.random_weights_init(X_train)
    som.train_random(X_train, cfg.som_iters)
```

```
370     return som
371
372 def som_features(som: MiniSom, X: np.ndarray) -> np.ndarray:
373     # BMU coordinates + quantisation error (compact +
            interpretable)
374     bmu = np.array([som.winner(x) for x in X], dtype=np.int32
            )
375     bmu_x = bmu[:, 0:1].astype(np.float32)
376     bmu_y = bmu[:, 1:2].astype(np.float32)
377     qe = np.array([som.quantization_error([x]) for x in X],
            dtype=np.float32).reshape(-1, 1)
378     return np.hstack([bmu_x, bmu_y, qe])
379
380 def build_somann_input(X_scaled: np.ndarray, som: MiniSom) ->
        np.ndarray:
381     return np.hstack([X_scaled, som_features(som, X_scaled)])
            .astype(np.float32)
382
383 def build_ann(input_dim: int) -> keras.Model:
384     inp = keras.Input(shape=(input_dim,), name="somann_input"
            )
385     x = inp
386     for i, hu in enumerate(cfg.hidden_units):
387         x = layers.Dense(hu, activation="relu", name=f"dense_
                {i+1}")(x)
388         x = layers.Dropout(cfg.dropout, name=f"dropout_{i+1}"
                )(x)
389     out = layers.Dense(1, activation="sigmoid", name="out")(x
            )
390     model = keras.Model(inp, out, name="SOMANN")
391     model.compile(
392         optimizer=keras.optimizers.Adam(1e-3),
393         loss="binary_crossentropy"
394     )
395     return model
396
397 def class_weights(y: np.ndarray) -> Dict[int, float]:
398     n_pos = max(1, int(y.sum()))
399     n_neg = max(1, int((y == 0).sum()))
400     w_pos = (n_pos + n_neg) / (2.0 * n_pos)
401     w_neg = (n_pos + n_neg) / (2.0 * n_neg)
402     return {0: w_neg, 1: w_pos}
403
```

```
404 def eval_metrics(y_true: np.ndarray, y_prob: np.ndarray, thr:
        float = 0.5) -> Dict[str, float]:
405     y_pred = (y_prob >= thr).astype(int)
406
407     p, r, f1, _ = precision_recall_fscore_support(
408         y_true, y_pred, average="binary", zero_division=0
409     )
410     pr_auc = average_precision_score(y_true, y_prob)
411
412     roc_auc = np.nan
413     if len(np.unique(y_true)) == 2:
414         roc_auc = roc_auc_score(y_true, y_prob)
415
416     tn, fp, fn, tp = confusion_matrix(y_true, y_pred).ravel()
417
418     return {
419         "precision": float(p),
420         "recall": float(r),
421         "f1": float(f1),
422         "pr_auc": float(pr_auc),
423         "roc_auc": float(roc_auc) if not np.isnan(roc_auc)
              else np.nan,
424         "tp": int(tp), "fp": int(fp), "tn": int(tn), "fn":
              int(fn)
425     }
426
427 #
    ================================================================

428 # 9) Blocked temporal folds evaluation (TimeSeriesSplit)
429 #
    ================================================================


430
431 tscv = TimeSeriesSplit(n_splits=cfg.n_splits)
432 results = []
433
434 for fold, (train_idx, test_idx) in enumerate(tscv.split(X_all
    ), start=1):
435
436     # Chronology is preserved by TimeSeriesSplit (blocked
          temporal evaluation).
437     X_train_raw, X_test_raw = X_all[train_idx], X_all[
```

```
    test_idx]
y_train, y_test = y_all[train_idx], y_all[test_idx]

# Fit scaler on train only (prevents leakage)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train_raw)
X_test  = scaler.transform(X_test_raw)

# Fit SOM on train only (unsupervised)
som = fit_som(X_train)

# Build SOMANN features
X_train_somann = build_somann_input(X_train, som)
X_test_somann  = build_somann_input(X_test,  som)

# ANN training with temporal validation split (tail of
    training set)
n_train = len(X_train_somann)
val_frac = 0.2
split = int((1.0 - val_frac) * n_train)

X_tr, X_val = X_train_somann[:split], X_train_somann[
    split:]
y_tr, y_val = y_train[:split], y_train[split:]

model = build_ann(X_train_somann.shape[1])
callbacks = [
    keras.callbacks.EarlyStopping(
        monitor="val_loss", patience=cfg.patience,
            restore_best_weights=True
    )
]

cw = class_weights(y_tr) if cfg.use_class_weights else
    None

model.fit(
    X_tr, y_tr,
    validation_data=(X_val, y_val),
    epochs=cfg.epochs,
    batch_size=cfg.batch_size,
    verbose=0,
    class_weight=cw,
```

```
        callbacks=callbacks
    )

    y_prob = model.predict(X_test_somann, batch_size=cfg.
       batch_size, verbose=0).ravel()
    m = eval_metrics(y_test, y_prob, thr=0.5)

    m["fold"] = fold
    m["n_test"] = int(len(test_idx))
    m["pos_rate_test"] = float(y_test.mean())

    print(
        f"Fold {fold}/{cfg.n_splits} | "
        f"F1={m['f1']:.3f} PR-AUC={m['pr_auc']:.3f} "
        f"P={m['precision']:.3f} R={m['recall']:.3f} | "
        f"TP={m['tp']} FP={m['fp']} FN={m['fn']} TN={m['tn']}
           "
    )

    results.append(m)

    # Cleanup
    tf.keras.backend.clear_session()
    del model, som, scaler, X_train_somann, X_test_somann,
       X_train, X_test
    gc.collect()

res_df = pd.DataFrame(results)
print("\nPer-fold results:\n", res_df)

# Summary (Mean    Std)
summary = res_df[["precision", "recall", "f1", "pr_auc", "
   roc_auc"]].agg(["mean", "std"]).T
summary.columns = ["mean", "std"]
print("\nSummary (Mean    Std):\n", summary)

# Export artefacts (safe: aggregated metrics only)
res_path = "/kaggle/working/somann_cert_temporal_folds.csv"
sum_path = "/kaggle/working/somann_cert_summary.csv"
res_df.to_csv(res_path, index=False)
summary.to_csv(sum_path)

print("\nSaved artefacts:")
```

```
print(" -", res_path)
print(" -", sum_path)

# ============================================================

# 10) Short ethics-focused model card (Appendix-friendly)
# ============================================================


model_card = f"""
SOMANN Model Card (CERT research artefact)
- Intended use: Research-oriented anomaly/risk scoring for insider-like behaviours.
- Data: Aggregated user-day behavioural signals derived from CERT logs (minimised features).
- Label: Proxy event derived from file.csv ({cfg.label_mode}) aggregated at user-day level.
- Evaluation: Blocked temporal folds to preserve chronology and reduce leakage.
- Primary metrics: PR-AUC, Precision, Recall, F1 (PR-AUC prioritised under class imbalance).
- Key risks: Privacy intrusion, proxy-label bias, false positives, role/department confounding.
- Mitigations: Data minimisation, temporal splits, human-in-the-loop review, governance/consent checks.
- Prohibited use: Automated punitive action or covert surveillance without due process and oversight.
"""
print(model_card)
```

Listing D.1: SOMANN (SOM + ANN) Implementation for the CERT dataset using blocked temporal folds.

D.2 Artefact—MANN Implementation

The artefact in this section provides a reproducible implementation of the Modular Artificial Neural Network (MANN) framework developed in Chapter 4. The implementation reflects the core architectural principle of behavioural domain decomposition by modelling semantically distinct activity streams through domain-specific modules prior to structured fusion, thereby preserving domain semantics before risk estimation. Consistent with the experimental design of the thesis, the artefact adopts chronology-preserving blocked temporal folds and leakage-aware preprocessing. It is presented for research validation and architectural reproducibility, with explicit recognition that deployment-oriented adaptation requires careful attention to proportional monitoring, interpretability, false-positive impact, and ethical oversight.

```
#
    ============================================================

# Appendix D2 Artefact: MANN Implementation (CERT Kaggle)
# Modular Artificial Neural Network for insider-style
    detection
# Pipeline: ingest -> per-domain features -> modular ANN ->
    eval
#
    ============================================================

# ETHICAL USE NOTICE:
# This artefact demonstrates modular modelling on behavioural
     logs.
# Before adapting to real organisational data:
#   - confirm lawful basis/consent, transparency, governance
    oversight
#   - minimise features and avoid unnecessary sensitive
    content processing
#   - assess bias (role/department proxies, skewed monitoring
     coverage)
#   - keep human-in-the-loop; prohibit automated punitive
    action
#   - secure access, audit, retention limits, and purpose
    limitation
#
    ============================================================


import os
```

```
import gc
import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd

from dataclasses import dataclass
from typing import Dict, Tuple, List

from sklearn.model_selection import TimeSeriesSplit
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import (
    precision_recall_fscore_support,
    average_precision_score,
    roc_auc_score,
    confusion_matrix
)

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

SEED = 42
np.random.seed(SEED)
tf.random.set_seed(SEED)

# ============================================================

# 1) Configuration
# ============================================================


@dataclass
class CFG:
    base_path: str = "/kaggle/input/datasets/mrajaxnp/cert-
        insider-threat-detection-research"

    users_path: str = f"{base_path}/users.csv"
    logon_path: str = f"{base_path}/logon.csv"
    device_path: str = f"{base_path}/device.csv"
```

```
    psych_path: str = f"{base_path}/psychometric.csv"
    http_path: str = f"{base_path}/http.csv"
    email_path: str = f"{base_path}/email.csv"
    file_path: str = f"{base_path}/file.csv"

    time_col: str = "date"
    user_col: str = "user"

    # Proxy label derived from file.csv aggregated at user-day level
    label_mode: str = "to_removable_media"  # or "from_removable_media"

    # Temporal blocked folds
    n_splits: int = 5

    # file.csv chunking
    file_chunksize: int = 2_000_000

    # MANN architecture
    # Each domain has its own sub-network (module). Dimensions are set after feature build.
    module_hidden: Tuple[int, ...] = (64, 32)
    fusion_hidden: Tuple[int, ...] = (64,)
    dropout: float = 0.2

    # Training
    batch_size: int = 512
    epochs: int = 40
    patience: int = 6
    use_class_weights: bool = True

    # Optional: simple attention-like gating at fusion (lightweight, interpretable)
    use_gating: bool = True

cfg = CFG()

def ensure_exists(path: str):
    if not os.path.exists(path):
        raise FileNotFoundError(f"Missing file: {path}")

for p in [cfg.users_path, cfg.logon_path, cfg.device_path,
```

```
    cfg.psych_path, cfg.http_path, cfg.email_path, cfg.
    file_path]:
94      ensure_exists(p)
95
96  #
    ================================================================

97  # 2) Utilities: datetime + time features (privacy-minimised)
98  #
    ================================================================

99
100 def parse_datetime_safe(s: pd.Series) -> pd.Series:
101     return pd.to_datetime(s, errors="coerce",
            infer_datetime_format=True)
102
103 def add_time_features(df: pd.DataFrame, time_col: str) -> pd.
    DataFrame:
104     dt = df[time_col]
105     df["hour"] = dt.dt.hour
106     df["dayofweek"] = dt.dt.dayofweek
107     df["is_weekend"] = (df["dayofweek"] >= 5).astype(int)
108     df["is_after_hours"] = ((df["hour"] < 7) | (df["hour"] >=
            19)).astype(int)
109     df["date_only"] = dt.dt.floor("D")
110     return df
111
112 # Psychometric is ethically sensitive; default exclude unless
        explicitly justified.
113 INCLUDE_PSYCH = False
114
115 #
    ================================================================

116 # 3) Domain feature engineering (user-day aggregation)
117 #
    ================================================================

118
119 def agg_logon(path: str) -> pd.DataFrame:
120     df = pd.read_csv(path)
121     df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
122     df = df.dropna(subset=[cfg.time_col, cfg.user_col])
```

```
    df = add_time_features(df, cfg.time_col)

    out = df.groupby([cfg.user_col, "date_only"]).agg(
        logon_events=("activity", "size"),
        logon_after_hours=("is_after_hours", "sum"),
        logon_weekend=("is_weekend", "sum"),
        logon_unique_pc=("pc", pd.Series.nunique)
    ).reset_index()
    out["logon_after_hours_ratio"] = out["logon_after_hours"] / out["logon_events"].clip(lower=1)
    out["logon_weekend_ratio"] = out["logon_weekend"] / out["logon_events"].clip(lower=1)
    return out

def agg_device(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    out = df.groupby([cfg.user_col, "date_only"]).agg(
        device_events=("activity", "size"),
        device_after_hours=("is_after_hours", "sum"),
        device_weekend=("is_weekend", "sum"),
        device_unique_pc=("pc", pd.Series.nunique)
    ).reset_index()
    out["device_after_hours_ratio"] = out["device_after_hours"] / out["device_events"].clip(lower=1)
    out["device_weekend_ratio"] = out["device_weekend"] / out["device_events"].clip(lower=1)
    return out

def agg_http(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    # Data minimisation: do not parse URL content beyond aggregate counts.
    key = "url" if "url" in df.columns else cfg.user_col
    out = df.groupby([cfg.user_col, "date_only"]).agg(
        http_events=(key, "size"),
```

```
        http_after_hours=("is_after_hours", "sum"),
        http_weekend=("is_weekend", "sum")
    ).reset_index()
    out["http_after_hours_ratio"] = out["http_after_hours"] /
        out["http_events"].clip(lower=1)
    out["http_weekend_ratio"] = out["http_weekend"] / out["
       http_events"].clip(lower=1)
    return out

def agg_email(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    out = df.groupby([cfg.user_col, "date_only"]).agg(
        email_events=(cfg.user_col, "size"),
        email_after_hours=("is_after_hours", "sum"),
        email_weekend=("is_weekend", "sum")
    ).reset_index()
    out["email_after_hours_ratio"] = out["email_after_hours"]
        / out["email_events"].clip(lower=1)
    out["email_weekend_ratio"] = out["email_weekend"] / out["
       email_events"].clip(lower=1)
    return out

def agg_file_chunked(path: str, chunksize: int) -> Tuple[pd.
   DataFrame, pd.DataFrame]:
    feats_accum, labels_accum = [], []

    use_cols = [cfg.time_col, cfg.user_col, "pc", "filename",
        "activity", "to_removable_media", "
       from_removable_media"]
    for chunk in pd.read_csv(path, usecols=lambda c: c in
       use_cols, chunksize=chunksize):
        chunk[cfg.time_col] = parse_datetime_safe(chunk[cfg.
           time_col])
        chunk = chunk.dropna(subset=[cfg.time_col, cfg.
           user_col])
        chunk = add_time_features(chunk, cfg.time_col)

        for b in ["to_removable_media", "from_removable_media
           "]:
```

```
            if b in chunk.columns:
                if chunk[b].dtype == object:
                    chunk[b] = chunk[b].astype(str).str.lower
                        ().map({"true": 1, "false": 0})
                chunk[b] = chunk[b].fillna(0).astype(int)
            else:
                chunk[b] = 0

        grp = chunk.groupby([cfg.user_col, "date_only"]).agg(
            file_events=("activity", "size"),
            file_after_hours=("is_after_hours", "sum"),
            file_weekend=("is_weekend", "sum"),
            file_unique_pc=("pc", pd.Series.nunique),
            file_unique_filename=("filename", pd.Series.
               nunique),
            file_to_rm=("to_removable_media", "sum"),
            file_from_rm=("from_removable_media", "sum")
        ).reset_index()

        grp["file_after_hours_ratio"] = grp["file_after_hours
           "] / grp["file_events"].clip(lower=1)
        grp["file_weekend_ratio"] = grp["file_weekend"] / grp
           ["file_events"].clip(lower=1)

        feats_accum.append(grp)

        label_col = "file_to_rm" if cfg.label_mode == "
           to_removable_media" else "file_from_rm"
        lab = grp[[cfg.user_col, "date_only", label_col]].
           copy()
        lab["label"] = (lab[label_col] > 0).astype(int)
        lab = lab.drop(columns=[label_col])
        labels_accum.append(lab)

        del chunk, grp, lab
        gc.collect()

    file_feats = pd.concat(feats_accum, ignore_index=True)
    y_day = pd.concat(labels_accum, ignore_index=True)

    file_feats = file_feats.groupby([cfg.user_col, "date_only
       "], as_index=False).sum(numeric_only=True)
    y_day = y_day.groupby([cfg.user_col, "date_only"],
```

```
        as_index=False).max(numeric_only=True)
    return file_feats, y_day

# Build domain tables
logon_df = agg_logon(cfg.logon_path)
device_df = agg_device(cfg.device_path)
http_df  = agg_http(cfg.http_path)
email_df = agg_email(cfg.email_path)
file_df, y_day = agg_file_chunked(cfg.file_path, cfg.
    file_chunksize)

print("Domain shapes:",
      "logon", logon_df.shape,
      "device", device_df.shape,
      "http", http_df.shape,
      "email", email_df.shape,
      "file", file_df.shape,
      "labels", y_day.shape)

# Optional static tables
users_df = pd.read_csv(cfg.users_path)
psych_df = pd.read_csv(cfg.psych_path)

#
    ============================================================

# 4) Merge to user-day master index
#
    ============================================================


df_day = file_df.merge(logon_df, on=[cfg.user_col, "date_only
    "], how="left") \
               .merge(device_df, on=[cfg.user_col, "date_only
                  "], how="left") \
               .merge(http_df,  on=[cfg.user_col, "date_only"
                  ], how="left") \
               .merge(email_df, on=[cfg.user_col, "date_only"
                  ], how="left") \
               .merge(y_day,    on=[cfg.user_col, "date_only"
                  ], how="inner")

df_day = df_day.fillna(0)
```

```
261
262 # Merge minimal users (avoid unnecessary sensitive attributes
    )
263 if cfg.user_col in users_df.columns:
264     df_day = df_day.merge(users_df[[cfg.user_col]].
        drop_duplicates(), on=cfg.user_col, how="left")
265
266 if INCLUDE_PSYCH and (cfg.user_col in psych_df.columns):
267     # Psychometric is sensitive include only if justified
        and governed.
268     psych_num = psych_df.select_dtypes(include=[np.number]).
        copy()
269     psych_min = pd.concat([psych_df[[cfg.user_col]],
        psych_num], axis=1)
270     df_day = df_day.merge(psych_min, on=cfg.user_col, how="
        left").fillna(0)
271
272 df_day = df_day.sort_values("date_only").reset_index(drop=
    True)
273 print("Unified user-day:", df_day.shape, "| pos rate:",
    df_day["label"].mean())
274
275 #
    ============================================================

276 # 5) Define per-domain feature columns (MANN modules)
277 #
    ============================================================

278 # We build separate input tensors per behavioural domain.
279 # This is the core of modularity: each domain learns its own
    representation.
280
281 ID_COLS = [cfg.user_col, "date_only", "label"]
282
283 logon_cols  = [c for c in df_day.columns if c.startswith("
    logon_")]
284 device_cols = [c for c in df_day.columns if c.startswith("
    device_")]
285 http_cols   = [c for c in df_day.columns if c.startswith("
    http_")]
286 email_cols  = [c for c in df_day.columns if c.startswith("
    email_")]
```

```
287 file_cols   = [c for c in df_day.columns if c.startswith("
    file_") and c not in ["file_to_rm", "file_from_rm"]]
288
289 # Note: file_to_rm/file_from_rm are label-related counts;
    keep them out of features to reduce leakage risk.
290 # If you decide to keep them, you must justify why they are
    not direct proxies for the label.
291 # We exclude them here.
292
293 domain_cols = {
294     "logon": logon_cols,
295     "device": device_cols,
296     "http": http_cols,
297     "email": email_cols,
298     "file": file_cols
299 }
300
301 for k, v in domain_cols.items():
302     print(k, "features:", len(v))
303
304 y_all = df_day["label"].astype(int).values
305
306 #
    ============================================================

307 # 6) Model: Modular ANN + Fusion (+ optional gating)
308 #
    ============================================================


309
310 def build_module(name: str, input_dim: int) -> keras.Model:
311     inp = keras.Input(shape=(input_dim,), name=f"{name}_in")
312     x = inp
313     for i, hu in enumerate(cfg.module_hidden):
314         x = layers.Dense(hu, activation="relu", name=f"{name}
            _dense_{i+1}")(x)
315         x = layers.Dropout(cfg.dropout, name=f"{name}_drop_{i
            +1}")(x)
316     return keras.Model(inp, x, name=f"{name}_module")
317
318 def build_mann(domain_input_dims: Dict[str, int]) -> keras.
    Model:
319     inputs = {}
```

```
    embeddings = []

    for domain, dim in domain_input_dims.items():
        inp = keras.Input(shape=(dim,), name=f"{domain}_in")
        inputs[domain] = inp
        module = build_module(domain, dim)
        emb = module(inp)
        embeddings.append(emb)

    # Fusion
    fused = layers.Concatenate(name="fusion_concat")(
       embeddings)

    if cfg.use_gating:
        # Lightweight attention-like gating:
        # Produces a per-feature gate in [0,1] that can be
           inspected for interpretability.
        gate = layers.Dense(fused.shape[-1], activation="
           sigmoid", name="fusion_gate")(fused)
        fused = layers.Multiply(name="fusion_gated")([fused,
           gate])

    x = fused
    for i, hu in enumerate(cfg.fusion_hidden):
        x = layers.Dense(hu, activation="relu", name=f"
           fusion_dense_{i+1}")(x)
        x = layers.Dropout(cfg.dropout, name=f"fusion_drop_{i
           +1}")(x)

    out = layers.Dense(1, activation="sigmoid", name="out")(x
       )

    model = keras.Model(inputs=inputs, outputs=out, name="
       MANN")
    model.compile(optimizer=keras.optimizers.Adam(1e-3), loss
       ="binary_crossentropy")
    return model

def class_weights(y: np.ndarray) -> Dict[int, float]:
    n_pos = max(1, int(y.sum()))
    n_neg = max(1, int((y == 0).sum()))
    w_pos = (n_pos + n_neg) / (2.0 * n_pos)
    w_neg = (n_pos + n_neg) / (2.0 * n_neg)
```

```
354     return {0: w_neg, 1: w_pos}
355
356 def eval_metrics(y_true: np.ndarray, y_prob: np.ndarray, thr:
        float = 0.5) -> Dict[str, float]:
357     y_pred = (y_prob >= thr).astype(int)
358     p, r, f1, _ = precision_recall_fscore_support(y_true,
            y_pred, average="binary", zero_division=0)
359     pr_auc = average_precision_score(y_true, y_prob)
360
361     roc_auc = np.nan
362     if len(np.unique(y_true)) == 2:
363         roc_auc = roc_auc_score(y_true, y_prob)
364
365     tn, fp, fn, tp = confusion_matrix(y_true, y_pred).ravel()
366     return {
367         "precision": float(p),
368         "recall": float(r),
369         "f1": float(f1),
370         "pr_auc": float(pr_auc),
371         "roc_auc": float(roc_auc) if not np.isnan(roc_auc)
                else np.nan,
372         "tp": int(tp), "fp": int(fp), "tn": int(tn), "fn":
                int(fn)
373     }
374
375 #
        ============================================================

376 # 7) Blocked temporal folds (scalers fit per fold, per domain
        )
377 #
        ============================================================


378
379 def get_domain_arrays(df: pd.DataFrame, cols: Dict[str, List[
        str]]) -> Dict[str, np.ndarray]:
380     return {d: df[c].astype(np.float32).values for d, c in
            cols.items()}
381
382 X_domains_all = get_domain_arrays(df_day, domain_cols)
383
384 tscv = TimeSeriesSplit(n_splits=cfg.n_splits)
385 results = []
```

```
386
387 for fold, (train_idx, test_idx) in enumerate(tscv.split(
        df_day), start=1):
388
389     # Per-domain scaling on TRAIN only (prevents leakage and
            preserves modular semantics)
390     scalers = {}
391     X_train_domains = {}
392     X_test_domains = {}
393
394     for domain, X in X_domains_all.items():
395         scaler = StandardScaler()
396         X_train = scaler.fit_transform(X[train_idx])
397         X_test  = scaler.transform(X[test_idx])
398         scalers[domain] = scaler
399         X_train_domains[domain] = X_train.astype(np.float32)
400         X_test_domains[domain]  = X_test.astype(np.float32)
401
402     y_train = y_all[train_idx]
403     y_test  = y_all[test_idx]
404
405     # Temporal validation from tail of training period
406     n_train = len(train_idx)
407     val_frac = 0.2
408     split = int((1.0 - val_frac) * n_train)
409
410     tr_slice = slice(0, split)
411     val_slice = slice(split, n_train)
412
413     X_tr = {d: X_train_domains[d][tr_slice] for d in
            X_train_domains}
414     X_val = {d: X_train_domains[d][val_slice] for d in
            X_train_domains}
415     y_tr, y_val = y_train[tr_slice], y_train[val_slice]
416
417     # Build model
418     domain_input_dims = {d: X_tr[d].shape[1] for d in X_tr}
419     model = build_mann(domain_input_dims)
420
421     callbacks = [
422         keras.callbacks.EarlyStopping(
423             monitor="val_loss", patience=cfg.patience,
                    restore_best_weights=True
```

```
        )
    ]

    cw = class_weights(y_tr) if cfg.use_class_weights else
       None

    model.fit(
        X_tr, y_tr,
        validation_data=(X_val, y_val),
        epochs=cfg.epochs,
        batch_size=cfg.batch_size,
        verbose=0,
        class_weight=cw,
        callbacks=callbacks
    )

    y_prob = model.predict(X_test_domains, batch_size=cfg.
       batch_size, verbose=0).ravel()
    m = eval_metrics(y_test, y_prob, thr=0.5)
    m["fold"] = fold
    m["n_test"] = int(len(test_idx))
    m["pos_rate_test"] = float(y_test.mean())

    print(
        f"Fold {fold}/{cfg.n_splits} | "
        f"F1={m['f1']:.3f} PR-AUC={m['pr_auc']:.3f} "
        f"P={m['precision']:.3f} R={m['recall']:.3f} | "
        f"TP={m['tp']} FP={m['fp']} FN={m['fn']} TN={m['tn']}
           "
    )

    results.append(m)

    # Cleanup
    tf.keras.backend.clear_session()
    del model, scalers, X_train_domains, X_test_domains, X_tr
       , X_val
    gc.collect()

res_df = pd.DataFrame(results)
print("\nPer-fold results:\n", res_df)

summary = res_df[["precision", "recall", "f1", "pr_auc", "
```

```
    roc_auc"]].agg(["mean", "std"]).T
summary.columns = ["mean", "std"]
print("\nSummary (Mean    Std):\n", summary)

# Export artefacts (aggregated metrics only)
res_path = "/kaggle/working/mann_cert_temporal_folds.csv"
sum_path = "/kaggle/working/mann_cert_summary.csv"
res_df.to_csv(res_path, index=False)
summary.to_csv(sum_path)

print("\nSaved artefacts:")
print(" -", res_path)
print(" -", sum_path)

# ============================================================

# 8) Ethics-focused model card (Appendix-friendly)
# ============================================================


model_card = f"""
MANN Model Card (CERT research artefact)
- Intended use: Research-oriented risk scoring from multi-domain behavioural logs.
- Data: CERT logs aggregated into user-day signals (logon/device/http/email/file).
- Label: Proxy from file.csv ({cfg.label_mode}) at user-day level.
- Architecture: Domain-specific modules + fusion layer (optional gating for transparency).
- Evaluation: Blocked temporal folds to preserve chronology and reduce leakage.
- Primary metrics: PR-AUC, Precision, Recall, F1 (PR-AUC prioritised under imbalance).
- Key risks: Privacy intrusion, proxy-label bias, false positives, behaviour-role confounding.
- Mitigations: Data minimisation, per-fold training-only scaling, temporal splitting,
  human-in-the-loop review and governance controls.
- Prohibited use: Automated punitive action or covert surveillance without due process.
```

```
"""
print(model_card)
```

Listing D.2: MANN (Modular ANN) implementation for CERT with blocked temporal folds.

D.3 Artefact—Multi-Layered Machine Learning (MLML) Implementation

The artefact in this section provides a reproducible implementation of the Multi-Layered Machine Learning (MLML) framework used to model insider-risk detection as a staged learning process over heterogeneous behavioural domains. The implementation adopts a layered design in which domain-specific learners generate intermediate risk signals, which are subsequently fused into higher-level meta-features and a final decision layer. In alignment with the methodological discipline of this thesis, the artefact uses privacy-minimised user-day behavioural aggregates, blocked temporal folds, and leakage-aware preprocessing to support robust and comparable evaluation. The implementation is provided for research and reproducibility purposes, and any adaptation to operational environments should be preceded by ethical review, fairness assessment, and governance controls appropriate to sensitive behavioural analytics.

```
#
    ============================================================

# Appendix D3 Artefact: Multi-Layered Machine Learning (MLML)
# CERT Implementation (Blocked Temporal Folds)
#
    ============================================================



import os
import gc
import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd

from dataclasses import dataclass
from typing import Dict, List, Tuple

```

```
from sklearn.model_selection import TimeSeriesSplit
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import (
    precision_recall_fscore_support,
    average_precision_score,
    roc_auc_score,
    confusion_matrix
)
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier,
   GradientBoostingClassifier

SEED = 42
np.random.seed(SEED)

#
   ============================================================

# 1) Configuration
#
   ============================================================


@dataclass
class CFG:
    # Update this if your Kaggle mount folder differs
    base_path: str = "/kaggle/input/cert-insider-threat-
       detection-research"

    users_path: str = f"{base_path}/users.csv"
    logon_path: str = f"{base_path}/logon.csv"
    device_path: str = f"{base_path}/device.csv"
    psych_path: str = f"{base_path}/psychometric.csv"
    http_path: str = f"{base_path}/http.csv"
    email_path: str = f"{base_path}/email.csv"
    file_path: str = f"{base_path}/file.csv"

    time_col: str = "date"
    user_col: str = "user"

    # Proxy label (user-day) derived from file.csv
    # Recommended for a clear binary task in CERT:
    #   "to_removable_media" or "from_removable_media"
```

```
54      label_mode: str = "to_removable_media"
55
56      # Temporal blocked evaluation
57      n_splits: int = 5
58
59      # Large file chunking
60      file_chunksize: int = 2_000_000
61
62      # Layer 1 base learners (per domain)
63      # Using simple, robust models for reproducibility and
            interpretability.
64      rf_estimators: int = 120
65      rf_max_depth: int = 8
66
67      # Layer 3 meta-classifier
68      meta_model_type: str = "logreg"  # options: "logreg", "
            gbm", "rf"
69
70      # Threshold for binary decisions (deployment threshold
            should be calibrated)
71      decision_threshold: float = 0.5
72
73  cfg = CFG()
74
75  #
        ============================================================
76  # 2) Path checks + simple auto-detection (optional robustness
        )
77  #
        ============================================================
78
79  def detect_base_path(candidates: List[str]) -> str:
80      for p in candidates:
81          if os.path.exists(p):
82              return p
83      if os.path.exists("/kaggle/input"):
84          available = [os.path.join("/kaggle/input", d) for d
                in os.listdir("/kaggle/input")]
85          raise FileNotFoundError(
86              "Could not detect CERT dataset base path.\n"
87              "Tried:\n" + "\n".join(candidates) +
```

```
            "\nAvailable folders:\n" + "\n".join(available)
        )
    raise FileNotFoundError("Kaggle input folder not found.")

# Optional auto-detect: uncomment if needed
# cfg.base_path = detect_base_path([
#     "/kaggle/input/cert-insider-threat-detection-research",
#     "/kaggle/input/datasets/mrajaxnp/cert-insider-threat-detection-research"
# ])
# cfg.users_path = f"{cfg.base_path}/users.csv"
# cfg.logon_path = f"{cfg.base_path}/logon.csv"
# cfg.device_path = f"{cfg.base_path}/device.csv"
# cfg.psych_path = f"{cfg.base_path}/psychometric.csv"
# cfg.http_path = f"{cfg.base_path}/http.csv"
# cfg.email_path = f"{cfg.base_path}/email.csv"
# cfg.file_path = f"{cfg.base_path}/file.csv"

def ensure_exists(path: str):
    if not os.path.exists(path):
        raise FileNotFoundError(f"Missing file: {path}")

for p in [cfg.users_path, cfg.logon_path, cfg.device_path, cfg.psych_path, cfg.http_path, cfg.email_path, cfg.file_path]:
    ensure_exists(p)

# ============================================================

# 3) Utilities: parsing and time-derived features (minimised)
# ============================================================


def parse_datetime_safe(s: pd.Series) -> pd.Series:
    return pd.to_datetime(s, errors="coerce")

def add_time_features(df: pd.DataFrame, time_col: str) -> pd.DataFrame:
    dt = df[time_col]
    df["hour"] = dt.dt.hour
    df["dayofweek"] = dt.dt.dayofweek
```

```
    df["is_weekend"] = (df["dayofweek"] >= 5).astype(int)
    df["is_after_hours"] = ((df["hour"] < 7) | (df["hour"] >=
        19)).astype(int)
    df["date_only"] = dt.dt.floor("D")
    return df

def normalise_user_key(df: pd.DataFrame, user_col_target: str
    = "user") -> pd.DataFrame:
    # CERT variants may use user_id in some files (e.g.,
       users.csv, psychometric.csv)
    if user_col_target in df.columns:
        return df
    if "user_id" in df.columns:
        return df.rename(columns={"user_id": user_col_target
           })
    raise ValueError(f"Expected '{user_col_target}' or '
       user_id' column. Found: {list(df.columns)}")

#
   ============================================================

# 4) Load users/psychometric (minimal use)
#
   ============================================================


def load_users(path: str) -> pd.DataFrame:
    users = pd.read_csv(path)
    users = normalise_user_key(users, "user")
    # Data minimisation: keep only user identifier unless
       static attributes are justified.
    return users

def load_psych(path: str) -> pd.DataFrame:
    psych = pd.read_csv(path)
    psych = normalise_user_key(psych, "user")
    # Psychometric data is sensitive; excluded by default
       from modelling.
    return psych

users_df = load_users(cfg.users_path)
psych_df = load_psych(cfg.psych_path)

```

```
155 # Default: do NOT include psychometric features unless ethics
    /governance are explicit.
156 INCLUDE_PSYCH = False
157
158 #
    ============================================================

159 # 5) Per-domain feature engineering (user-day aggregation)
160 #
    ============================================================

161 # Ethical note:
162 # We prioritise aggregate counts and timing patterns rather
    than content
163 # inspection (e.g., email body text, file contents, URL
    semantics).
164
165 def agg_logon(path: str) -> pd.DataFrame:
166     df = pd.read_csv(path)
167     df = normalise_user_key(df, cfg.user_col)
168     df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
169     df = df.dropna(subset=[cfg.time_col, cfg.user_col])
170     df = add_time_features(df, cfg.time_col)
171
172     out = df.groupby([cfg.user_col, "date_only"]).agg(
173         logon_events=("activity", "size") if "activity" in df
            .columns else (cfg.user_col, "size"),
174         logon_after_hours=("is_after_hours", "sum"),
175         logon_weekend=("is_weekend", "sum"),
176         logon_unique_pc=("pc", pd.Series.nunique) if "pc" in
            df.columns else (cfg.user_col, "nunique")
177     ).reset_index()
178
179     out["logon_after_hours_ratio"] = out["logon_after_hours"]
        / out["logon_events"].clip(lower=1)
180     out["logon_weekend_ratio"] = out["logon_weekend"] / out["
       logon_events"].clip(lower=1)
181     return out
182
183 def agg_device(path: str) -> pd.DataFrame:
184     df = pd.read_csv(path)
185     df = normalise_user_key(df, cfg.user_col)
186     df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
```

```
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    out = df.groupby([cfg.user_col, "date_only"]).agg(
        device_events=("activity", "size") if "activity" in
           df.columns else (cfg.user_col, "size"),
        device_after_hours=("is_after_hours", "sum"),
        device_weekend=("is_weekend", "sum"),
        device_unique_pc=("pc", pd.Series.nunique) if "pc" in
            df.columns else (cfg.user_col, "nunique")
    ).reset_index()

    out["device_after_hours_ratio"] = out["device_after_hours
       "] / out["device_events"].clip(lower=1)
    out["device_weekend_ratio"] = out["device_weekend"] / out
       ["device_events"].clip(lower=1)
    return out

def agg_http(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df = normalise_user_key(df, cfg.user_col)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    count_key = "url" if "url" in df.columns else cfg.
       user_col
    out = df.groupby([cfg.user_col, "date_only"]).agg(
        http_events=(count_key, "size"),
        http_after_hours=("is_after_hours", "sum"),
        http_weekend=("is_weekend", "sum")
    ).reset_index()

    out["http_after_hours_ratio"] = out["http_after_hours"] /
        out["http_events"].clip(lower=1)
    out["http_weekend_ratio"] = out["http_weekend"] / out["
       http_events"].clip(lower=1)
    return out

def agg_email(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df = normalise_user_key(df, cfg.user_col)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
```

```
223     df = df.dropna(subset=[cfg.time_col, cfg.user_col])
224     df = add_time_features(df, cfg.time_col)
225 
226     out = df.groupby([cfg.user_col, "date_only"]).agg(
227         email_events=(cfg.user_col, "size"),
228         email_after_hours=("is_after_hours", "sum"),
229         email_weekend=("is_weekend", "sum")
230     ).reset_index()
231 
232     out["email_after_hours_ratio"] = out["email_after_hours"]
            / out["email_events"].clip(lower=1)
233     out["email_weekend_ratio"] = out["email_weekend"] / out["
           email_events"].clip(lower=1)
234     return out
235 
236 def agg_file_chunked(path: str, chunksize: int) -> Tuple[pd.
       DataFrame, pd.DataFrame]:
237     """
238     Returns:
239       file_feats: user-day aggregated file-domain features
240       labels: user-day binary labels (proxy from removable
             media activity)
241     """
242     feats_accum, labels_accum = [], []
243 
244     use_cols = [cfg.time_col, cfg.user_col, "pc", "filename",
            "activity",
245                 "to_removable_media", "from_removable_media"]
246 
247     for chunk in pd.read_csv(path, usecols=lambda c: c in
           use_cols, chunksize=chunksize):
248         chunk = normalise_user_key(chunk, cfg.user_col)
249         chunk[cfg.time_col] = parse_datetime_safe(chunk[cfg.
              time_col])
250         chunk = chunk.dropna(subset=[cfg.time_col, cfg.
              user_col])
251         chunk = add_time_features(chunk, cfg.time_col)
252 
253         # Robust boolean conversion (CERT variants may store
              True/False as strings)
254         for b in ["to_removable_media", "from_removable_media
              "]:
255             if b in chunk.columns:
```

```
                if chunk[b].dtype == object:
                    chunk[b] = chunk[b].astype(str).str.lower
                        ().map({"true": 1, "false": 0})
                chunk[b] = chunk[b].fillna(0).astype(int)
            else:
                chunk[b] = 0

        grp = chunk.groupby([cfg.user_col, "date_only"]).agg(
            file_events=("activity", "size"),
            file_after_hours=("is_after_hours", "sum"),
            file_weekend=("is_weekend", "sum"),
            file_unique_pc=("pc", pd.Series.nunique),
            file_unique_filename=("filename", pd.Series.
                nunique),
            file_to_rm=("to_removable_media", "sum"),
            file_from_rm=("from_removable_media", "sum")
        ).reset_index()

        grp["file_after_hours_ratio"] = grp["file_after_hours
            "] / grp["file_events"].clip(lower=1)
        grp["file_weekend_ratio"] = grp["file_weekend"] / grp
            ["file_events"].clip(lower=1)

        feats_accum.append(grp)

        label_col = "file_to_rm" if cfg.label_mode == "
            to_removable_media" else "file_from_rm"
        lab = grp[[cfg.user_col, "date_only", label_col]].
            copy()
        lab["label"] = (lab[label_col] > 0).astype(int)
        lab = lab.drop(columns=[label_col])
        labels_accum.append(lab)

        del chunk, grp, lab
        gc.collect()

    file_feats = pd.concat(feats_accum, ignore_index=True)
    labels = pd.concat(labels_accum, ignore_index=True)

    # Re-aggregate across chunk boundaries
    file_feats = file_feats.groupby([cfg.user_col, "date_only
        "], as_index=False).sum(numeric_only=True)
    labels = labels.groupby([cfg.user_col, "date_only"],
```

```
        as_index=False).max(numeric_only=True)
292
293     return file_feats, labels
294
295 # Build domain tables
296 logon_df = agg_logon(cfg.logon_path)
297 device_df = agg_device(cfg.device_path)
298 http_df  = agg_http(cfg.http_path)
299 email_df = agg_email(cfg.email_path)
300 file_df, y_day = agg_file_chunked(cfg.file_path, cfg.
    file_chunksize)
301
302 # Merge into user-day master table
303 df_day = file_df.merge(logon_df, on=[cfg.user_col, "date_only
    "], how="left") \
304                 .merge(device_df, on=[cfg.user_col, "
                       date_only"], how="left") \
305                 .merge(http_df,  on=[cfg.user_col, "date_only
                       "], how="left") \
306                 .merge(email_df, on=[cfg.user_col, "date_only
                       "], how="left") \
307                 .merge(y_day,    on=[cfg.user_col, "date_only
                       "], how="inner")
308
309 df_day = df_day.fillna(0)
310
311 # Minimal user merge (identifier only)
312 if "user" in users_df.columns:
313     df_day = df_day.merge(users_df[["user"]].drop_duplicates
        (), on="user", how="left")
314
315 # Optional psychometric merge (sensitive; OFF by default)
316 if INCLUDE_PSYCH and ("user" in psych_df.columns):
317     psych_num = psych_df.select_dtypes(include=[np.number]).
        copy()
318     psych_min = pd.concat([psych_df[["user"]], psych_num],
        axis=1)
319     df_day = df_day.merge(psych_min, on="user", how="left").
        fillna(0)
320
321 # Chronological order required for blocked temporal folds
322 df_day = df_day.sort_values("date_only").reset_index(drop=
    True)
```

```
323
324 print("Unified user-day dataset shape:", df_day.shape)
325 print("Positive rate:", float(df_day["label"].mean()))
326
327 # ============================================================
328 # 6) Define domain feature groups (Layer 1 inputs)
329 # ============================================================
330 # Direct label proxies are excluded from file features to reduce leakage:
331 # file_to_rm and file_from_rm are derived from the same signal used for labels.
332
333 logon_cols  = [c for c in df_day.columns if c.startswith("logon_")]
334 device_cols = [c for c in df_day.columns if c.startswith("device_")]
335 http_cols   = [c for c in df_day.columns if c.startswith("http_")]
336 email_cols  = [c for c in df_day.columns if c.startswith("email_")]
337 file_cols   = [c for c in df_day.columns if c.startswith("file_") and c not in ["file_to_rm", "file_from_rm"]]
338
339 domain_cols = {
340     "logon": logon_cols,
341     "device": device_cols,
342     "http": http_cols,
343     "email": email_cols,
344     "file": file_cols,
345 }
346
347 for d, cols in domain_cols.items():
348     print(f"{d}: {len(cols)} features")
349
350 y_all = df_day["label"].astype(int).values
351
352 # ============================================================
```

```
# 7) MLML Architecture (three layers)
#    ============================================================

# Layer 1: Domain-specific models (one per domain)
# Layer 2: Meta-feature fusion (stack domain probabilities + selected stats)
# Layer 3: Final meta-classifier

def build_base_model(seed_offset: int = 0):
    # RandomForest chosen for robustness + nonlinearity + ease of use on tabular signals.
    return RandomForestClassifier(
        n_estimators=cfg.rf_estimators,
        max_depth=cfg.rf_max_depth,
        random_state=SEED + seed_offset,
        n_jobs=-1,
        class_weight="balanced_subsample"
    )

def build_meta_model():
    if cfg.meta_model_type == "logreg":
        # Interpretable meta-layer; supports calibrated linear fusion of domain signals.
        return LogisticRegression(
            max_iter=1000,
            class_weight="balanced",
            random_state=SEED
        )
    elif cfg.meta_model_type == "gbm":
        return GradientBoostingClassifier(random_state=SEED)
    elif cfg.meta_model_type == "rf":
        return RandomForestClassifier(
            n_estimators=150,
            max_depth=6,
            random_state=SEED,
            n_jobs=-1,
            class_weight="balanced_subsample"
        )
    else:
        raise ValueError(f"Unsupported meta_model_type: {cfg.meta_model_type}")

```

```
390 def get_domain_arrays(df: pd.DataFrame, domain_cols: Dict[str
    , List[str]]) -> Dict[str, np.ndarray]:
391     return {d: df[cols].astype(np.float32).values for d, cols
        in domain_cols.items()}
392
393 def compute_metrics(y_true: np.ndarray, y_prob: np.ndarray,
    thr: float = 0.5) -> Dict[str, float]:
394     y_pred = (y_prob >= thr).astype(int)
395
396     p, r, f1, _ = precision_recall_fscore_support(
397         y_true, y_pred, average="binary", zero_division=0
398     )
399     pr_auc = average_precision_score(y_true, y_prob)
400
401     roc_auc = np.nan
402     if len(np.unique(y_true)) == 2:
403         roc_auc = roc_auc_score(y_true, y_prob)
404
405     tn, fp, fn, tp = confusion_matrix(y_true, y_pred).ravel()
406
407     return {
408         "precision": float(p),
409         "recall": float(r),
410         "f1": float(f1),
411         "pr_auc": float(pr_auc),
412         "roc_auc": float(roc_auc) if not np.isnan(roc_auc)
            else np.nan,
413         "tp": int(tp), "fp": int(fp), "tn": int(tn), "fn":
            int(fn)
414     }
415
416 #
    ============================================================
417 # 8) Blocked temporal evaluation (chronology-preserving)
418 #
    ============================================================
419
420 X_domains_all = get_domain_arrays(df_day, domain_cols)
421 tscv = TimeSeriesSplit(n_splits=cfg.n_splits)
422
423 fold_results = []
```

```
424
425 for fold, (train_idx, test_idx) in enumerate(tscv.split(
       df_day), start=1):
426     print(f"\n========== Fold {fold}/{cfg.n_splits}
          ==========")
427
428     y_train = y_all[train_idx]
429     y_test  = y_all[test_idx]
430
431     # ----------------------------
432     # 8.1 Per-domain scaling (Layer 1 preprocessing)
433     # Fit scalers on TRAIN only to prevent temporal leakage
434     # ----------------------------
435     scalers = {}
436     X_train_domains = {}
437     X_test_domains = {}
438
439     for d, X in X_domains_all.items():
440         scaler = StandardScaler()
441         X_train = scaler.fit_transform(X[train_idx])
442         X_test  = scaler.transform(X[test_idx])
443         scalers[d] = scaler
444         X_train_domains[d] = X_train
445         X_test_domains[d] = X_test
446
447     # ----------------------------
448     # 8.2 Layer 1: domain-specific models
449     # Each domain model produces probabilities
450     # ----------------------------
451     domain_models = {}
452     train_domain_probs = {}
453     test_domain_probs = {}
454
455     for i, d in enumerate(domain_cols.keys()):
456         Xtr_d = X_train_domains[d]
457         Xte_d = X_test_domains[d]
458
459         # If a domain has no features (schema variation),
              skip gracefully
460         if Xtr_d.shape[1] == 0:
461             # Use neutral probability if no domain features
                  are available
462             train_domain_probs[d] = np.full(len(Xtr_d),
```

```
            y_train.mean() if len(y_train) else 0.0)
463         test_domain_probs[d] = np.full(len(Xte_d),
               y_train.mean() if len(y_train) else 0.0)
464         print(f"[WARN] Domain '{d}' has 0 features; using
                neutral prior.")
465         continue
466
467     model_d = build_base_model(seed_offset=i)
468     model_d.fit(Xtr_d, y_train)
469
470     p_tr = model_d.predict_proba(Xtr_d)[:, 1]
471     p_te = model_d.predict_proba(Xte_d)[:, 1]
472
473     domain_models[d] = model_d
474     train_domain_probs[d] = p_tr
475     test_domain_probs[d] = p_te
476
477     print(f"Layer1 domain='{d}' trained | features={Xtr_d
           .shape[1]}")
478
479 # ----------------------------
480 # 8.3 Layer 2: meta-feature fusion
481 # Create stacked features from domain probabilities
482 # and simple uncertainty proxies (distance from 0.5)
483 # ----------------------------
484 layer2_train = pd.DataFrame({
485     f"{d}_prob": train_domain_probs[d] for d in
           domain_cols.keys()
486 })
487 layer2_test = pd.DataFrame({
488     f"{d}_prob": test_domain_probs[d] for d in
           domain_cols.keys()
489 })
490
491 # Additional fused descriptors (helpful and interpretable
       )
492 for df_meta in [layer2_train, layer2_test]:
493     prob_cols = [c for c in df_meta.columns if c.endswith
           ("_prob")]
494     df_meta["prob_mean"] = df_meta[prob_cols].mean(axis
           =1)
495     df_meta["prob_max"] = df_meta[prob_cols].max(axis=1)
496     df_meta["prob_min"] = df_meta[prob_cols].min(axis=1)
```

```
497         df_meta["prob_std"] = df_meta[prob_cols].std(axis=1).
               fillna(0.0)
498         # Disagreement/uncertainty proxy (high when domains
               disagree)
499         df_meta["disagreement"] = df_meta["prob_max"] -
               df_meta["prob_min"]
500
501     # ----------------------------
502     # 8.4 Layer 3: final meta-classifier
503     # ----------------------------
504     meta_model = build_meta_model()
505     meta_model.fit(layer2_train.values, y_train)
506
507     y_prob = meta_model.predict_proba(layer2_test.values)[:,
            1]
508
509     m = compute_metrics(y_test, y_prob, thr=cfg.
            decision_threshold)
510     m["fold"] = fold
511     m["n_test"] = int(len(test_idx))
512     m["pos_rate_test"] = float(y_test.mean())
513     m["meta_model"] = cfg.meta_model_type
514
515     # Optional interpretability snapshot: coefficients if
            logistic regression
516     if cfg.meta_model_type == "logreg":
517         coef_map = dict(zip(layer2_train.columns.tolist(),
               meta_model.coef_[0]))
518         # Store only a compact summary (not full arrays) to
               keep artefact tidy
519         m["top_positive_signal"] = max(coef_map, key=coef_map
               .get)
520         m["top_negative_signal"] = min(coef_map, key=coef_map
               .get)
521     else:
522         m["top_positive_signal"] = ""
523         m["top_negative_signal"] = ""
524
525     print(
526         f"Fold {fold} | "
527         f"P={m['precision']:.3f} R={m['recall']:.3f} F1={m['
               f1']:.3f} "
528         f"PR-AUC={m['pr_auc']:.3f} ROC-AUC={m['roc_auc']:.3f
```

```
            if not np.isnan(m['roc_auc']) else float('nan')}"
529     )
530     print(
531         f"Confusion: TP={m['tp']} FP={m['fp']} FN={m['fn']}
            TN={m['tn']} | "
532         f"PosRate(test)={m['pos_rate_test']:.4f}"
533     )
534
535     fold_results.append(m)
536
537     # Cleanup to reduce Kaggle memory pressure
538     del scalers, X_train_domains, X_test_domains,
        domain_models
539     del layer2_train, layer2_test, meta_model,
        train_domain_probs, test_domain_probs
540     gc.collect()
541
542 #
    ============================================================

543 # 9) Aggregate results (Mean    Std)
544 #
    ============================================================


545
546 res_df = pd.DataFrame(fold_results)
547 print("\nPer-fold results:")
548 print(res_df)
549
550 metric_cols = ["precision", "recall", "f1", "pr_auc", "
    roc_auc"]
551 summary = res_df[metric_cols].agg(["mean", "std"]).T.
    reset_index()
552 summary.columns = ["metric", "mean", "std"]
553
554 print("\nSummary (Mean    Std):")
555 print(summary)
556
557 #
    ============================================================

558 # 10) Export artefacts (aggregated metrics only)
559 #
```

```
    ============================================================

# Ethical note:
# Export only aggregated metrics unless governance explicitly
    permits
# controlled access to intermediate artefacts.

res_path = "/kaggle/working/mlml_cert_temporal_folds.csv"
sum_path = "/kaggle/working/mlml_cert_summary.csv"
res_df.to_csv(res_path, index=False)
summary.to_csv(sum_path, index=False)

print("\nSaved artefacts:")
print(" -", res_path)
print(" -", sum_path)

#
    ============================================================

# 11) Appendix-friendly ethics-oriented model card
#
    ============================================================


model_card = f"""
MLML Model Card (CERT research artefact)
- Intended use: Research-oriented multi-stage risk scoring
   using aggregated behavioural logs.
- Data: User-day features from CERT domains (logon, device,
   http, email, file).
- Label: Proxy from file.csv ({cfg.label_mode}) aggregated at
    user-day level.
- Architecture:
    Layer 1 = domain-specific base learners,
    Layer 2 = meta-feature fusion (stacked domain
       probabilities + disagreement features),
    Layer 3 = final meta-classifier.
- Evaluation: Blocked temporal folds to preserve chronology
   and reduce leakage.
- Primary metrics: PR-AUC, Precision, Recall, F1 (PR-AUC
   prioritised under class imbalance).
- Key risks: Proxy-label bias, privacy intrusion, role
   confounding, false positives.
```

```
- Mitigations: Data minimisation, chronology-preserving evaluation, human-in-the-loop review,
  threshold calibration, governance and auditing.
- Prohibited use: Automated punitive action or covert employee surveillance without due process.
"""
print(model_card)
```

Listing D.3: Multi-Layered Machine Learning (MLML) implementation for CERT with blocked temporal folds.

D.4 Artefact—Guarding the Gates Implementation

The artefact in this section provides a reproducible implementation of the *Guarding the Gates* adversarial robustness evaluation pipeline for CERT-style insider-threat detection. The implementation adopts a privacy-minimised user-day feature representation, chronology-preserving blocked temporal folds, and bounded adversarial perturbations in scaled feature space to evaluate model degradation under constrained adversarial pressure. The artefact is presented for defensive research and validation purposes only, with explicit safeguards and ethical use constraints.

```
#
    ============================================================

# Appendix D4 Artefact: Guarding the Gates (Adversarial ML)
# CERT Kaggle Implementation (Blocked Temporal Folds)
#
    ============================================================

# PURPOSE:
# This artefact evaluates adversarial robustness of a tabular
    insider-
# threat detector using bounded, research-safe perturbations
    on
# aggregated behavioural features (user-day level).
#
# ETHICAL USE NOTICE:
# This code is for defensive research and robustness
    evaluation only.
# It must not be used to evade monitoring, conceal malicious
    activity,
```

```
# or facilitate unauthorized access. Before adapting to real data:
#   - ensure lawful basis/consent and transparent governance
#   - minimise data and avoid unnecessary sensitive feature extraction
#   - use human-in-the-loop review (no automated punitive action)
#   - calibrate thresholds and assess fairness impacts
#   - secure outputs and audit all experimentation
# ============================================================


import os
import gc
import warnings
warnings.filterwarnings("ignore")

import numpy as np
import pandas as pd

from dataclasses import dataclass
from typing import Dict, List, Tuple

from sklearn.model_selection import TimeSeriesSplit
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import (
    precision_recall_fscore_support,
    average_precision_score,
    roc_auc_score,
    confusion_matrix
)

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

SEED = 42
np.random.seed(SEED)
tf.random.set_seed(SEED)

# ============================================================
```

```
# 1) Configuration
#
    ============================================================


@dataclass
class CFG:
    # Adjust to your Kaggle mount path if needed
    base_path: str = "/kaggle/input/cert-insider-threat-
        detection-research"

    users_path: str = f"{base_path}/users.csv"
    logon_path: str = f"{base_path}/logon.csv"
    device_path: str = f"{base_path}/device.csv"
    psych_path: str = f"{base_path}/psychometric.csv"
    http_path: str = f"{base_path}/http.csv"
    email_path: str = f"{base_path}/email.csv"
    file_path: str = f"{base_path}/file.csv"

    time_col: str = "date"
    user_col: str = "user"
    label_mode: str = "to_removable_media"  # or "
        from_removable_media"

    # Temporal blocked evaluation
    n_splits: int = 5
    file_chunksize: int = 2_000_000

    # Base detector (simple ANN for reproducible robustness
        tests)
    hidden_units: Tuple[int, ...] = (128, 64)
    dropout: float = 0.2
    batch_size: int = 512
    epochs: int = 35
    patience: int = 5
    use_class_weights: bool = True

    # Adversarial evaluation settings (bounded perturbation,
        tabular-safe)
    adv_eps: float = 0.08              # perturbation
        magnitude in scaled feature space
    adv_alpha: float = 0.02            # step size for
```

```
        iterative attack
    adv_steps: int = 5                  # PGD-like steps
    adv_clip_min: float = -5.0          # clip in scaled space
    adv_clip_max: float = 5.0
    attack_positive_only: bool = True  # evaluate attack on
        positive-class samples (harder case)
    use_adversarial_training: bool = False  # defensive fine-
        tuning option
    adv_train_fraction: float = 0.3     # fraction of each
        batch replaced by adversarial examples

cfg = CFG()

# ============================================================

# 2) Path/schema robustness helpers
# ============================================================


def detect_base_path(candidates: List[str]) -> str:
    for p in candidates:
        if os.path.exists(p):
            return p
    if os.path.exists("/kaggle/input"):
        available = [os.path.join("/kaggle/input", d) for d
            in os.listdir("/kaggle/input")]
        raise FileNotFoundError(
            "Could not detect CERT dataset base path.\n"
            "Tried:\n" + "\n".join(candidates) +
            "\nAvailable folders:\n" + "\n".join(available)
        )
    raise FileNotFoundError("Kaggle input folder not found.")

def ensure_exists(path: str):
    if not os.path.exists(path):
        raise FileNotFoundError(f"Missing file: {path}")

def normalise_user_key(df: pd.DataFrame, user_col_target: str
    = "user") -> pd.DataFrame:
    if user_col_target in df.columns:
        return df
```

```
    if "user_id" in df.columns:
        return df.rename(columns={"user_id": user_col_target})
    raise ValueError(f"Expected '{user_col_target}' or 'user_id'. Found columns: {list(df.columns)}")

# Optional auto-detect (uncomment if needed)
# cfg.base_path = detect_base_path([
#     "/kaggle/input/cert-insider-threat-detection-research",
#     "/kaggle/input/datasets/mrajaxnp/cert-insider-threat-detection-research"
# ])
# cfg.users_path = f"{cfg.base_path}/users.csv"
# cfg.logon_path = f"{cfg.base_path}/logon.csv"
# cfg.device_path = f"{cfg.base_path}/device.csv"
# cfg.psych_path = f"{cfg.base_path}/psychometric.csv"
# cfg.http_path = f"{cfg.base_path}/http.csv"
# cfg.email_path = f"{cfg.base_path}/email.csv"
# cfg.file_path = f"{cfg.base_path}/file.csv"

for p in [cfg.users_path, cfg.logon_path, cfg.device_path, cfg.psych_path, cfg.http_path, cfg.email_path, cfg.file_path]:
    ensure_exists(p)

#
    ================================================================

# 3) Utilities: datetime + privacy-minimised time features
#
    ================================================================


def parse_datetime_safe(s: pd.Series) -> pd.Series:
    return pd.to_datetime(s, errors="coerce")

def add_time_features(df: pd.DataFrame, time_col: str) -> pd.DataFrame:
    dt = df[time_col]
    df["hour"] = dt.dt.hour
    df["dayofweek"] = dt.dt.dayofweek
    df["is_weekend"] = (df["dayofweek"] >= 5).astype(int)
    df["is_after_hours"] = ((df["hour"] < 7) | (df["hour"] >=
```

```
        19)).astype(int)
    df["date_only"] = dt.dt.floor("D")
    return df

#
    ============================================================

# 4) Load minimal users/psychometric (psychometric excluded
    by default)
#
    ============================================================


def load_users(path: str) -> pd.DataFrame:
    users = pd.read_csv(path)
    users = normalise_user_key(users, "user")
    return users

def load_psych(path: str) -> pd.DataFrame:
    psych = pd.read_csv(path)
    psych = normalise_user_key(psych, "user")
    return psych

users_df = load_users(cfg.users_path)
psych_df = load_psych(cfg.psych_path)
INCLUDE_PSYCH = False  # sensitive; default off

#
    ============================================================

# 5) CERT user-day feature engineering (shared with D1-D3
    style)
#
    ============================================================


def agg_logon(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df = normalise_user_key(df, cfg.user_col)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

```

```
    out = df.groupby([cfg.user_col, "date_only"]).agg(
        logon_events=("activity", "size") if "activity" in df.columns else (cfg.user_col, "size"),
        logon_after_hours=("is_after_hours", "sum"),
        logon_weekend=("is_weekend", "sum"),
        logon_unique_pc=("pc", pd.Series.nunique) if "pc" in df.columns else (cfg.user_col, "nunique")
    ).reset_index()
    out["logon_after_hours_ratio"] = out["logon_after_hours"] / out["logon_events"].clip(lower=1)
    out["logon_weekend_ratio"] = out["logon_weekend"] / out["logon_events"].clip(lower=1)
    return out

def agg_device(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df = normalise_user_key(df, cfg.user_col)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    out = df.groupby([cfg.user_col, "date_only"]).agg(
        device_events=("activity", "size") if "activity" in df.columns else (cfg.user_col, "size"),
        device_after_hours=("is_after_hours", "sum"),
        device_weekend=("is_weekend", "sum"),
        device_unique_pc=("pc", pd.Series.nunique) if "pc" in df.columns else (cfg.user_col, "nunique")
    ).reset_index()
    out["device_after_hours_ratio"] = out["device_after_hours"] / out["device_events"].clip(lower=1)
    out["device_weekend_ratio"] = out["device_weekend"] / out["device_events"].clip(lower=1)
    return out

def agg_http(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df = normalise_user_key(df, cfg.user_col)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    count_key = "url" if "url" in df.columns else cfg.
```

```
        user_col
    out = df.groupby([cfg.user_col, "date_only"]).agg(
        http_events=(count_key, "size"),
        http_after_hours=("is_after_hours", "sum"),
        http_weekend=("is_weekend", "sum")
    ).reset_index()
    out["http_after_hours_ratio"] = out["http_after_hours"] /
        out["http_events"].clip(lower=1)
    out["http_weekend_ratio"] = out["http_weekend"] / out["
        http_events"].clip(lower=1)
    return out

def agg_email(path: str) -> pd.DataFrame:
    df = pd.read_csv(path)
    df = normalise_user_key(df, cfg.user_col)
    df[cfg.time_col] = parse_datetime_safe(df[cfg.time_col])
    df = df.dropna(subset=[cfg.time_col, cfg.user_col])
    df = add_time_features(df, cfg.time_col)

    out = df.groupby([cfg.user_col, "date_only"]).agg(
        email_events=(cfg.user_col, "size"),
        email_after_hours=("is_after_hours", "sum"),
        email_weekend=("is_weekend", "sum")
    ).reset_index()
    out["email_after_hours_ratio"] = out["email_after_hours"]
        / out["email_events"].clip(lower=1)
    out["email_weekend_ratio"] = out["email_weekend"] / out["
        email_events"].clip(lower=1)
    return out

def agg_file_chunked(path: str, chunksize: int) -> Tuple[pd.
    DataFrame, pd.DataFrame]:
    feats_accum, labels_accum = [], []

    use_cols = [cfg.time_col, cfg.user_col, "pc", "filename",
        "activity",
                "to_removable_media", "from_removable_media"]

    for chunk in pd.read_csv(path, usecols=lambda c: c in
        use_cols, chunksize=chunksize):
        chunk = normalise_user_key(chunk, cfg.user_col)
        chunk[cfg.time_col] = parse_datetime_safe(chunk[cfg.
            time_col])
```

```
        chunk = chunk.dropna(subset=[cfg.time_col, cfg.
            user_col])
        chunk = add_time_features(chunk, cfg.time_col)

        for b in ["to_removable_media", "from_removable_media
            "]:
            if b in chunk.columns:
                if chunk[b].dtype == object:
                    chunk[b] = chunk[b].astype(str).str.lower
                        ().map({"true": 1, "false": 0})
                chunk[b] = chunk[b].fillna(0).astype(int)
            else:
                chunk[b] = 0

        grp = chunk.groupby([cfg.user_col, "date_only"]).agg(
            file_events=("activity", "size"),
            file_after_hours=("is_after_hours", "sum"),
            file_weekend=("is_weekend", "sum"),
            file_unique_pc=("pc", pd.Series.nunique),
            file_unique_filename=("filename", pd.Series.
                nunique),
            file_to_rm=("to_removable_media", "sum"),
            file_from_rm=("from_removable_media", "sum")
        ).reset_index()

        grp["file_after_hours_ratio"] = grp["file_after_hours
            "] / grp["file_events"].clip(lower=1)
        grp["file_weekend_ratio"] = grp["file_weekend"] / grp
            ["file_events"].clip(lower=1)
        feats_accum.append(grp)

        label_col = "file_to_rm" if cfg.label_mode == "
            to_removable_media" else "file_from_rm"
        lab = grp[[cfg.user_col, "date_only", label_col]].
            copy()
        lab["label"] = (lab[label_col] > 0).astype(int)
        lab = lab.drop(columns=[label_col])
        labels_accum.append(lab)

        del chunk, grp, lab
        gc.collect()

    file_feats = pd.concat(feats_accum, ignore_index=True)
```

```
287     labels = pd.concat(labels_accum, ignore_index=True)
288 
289     file_feats = file_feats.groupby([cfg.user_col, "date_only
            "], as_index=False).sum(numeric_only=True)
290     labels = labels.groupby([cfg.user_col, "date_only"],
            as_index=False).max(numeric_only=True)
291 
292     return file_feats, labels
293 
294 # Build domain features
295 logon_df = agg_logon(cfg.logon_path)
296 device_df = agg_device(cfg.device_path)
297 http_df = agg_http(cfg.http_path)
298 email_df = agg_email(cfg.email_path)
299 file_df, y_day = agg_file_chunked(cfg.file_path, cfg.
        file_chunksize)
300 
301 # Merge to user-day table
302 df_day = file_df.merge(logon_df, on=[cfg.user_col, "date_only
        "], how="left") \
303                 .merge(device_df, on=[cfg.user_col, "
                        date_only"], how="left") \
304                 .merge(http_df, on=[cfg.user_col, "date_only"
                        ], how="left") \
305                 .merge(email_df, on=[cfg.user_col, "date_only
                        "], how="left") \
306                 .merge(y_day, on=[cfg.user_col, "date_only"],
                         how="inner")
307 
308 df_day = df_day.fillna(0)
309 
310 if "user" in users_df.columns:
311     df_day = df_day.merge(users_df[["user"]].drop_duplicates
            (), on="user", how="left")
312 
313 if INCLUDE_PSYCH and ("user" in psych_df.columns):
314     psych_num = psych_df.select_dtypes(include=[np.number]).
            copy()
315     psych_min = pd.concat([psych_df[["user"]], psych_num],
            axis=1)
316     df_day = df_day.merge(psych_min, on="user", how="left").
            fillna(0)
317 
```

```
# Exclude direct label proxies to reduce leakage risk
drop_cols = [cfg.user_col, "date_only", "label", "file_to_rm", "file_from_rm"]
feature_cols = [c for c in df_day.columns if c not in drop_cols]

df_day = df_day.sort_values("date_only").reset_index(drop=True)
X_all_raw = df_day[feature_cols].astype(np.float32).values
y_all = df_day["label"].astype(int).values

print("User-day dataset:", df_day.shape)
print("Num features (post-leakage guard):", len(feature_cols))
print("Positive rate:", float(y_all.mean()))

# ============================================================

# 6) Base detector (ANN) for robustness evaluation
# ============================================================


def build_detector(input_dim: int) -> keras.Model:
    inp = keras.Input(shape=(input_dim,), name="features_in")
    x = inp
    for i, hu in enumerate(cfg.hidden_units):
        x = layers.Dense(hu, activation="relu", name=f"dense_{i+1}")(x)
        x = layers.Dropout(cfg.dropout, name=f"dropout_{i+1}")(x)
    out = layers.Dense(1, activation="sigmoid", name="out")(x)
    model = keras.Model(inp, out, name="GuardingTheGatesDetector")
    model.compile(optimizer=keras.optimizers.Adam(1e-3), loss="binary_crossentropy")
    return model

def class_weights(y: np.ndarray) -> Dict[int, float]:
    n_pos = max(1, int(y.sum()))
    n_neg = max(1, int((y == 0).sum()))
```

```
348     return {
349         0: (n_pos + n_neg) / (2.0 * n_neg),
350         1: (n_pos + n_neg) / (2.0 * n_pos),
351     }
352
353 def compute_metrics(y_true: np.ndarray, y_prob: np.ndarray,
        thr: float = 0.5) -> Dict[str, float]:
354     y_pred = (y_prob >= thr).astype(int)
355     p, r, f1, _ = precision_recall_fscore_support(y_true,
            y_pred, average="binary", zero_division=0)
356     pr_auc = average_precision_score(y_true, y_prob)
357
358     roc_auc = np.nan
359     if len(np.unique(y_true)) == 2:
360         roc_auc = roc_auc_score(y_true, y_prob)
361
362     tn, fp, fn, tp = confusion_matrix(y_true, y_pred).ravel()
363     return {
364         "precision": float(p),
365         "recall": float(r),
366         "f1": float(f1),
367         "pr_auc": float(pr_auc),
368         "roc_auc": float(roc_auc) if not np.isnan(roc_auc)
                else np.nan,
369         "tp": int(tp), "fp": int(fp), "tn": int(tn), "fn":
                int(fn),
370     }
371
372 #
        ============================================================

373 # 7) Adversarial perturbation (tabular, bounded, research-
        safe)
374 #
        ============================================================

375 # This section creates bounded perturbations in scaled
        feature space to
376 # stress-test the detector. It is designed for defensive
        evaluation.
377 #
378 # SAFETY DESIGN CHOICES:
379 # - operates on aggregated user-day features (not raw logs/
```

```
    content)
# - bounded epsilon and clipping
# - optionally targets only positive-class test samples for
    robustness study
# - no instructions for evasion deployment or operational
    misuse

@tf.function
def _gradient_sign_step(model, x, y_true):
    x_var = tf.cast(x, tf.float32)
    y_true = tf.cast(tf.reshape(y_true, (-1, 1)), tf.float32)

    with tf.GradientTape() as tape:
        tape.watch(x_var)
        y_pred = model(x_var, training=False)
        loss = tf.keras.losses.binary_crossentropy(y_true,
            y_pred)
        loss = tf.reduce_mean(loss)

    grads = tape.gradient(loss, x_var)
    signed = tf.sign(grads)
    return signed

def fgsm_attack(model, X, y, eps=0.05, clip_min=-5.0,
    clip_max=5.0):
    """
    Fast Gradient Sign Method (bounded, defensive evaluation
        context).
    Returns adversarially perturbed X in scaled feature space
        .
    """
    x_tf = tf.convert_to_tensor(X, dtype=tf.float32)
    y_tf = tf.convert_to_tensor(y, dtype=tf.float32)
    signed_grad = _gradient_sign_step(model, x_tf, y_tf)
    x_adv = x_tf + eps * signed_grad
    x_adv = tf.clip_by_value(x_adv, clip_min, clip_max)
    return x_adv.numpy()

def pgd_like_attack(model, X, y, eps=0.08, alpha=0.02, steps
    =5, clip_min=-5.0, clip_max=5.0):
    """
    Iterative bounded perturbation (PGD-like) for robustness
        evaluation.
```

```
414     This is constrained and intended for defensive testing
           only.
415     """
416     X_orig = X.astype(np.float32).copy()
417     X_adv = X_orig.copy()
418 
419     # Small random start within epsilon ball (optional
           robustness stress)
420     X_adv = X_adv + np.random.uniform(-eps, eps, size=X_adv.
           shape).astype(np.float32)
421     X_adv = np.clip(X_adv, clip_min, clip_max)
422 
423     for _ in range(steps):
424         x_tf = tf.convert_to_tensor(X_adv, dtype=tf.float32)
425         y_tf = tf.convert_to_tensor(y, dtype=tf.float32)
426         signed_grad = _gradient_sign_step(model, x_tf, y_tf).
               numpy()
427 
428         X_adv = X_adv + alpha * signed_grad
429 
430         # Project back to epsilon-ball around original input
431         delta = np.clip(X_adv - X_orig, -eps, eps)
432         X_adv = X_orig + delta
433 
434         # Global clipping in scaled feature space
435         X_adv = np.clip(X_adv, clip_min, clip_max)
436 
437     return X_adv.astype(np.float32)
438 
439 #
       ============================================================
440 # 8) Optional adversarial training (defensive fine-tuning)
441 #
       ============================================================
442 
443 def adversarial_train_epoch(model, X_train, y_train,
       batch_size, eps, frac_adv):
444     """
445     One simple adversarial training epoch (FGSM-style mixed
           batches).
446     Defensive purpose only. Keeps implementation compact for
```

```
        appendix use.
    """
    n = len(X_train)
    idx = np.random.permutation(n)
    X_train = X_train[idx]
    y_train = y_train[idx]

    num_batches = int(np.ceil(n / batch_size))
    losses = []

    for b in range(num_batches):
        s = b * batch_size
        e = min((b + 1) * batch_size, n)
        xb = X_train[s:e].copy()
        yb = y_train[s:e].copy()

        if len(xb) == 0:
            continue

        k = int(len(xb) * frac_adv)
        if k > 0:
            adv_idx = np.random.choice(len(xb), size=k,
                replace=False)
            xb_adv = fgsm_attack(
                model, xb[adv_idx], yb[adv_idx],
                eps=eps, clip_min=cfg.adv_clip_min, clip_max=
                    cfg.adv_clip_max
            )
            xb[adv_idx] = xb_adv

        loss = model.train_on_batch(xb, yb)
        losses.append(float(loss))

    return float(np.mean(losses)) if losses else np.nan

#
    ============================================================

# 9) Blocked temporal robustness evaluation
#
    ============================================================


```

```
tscv = TimeSeriesSplit(n_splits=cfg.n_splits)
results = []

for fold, (train_idx, test_idx) in enumerate(tscv.split(X_all_raw), start=1):
    print(f"\n========== Fold {fold}/{cfg.n_splits} ==========")

    # Split
    X_train_raw, X_test_raw = X_all_raw[train_idx], X_all_raw[test_idx]
    y_train, y_test = y_all[train_idx], y_all[test_idx]

    # Scale on train only
    scaler = StandardScaler()
    X_train = scaler.fit_transform(X_train_raw).astype(np.float32)
    X_test = scaler.transform(X_test_raw).astype(np.float32)

    # Temporal validation split (tail of training)
    n_train = len(X_train)
    split = int(0.8 * n_train)
    X_tr, X_val = X_train[:split], X_train[split:]
    y_tr, y_val = y_train[:split], y_train[split:]

    # Build/train detector
    model = build_detector(X_train.shape[1])
    callbacks = [
        keras.callbacks.EarlyStopping(monitor="val_loss", patience=cfg.patience, restore_best_weights=True)
    ]
    cw = class_weights(y_tr) if cfg.use_class_weights else None

    model.fit(
        X_tr, y_tr,
        validation_data=(X_val, y_val),
        epochs=cfg.epochs,
        batch_size=cfg.batch_size,
        verbose=0,
        class_weight=cw,
        callbacks=callbacks
    )
```

```
520
521     # Optional adversarial training fine-tune (defensive)
522     if cfg.use_adversarial_training:
523         for _ in range(2):  # short fine-tune to keep runtime
                reasonable
524             adv_loss = adversarial_train_epoch(
525                 model, X_tr, y_tr,
526                 batch_size=cfg.batch_size,
527                 eps=cfg.adv_eps,
528                 frac_adv=cfg.adv_train_fraction
529             )
530         print(f"Adversarial fine-tuning completed (loss ~ {
               adv_loss:.4f})")
531
532     # -------------------------
533     # Clean evaluation
534     # -------------------------
535     y_prob_clean = model.predict(X_test, batch_size=cfg.
           batch_size, verbose=0).ravel()
536     m_clean = compute_metrics(y_test, y_prob_clean, thr=0.5)
537
538     # -------------------------
539     # Adversarial evaluation
540     # -------------------------
541     if cfg.attack_positive_only:
542         mask = (y_test == 1)
543         if mask.sum() == 0:
544             # Fallback if a fold has no positives
545             X_adv_target = X_test.copy()
546             y_adv_target = y_test.copy()
547             attack_scope = "all_samples_fallback"
548         else:
549             X_adv_target = X_test[mask].copy()
550             y_adv_target = y_test[mask].copy()
551             attack_scope = "positive_only"
552     else:
553         X_adv_target = X_test.copy()
554         y_adv_target = y_test.copy()
555         attack_scope = "all_samples"
556
557     # Create bounded adversarial samples for target subset
558     X_adv_target_pgd = pgd_like_attack(
559         model, X_adv_target, y_adv_target,
```

```
        eps=cfg.adv_eps, alpha=cfg.adv_alpha, steps=cfg.
            adv_steps,
        clip_min=cfg.adv_clip_min, clip_max=cfg.adv_clip_max
    )

    # Reconstruct full adversarial test set (only perturb
        selected subset)
    X_test_adv = X_test.copy()
    if cfg.attack_positive_only and attack_scope == "
        positive_only":
        X_test_adv[mask] = X_adv_target_pgd
    else:
        X_test_adv = X_adv_target_pgd

    y_prob_adv = model.predict(X_test_adv, batch_size=cfg.
        batch_size, verbose=0).ravel()
    m_adv = compute_metrics(y_test, y_prob_adv, thr=0.5)

    # Degradation summaries
    fold_res = {
        "fold": fold,
        "n_test": int(len(test_idx)),
        "pos_rate_test": float(y_test.mean()),
        "attack_scope": attack_scope,
        "eps": float(cfg.adv_eps),
        "alpha": float(cfg.adv_alpha),
        "steps": int(cfg.adv_steps),

        # Clean metrics
        "clean_precision": m_clean["precision"],
        "clean_recall": m_clean["recall"],
        "clean_f1": m_clean["f1"],
        "clean_pr_auc": m_clean["pr_auc"],
        "clean_roc_auc": m_clean["roc_auc"],

        # Adversarial metrics
        "adv_precision": m_adv["precision"],
        "adv_recall": m_adv["recall"],
        "adv_f1": m_adv["f1"],
        "adv_pr_auc": m_adv["pr_auc"],
        "adv_roc_auc": m_adv["roc_auc"],

        # Degradation (positive means performance dropped)
```

```
        "delta_precision": m_clean["precision"] - m_adv["
            precision"],
        "delta_recall": m_clean["recall"] - m_adv["recall"],
        "delta_f1": m_clean["f1"] - m_adv["f1"],
        "delta_pr_auc": m_clean["pr_auc"] - m_adv["pr_auc"],
        "delta_roc_auc": (
            (m_clean["roc_auc"] - m_adv["roc_auc"])
            if (not np.isnan(m_clean["roc_auc"]) and not np.
                isnan(m_adv["roc_auc"]))
            else np.nan
        ),

        # Confusions (clean)
        "clean_tp": m_clean["tp"], "clean_fp": m_clean["fp"],
        "clean_fn": m_clean["fn"], "clean_tn": m_clean["tn"],

        # Confusions (adv)
        "adv_tp": m_adv["tp"], "adv_fp": m_adv["fp"],
        "adv_fn": m_adv["fn"], "adv_tn": m_adv["tn"],
    }

    print(
        f"Clean  | P={m_clean['precision']:.3f} R={m_clean['
            recall']:.3f} "
        f"F1={m_clean['f1']:.3f} PR-AUC={m_clean['pr_auc']:.3
            f}"
    )
    print(
        f"Advers | P={m_adv['precision']:.3f} R={m_adv['
            recall']:.3f} "
        f"F1={m_adv['f1']:.3f} PR-AUC={m_adv['pr_auc']:.3f}"
    )
    print(
        f"Delta  | dF1={fold_res['delta_f1']:.3f} dPR-AUC={
            fold_res['delta_pr_auc']:.3f} "
        f"(eps={cfg.adv_eps}, steps={cfg.adv_steps}, scope={
            attack_scope})"
    )

    results.append(fold_res)

    # Cleanup
    tf.keras.backend.clear_session()
```

```
635     del model, scaler, X_train, X_test, X_tr, X_val,
            X_test_adv, y_prob_clean, y_prob_adv
636     gc.collect()
637
638 #
        ============================================================

639 # 10) Aggregate robustness summaries
640 #
        ============================================================

641
642 res_df = pd.DataFrame(results)
643 print("\nPer-fold robustness results:")
644 print(res_df)
645
646 summary_cols = [
647     "clean_precision", "clean_recall", "clean_f1", "
            clean_pr_auc", "clean_roc_auc",
648     "adv_precision", "adv_recall", "adv_f1", "adv_pr_auc", "
            adv_roc_auc",
649     "delta_precision", "delta_recall", "delta_f1", "
            delta_pr_auc", "delta_roc_auc"
650 ]
651 summary = res_df[summary_cols].agg(["mean", "std"]).T.
        reset_index()
652 summary.columns = ["metric", "mean", "std"]
653
654 print("\nRobustness summary (Mean    Std):")
655 print(summary)
656
657 #
        ============================================================

658 # 11) Export artefacts (aggregated metrics only)
659 #
        ============================================================

660 # Ethical note:
661 # Export aggregated robustness results only; avoid releasing
        sensitive
662 # intermediate records or perturbation details tied to real
        users.
```

```
663
664 res_path = "/kaggle/working/
       guarding_the_gates_adversarial_folds.csv"
665 sum_path = "/kaggle/working/
       guarding_the_gates_adversarial_summary.csv"
666
667 res_df.to_csv(res_path, index=False)
668 summary.to_csv(sum_path, index=False)
669
670 print("\nSaved artefacts:")
671 print(" -", res_path)
672 print(" -", sum_path)
673
674 #
       ============================================================

675 # 12) Appendix-friendly ethics-oriented model card
676 #
       ============================================================

677
678 model_card = f"""
679 Guarding the Gates (Adversarial ML) Model Card - CERT
       Research Artefact
680 - Intended use: Defensive robustness evaluation of insider-
       threat detectors under bounded perturbations.
681 - Data: Aggregated user-day behavioural features from CERT
       logs (privacy-minimised).
682 - Label: Proxy from file.csv ({cfg.label_mode}) at user-day
       level.
683 - Detector: Feed-forward ANN (tabular); robustness tested
       with bounded gradient-based perturbations.
684 - Evaluation: Blocked temporal folds to preserve chronology
       and reduce leakage.
685 - Primary reporting: Clean vs adversarial PR-AUC, Recall, F1,
        and degradation (delta metrics).
686 - Safety constraints: Bounded epsilon, clipped perturbations,
        aggregated features only, defensive framing.
687 - Key risks: Misuse for evasion research, proxy-label bias,
       false positives, privacy harms.
688 - Mitigations: Human-in-the-loop review, governance oversight
       , data minimisation, audit logging, threshold calibration.
689 - Prohibited use: Operational evasion, concealment of
```

```
   malicious activity, or covert surveillance without due
   process.
"""
print(model_card)
```

Listing D.4: Guarding the Gates: adversarial robustness evaluation (tabular behavioural features) for CERT-style insider-threat detection using blocked temporal folds.

D.5 Appendix D Artefact Summary

For convenience, Table D.1 provides a consolidated overview of the Appendix D artefacts (D1–D4), including their purpose, input logs, outputs, evaluation protocol, and ethical-use constraints.

Table D.1: Appendix D overview of implementation artefacts (D1–D4) and their methodological and ethical scope.

ID	Artefact Name	Purpose	Input Logs (CERT)	Output Files	Evaluation Protocol	Ethical Notes
D1	SOMANN Implementation	Reproducible SOM–ANN pipeline for insider-threat detection using SOM-based representation learning followed by ANN classification on aggregated behavioural features.	`file.csv`, `logon.csv`, `device.csv`, `http.csv`, `email.csv`; optional `users.csv`; psychometric excluded by default.	`somann\_cert\_temporal\_folds.csv`; `somann\_cert\_summary.csv`	Blocked temporal folds; train-only preprocessing/scaling; PR–AUC, Precision, Recall, F_1 (ROC–AUC secondary).	Research/educational use only; privacy-minimised user-day aggregation; no content parsing by default; lawful basis, proportionality, governance, and human oversight required for adaptation.
D2	MANN Implementation	Reproducible modular neural architecture with domain-specific modules and structured fusion for insider-risk estimation.	`file.csv`, `logon.csv`, `device.csv`, `http.csv`, `email.csv`; optional `users.csv`; psychometric excluded by default.	`mann\_cert\_temporal\_folds.csv`; `mann\_cert\_summary.csv`	Blocked temporal folds; per-domain train-only scaling; leakage-aware feature construction; PR–AUC-focused reporting.	Supports interpretability through domain decomposition, but outputs must remain decision-support inputs rather than sole grounds for punitive action.
D3	MLML Implementation	Reproducible layered ML framework with domain-specific base learners, meta-feature fusion, and final meta-classifier.	`file.csv`, `logon.csv`, `device.csv`, `http.csv`, `email.csv`; optional `users.csv`; psychometric excluded by default.	`mlml\_cert\_temporal\_folds.csv`; `mlml\_cert\_summary.csv`	Blocked temporal folds; train-only preprocessing; PR–AUC prioritised under imbalance; fold-wise mean ± std summaries.	Proxy-label bias, false positives, and role-confounding risks remain; fairness checks and impact analysis are required before deployment.
D4	*Guarding the Gates* (Adversarial ML)	Defensive robustness evaluation using bounded perturbations on tabular behavioural features, with optional adversarial fine-tuning.	User-day features derived from `file.csv`, `logon.csv`, `device.csv`, `http.csv`, `email.csv`; optional `users.csv`; psychometric excluded by default.	`guarding\_the\_gates\_adversarial\_folds.csv`; `guarding\_the\_gates\_adversarial\_summary.csv`	Blocked temporal folds; clean vs adversarial evaluation; bounded perturbations in scaled feature space; degradation (Δ) reporting.	Defensive research use only; not for evasion; bounded perturbations and aggregated features reduce misuse risk, but formal ethical review and governance remain essential.

Appendix E

List of Abbreviations

Abbreviation	Description
AI	Artificial Intelligence
AML	Adversarial Machine Learning
ANN	Artificial Neural Network
AP	Average Precision
AUC	Area Under the Curve
BMU	Best Matching Unit
CERT	CERT Insider Threat Dataset (synthetic multi-log benchmark)
CNN	Convolutional Neural Network
DBIR	Data Breach Investigations Report
DSR	Design Science Research
FOIA	Freedom of Information Act
FPR	False Positive Rate
GRU	Gated Recurrent Unit
GNN	Graph Neural Network
HMM	Hidden Markov Model
HTTP	Hypertext Transfer Protocol
IDS	Intrusion Detection System
IoT	Internet of Things
IP	Intellectual Property
KBS	Knowledge-Based Systems
LANL	Los Alamos National Laboratory
LSTM	Long Short-Term Memory
MANN	Modular Artificial Neural Network
ML	Machine Learning
MLP	Multilayer Perceptron
NCSC	National Cyber Security Centre (UK)
NIST	National Institute of Standards and Technology
PR-AUC	Area Under the Precision–Recall Curve
R-Guard	Robust Guard (behaviourally robust detection architecture)
RIT-IDEAL	RIT-IDEAL Insider Threat Dataset (Rochester Institute of Technology)
RL	Reinforcement Learning
ROC-AUC	Area Under the Receiver Operating Characteristic Curve

Abbreviation	Description
RQ	Research Question
RNN	Recurrent Neural Network
SIEM	Security Information and Event Management
SOM	Self-Organising Map
SOMANN	Self-Organising Map and Artificial Neural Network Hybrid
TCN	Temporal Convolutional Network
TNR	True Negative Rate
TPR	True Positive Rate
XAI	Explainable Artificial Intelligence
ZTA	Zero Trust Architecture

Appendix F

Terms and Terminology

This appendix defines key terms and concepts used throughout this thesis. The aim is to ensure terminological clarity and consistency across behavioural cybersecurity, machine learning, and adversarial robustness domains. Definitions are provided in the context in which the terms are used in this work.

Artificial Intelligence (AI) A broad field of computer science concerned with the development of systems capable of performing tasks that typically require human intelligence, such as reasoning, learning, decision-making, and pattern recognition.

Machine Learning (ML) A subfield of artificial intelligence focused on algorithms that learn patterns and relationships from data without explicit programming, enabling adaptive and data-driven decision-making.

Artificial Neural Network (ANN) A class of machine-learning models inspired by biological neural systems, consisting of interconnected layers of artificial neurons that learn nonlinear mappings between inputs and outputs through supervised or unsupervised training.

Insider Threat A security risk originating from individuals with legitimate access to organisational systems, data, or facilities, whose actions—whether malicious, negligent, or coerced—may compromise confidentiality, integrity, or availability.

Malicious Insider An individual who intentionally exploits authorised access to cause harm, such as data exfiltration, sabotage, fraud, or espionage.

Negligent Insider An individual whose actions unintentionally compromise security due to errors, policy violations, or lack of awareness, without deliberate malicious intent.

Compromised Insider An individual whose credentials or access have been hijacked or manipulated by an external adversary, causing insider-like behaviour without the individual's intent.

Behavioural Telemetry Digitally recorded traces of user activity, including authentication events, file access, device usage, communication metadata, and network interactions.

Behavioural Signal A derived or aggregated indicator extracted from raw telemetry that reflects user behaviour over time, such as access frequency, after-hours activity, or privilege use patterns.

Behavioural Drift Gradual, often subtle changes in user behaviour over time that may indicate evolving risk, adaptation, or escalation, rather than abrupt anomalies.

Behavioural Trajectory A temporally ordered sequence of behavioural states or signals representing how user activity evolves across multiple time horizons.

Perception–Inference Separation An architectural principle in which representation learning (perception) is decoupled from supervised decision-making (inference), enabling stable behavioural abstraction prior to classification.

Self-Organising Map (SOM) An unsupervised neural network that projects high-dimensional input data onto a low-dimensional (typically two-dimensional) topological grid, preserving neighbourhood relationships and supporting interpretable clustering.

Best Matching Unit (BMU) The SOM neuron whose prototype vector is closest to a given input sample according to a defined distance metric, typically Euclidean distance.

U-Matrix (Unified Distance Matrix) A visualisation of distances between neighbouring SOM neurons, commonly used to identify cluster boundaries and regions of high behavioural dissimilarity.

Mean Inter-Node Distance (MID) A quantitative SOM diagnostic measuring average distances between neighbouring neurons, used to identify regions of behavioural dispersion or anomaly concentration.

SOM–ANN (SOMANN) A hybrid architecture combining unsupervised SOM-based representation learning with supervised ANN inference, used in this thesis to stabilise behavioural embeddings prior to classification.

Representation Stabilisation The reduction of variance and noise in behavioural features through unsupervised neighbourhood-based aggregation, improving robustness and consistency for downstream learning.

Identity Confounding A modelling failure mode in which a classifier inadvertently learns user identity or role-specific artefacts rather than behaviourally meaningful patterns.

Unified Embedding A single latent representation produced by combining all behavioural features into one vector space, without explicit domain separation.

Semantic Dilution Loss of domain-specific behavioural meaning caused by combining heterogeneous features into a single embedding without structural alignment.

Modular Artificial Neural Network (MANN) A supervised learning architecture composed of multiple domain-specific subnetworks whose outputs are fused to produce a unified risk assessment.

Behavioural Domain A semantically coherent subset of behavioural telemetry, such as authentication, file access, communication, or device usage.

Domain Encoder A neural subnetwork trained to learn representations specific to a single behavioural domain.

Controlled Fusion A mechanism for integrating domain-specific embeddings using structured operators such as concatenation, weighted fusion, or gated/attention-based fusion.

Cross-Domain Masking A phenomenon in which high-volume or noisy domains dominate fused representations, obscuring weak but meaningful signals from other domains.

Hierarchical Behavioural Modelling An architectural approach that organises behavioural signals across multiple temporal or semantic layers, enabling both short-term detection and long-term progression analysis.

Temporal Horizon A defined time window over which behavioural features are aggregated or analysed, such as short-term (minutes/hours) or long-term (days/weeks).

Over-Smoothing Excessive temporal aggregation that delays detection by suppressing rapid behavioural changes.

Ablation Study An experimental technique in which components of a model are systematically removed or altered to assess their contribution to overall performance.

Sensitivity Analysis An evaluation of how model performance varies in response to changes in hyperparameters, architectural choices, or data characteristics.

Chronology-Preserving Split A dataset partitioning strategy that maintains temporal order between training, validation, and test sets to prevent future information leakage.

Data Leakage The unintended use of information from outside the training partition during model training or preprocessing, leading to inflated performance estimates.

Precision–Recall (PR) Curve A performance metric particularly suitable for imbalanced datasets, capturing the trade-off between precision (false positives) and recall (false negatives).

Adversarial Machine Learning (AML) The study of attacks and defences involving intentionally crafted inputs designed to deceive or degrade machine-learning systems.

Behaviourally Plausible Adversary An adversary that modifies behaviour in ways that remain consistent with legitimate roles, workflows, and organisational norms.

Insider Attack Adaptation (IAA) A sequential adversarial strategy in which an insider gradually adapts behaviour to evade detection while maintaining operational plausibility.

R-Guard A robustness framework introduced in this thesis that integrates detection, monitoring, and response mechanisms to maintain model stability under behavioural manipulation.

Robustness–Utility Trade-off The balance between maintaining detection performance and avoiding excessive false positives when defending against adaptive adversaries.

False Positive A benign behaviour incorrectly classified as malicious, often leading to alert fatigue and operational cost.

False Negative A malicious behaviour incorrectly classified as benign, representing missed detection.

Evaluation Discipline A methodological approach emphasising temporal faithfulness, appropriate metrics, and realistic threat assumptions in experimental design.

Operational Plausibility The requirement that detected behaviours align with realistic organisational workflows and user roles, supporting practical deployment.