FEPDS: A Proposal for the Extraction of Fuzzy Emerging Patterns in Data Streams

Ángel M. García-Vico, Cristóbal J. Carmona, Pedro González, Huseyin Seker and María J. del Jesus

***Abstract*—Nowadays, most data is generated by devices that produce data continuously. These kinds of data can be categorised as data streams and valuable insights can be extracted from them. In particular, the insights extracted by emerging patterns are interesting in a data stream context as easy, fast, reliable decisions can be made. However, their extraction is a challenge due to the necessary response time, memory and continuous model updates.**

In this paper, an approach for the extraction of emerging patterns in data streams is presented. It processes the instances by means of batches following an adaptive approach. The learning algorithm is an evolutionary fuzzy system where previous knowledge is employed in order to adapt to concept drift. A wide experimental study has been performed in order to show both the suitability of the approach in combating concept drift and the quality of the knowledge extracted. Finally, the proposal is applied to a case study related to the continuous determination of the profiles of New York City cab customers according to their fare amount, in order to show its potential.

***Index Terms*—Emerging pattern mining, Data stream mining, Evolutionary fuzzy systems, Multi-objective evolutionary algorithms.**

I. Introduction

THE data continuously sent by the devices surrounding us can be categorised as a data stream [1]. An analysis of such data could provide valuable insights into our activities and environments. Representative examples of these kinds of data are sensor networks, energy management and telecommunications [2]. In this kind of application one of the main characteristics is that data is less relevant as it becomes older. Therefore, a continuous analysis of data is better than a traditional static one. However, the characteristics of data streams pose several challenges for the development of methods. Among others, the most relevant constraints in data stream mining are the continuous updating and adaptation of the learning model as data arrive as underlying distributions can change over time. This is known in the literature as concept drift [1]. Moreover, the speed at which data arrive could be very high so the update of the model must be as fast as possible.

Emerging Pattern Mining (EPM) [3], [4] is a data mining technique framed within supervised descriptive rule discovery (SDRD) [5]. This task is half-way between descriptive and predictive inductions and attempts to describe relationships in data with respect to a given property of interest. In particular, EPM tries to describe discriminative relationships between different values of a property of interest or the description of emerging behaviour in data. EPM has been successfully applied in several fields such as management [6], disease detection [7], bioinformatics [8] and others [9]. Recently, several approaches based on evolutionary fuzzy systems (EFSs) have been proposed [10], [11] whose trade-off between the generality and reliability of the patterns extracted is improved with respect to other approaches developed for EPM [4]. In this way, a better description of the underlying phenomena in data is provided. This is quite relevant for data stream mining, as an easy, accurate description of what is going on within data is necessary in order to make decisions as fast as possible. However, to the best of our knowledge the extraction of emerging patterns (EPs) in data streams environments has not been tackled so far.

This paper presents an approach for the extraction of Fuzzy EPs in Data Streams (FEPDS), where its main components and working scheme are shown. The quality of the patterns extracted by FEPDS is analysed in a wide experimental study with a set of 94 synthetic datasets together with an analysis of the adaptation to the concept drift. In addition, the usefulness of FEPDS is demonstrated against a real dataset. In particular, the aim for that problem is the continuous description of the characteristics of New York City yellow cab customers with respect to the amount of money spent on their journeys, in order to improve the service.

This paper is organised as follows: Firstly, the main concepts related to concept drift, data streams and EPM are briefly described in Section II. Next, Section III presents the FEPDS algorithm, its components and its characteristics. Section IV presents the experimental framework and the analysis of the results. Section V presents the case study and its results. Finally, the conclusions are shown in Section VI.

II. Related work

In this section, the main related concepts in this paper are described such as concept drift (Section II-A), data streams with concept drift (Section II-B), and finally, fuzzy EPM (Section II-C).

A.M. García-Vico, C.J. Carmona, P. González and M.J. del Jesus are with the Andalusian Research Institute on Data Science and Computational Intelligence, University of Jaén, 23071 Jaén (Spain) (e-mail: {agvico|ccarmona|pglez|mjjesus}@ujaen.es)

Huseyin Seker is with the School of Computing and Digital Technologies, Staffordshire University, ST4 2DE, Stoke-On-Trent (United Kingdom) (e-mail: huseyin.seker@staffs.ac.uk)

This work was supported by the Spanish Ministry of Economy and Competitiveness under the project TIN2015-68454-R (FEDER Founds) and FPI 2016 Scholarship reference BES-2016-077738 (FEDER Founds). This paper was written in the Department of Computer and Information Sciences, Faculty of Engineering and Environment of The Northumbria University at Newcastle, Newcastle-Upon-Tyne, during the research visit of A.M. García-Vico. Thanks to Prof. H. Seker for the invitation.

A. Concept drift

Concept drift occurs when a non-stationary target concept is modified in the environment. Formally, a real drift is defined by a change in time t with respect to a time $t+\delta$ on the conditional probability of the classes given an instance x, i.e., $P\left(C|x\right)^{t} \neq P\left(C|x\right)^{t+\delta}$ [12]. Specifically, two types of drift are defined [13] considering both cause and effect of the probability distribution associated to the data (instances and classes):

- *Real drift*. It is considered when the probability distribution is modified with respect to the class but the incoming data probability could not be modified. The change could be visible or not from the data distribution.
- *Virtual drift*. It can be considered as incomplete data representation or changes in data distribution occurring without affecting the concept, amongst other factors.

The technology and their properties associated to the generation of stream data cause changes in data distribution over time. These changes can be reviewed in different ways. They can be classified as: abrupt, where a sudden drift in data occur instantly; incremental, where data changes gradually over time; gradual, which is a type of incremental drift with changes in class distribution too; and recurrent where concepts occur in a specific order again [14].

In a general way, it is important to remark that data stream mining should be able to work from evolving streams, regardless of classification or concept drift type. Therefore, concept drift must be analysed and taken into account for the design and development of data streaming algorithms.

B. Data streams with concept drift

A data stream is an unbounded, ordered sequence of instances that arrive at the system throughout time at a variable speed [15]. This simple definition of a data stream together with drifting concepts produces a huge amount of differences with respect to classical static datasets in data mining that must be taken into account. For example, their structures should be adapted with respect to the new incoming data in order to provide the best performance and a quick response [13]. Similarly, constraints arise and must be taken into consideration [16], [1], [17]. For instance, there are space constraints because it is impossible to store the whole data stream in memory [18]. Finally, concept drift produces a significant decrease in the performance of the learning algorithms [19]. Therefore, mechanisms for adapting the learner to these changes must be developed.

These aspects lead to online adaptive learning whereby data arrive in the system and a prediction value is obtained with the current model. In a later stage, the estimation of this prediction is analysed and the model can be modified or confirmed. In this way, data stream mining algorithms should determine how the instances will be processed, how they will learn from these instances in order to handle the concept drift and how the quality of the model will be determined. Thus, four important elements are defined in [12], [20]:

1) *Memory*. This is the element in charge of processing the new information and forgetting the old one. For processing new data these elements can be found: *sequential* or *online*, where the instances arrive one by one and are processed by the learning algorithm as soon as they are available (this is known as Online active learning [21]); and *Windowing* where informative data or summaries concerning model behaviour or data distribution are stored in a short memory. Usually, it is considered that the most recent instances are the most representative. These windows can be specific to the applied learner or for general purposes, i.e. for any learning algorithm. In addition, its size can be fixed or variable, and its position with respect to the stream can be based on the sliding window model [22] or on the landmark model [23].
2) *Learning process*. The learning process employed can be based on online learners, or based on a single adaptive learner. The latter usually incorporates forgetting mechanisms in order to discard old data. The idea is to mimic the way online learners work. These kinds of methods are interesting for controlling the complexity of the system in order to provide a fast response. In addition, ensembles can be employed which are able to handle several learners containing different concepts. Nevertheless, its complexity is higher.
3) *Change monitoring*. All the techniques and mechanisms which are able to detect concept drift or change within the distribution are considered in this component. When feedback is immediately available, these methods can be based on monitoring supervised indicators such as accuracy, recall or sensitivity and specificity [24]. Change monitoring elements are interesting for the detection of real and/or virtual drift. When feedback is delayed, it is interesting to use methods based on unsupervised indicators such as similarity in time [25] and space [26] of data, or based on measuring the model complexity.
4) *Learning adaptation*. This component is very relevant in evolving data environments. It contains the modules and operators for analysing and updating the structures in order to maintain the quality of the knowledge extracted. Two main types of adaptations in the learner can be found: *Adaptive or incremental*, where the model is adapted using forgetting mechanisms and discarding outdated concepts; and *Evolving*, where both concept drift adaptation mechanisms and model structure are modified by means of adding or pruning components according to the learning needs, usually processing the instances once [27], [28]. Moreover, it is important to remark that learning models can use different strategies for the adaptation of their structures, such as *Blind*, where the learner is triggered at regular intervals, so no change monitoring is needed; or reactive strategies, where the learner is triggered when a change monitoring method throws an alarm signal, known as *Informed*.

Finally, the development of an evaluation strategy in order to determine the quality of the model is necessary. In traditional data mining, this can be determined by several mechanisms such as cross-validation and hold-out, amongst others [29]. However, these mechanisms require a finite dataset in order to split the data into the training and test partitions. Data streams are unbounded so the use of these kinds of techniques is impossible. One of the most popular estimation techniques for the evaluation of data stream models is the well-known interleaved test-then-train method [30]. With this method, the model performs a prediction or an evaluation of the model using the new incoming instance or chunk. After that, once the real value of the class for the instance or chunk is known, the model is updated with it, i.e. it now belongs to the training data. By means of this estimation technique we are able to continuously evaluate the model. Recent applications of this technique can be observed in [31], [32], [33]. In addition, other factors such as performance or memory consumption over time should be taken into consideration.

C. Fuzzy emerging pattern mining

The EPM was defined by Dong and Li [3], [4] as the search for patterns whose support increases significantly from one dataset (D_1) to another (D_2). In particular, D_1 can be the dataset formed by examples that belongs to one class, and D_2 contains the examples that belong to the remaining classes. A pattern is considered as emerging if and only if its growth rate (GR) is greater than a given threshold ρ. The GR measure is defined as in Eq. 1

$$GR(x) = \begin{cases} 0, & IF\ Sup_{D_1}(x) = Sup_{D_2}(x) = 0, \\ \infty, & IF\ Sup_{D_1}(x) \neq 0 \wedge Sup_{D_2}(x) = 0, \\ \frac{Sup_{D_1}(x)}{Sup_{D_2}(x)}, & another\ case \end{cases} \tag{1}$$

where $Sup_{D_i}(x)$ is the support of the pattern x on dataset i (D_i). EPs are represented by means of conjunctions of attribute-value pairs, or attribute-value pairs in disjunctive normal form, which represents the discriminative characteristics they want to describe. For the determination of D_1 and D_2, these patterns are usually labelled with the class or the dataset they try to describe.

The main purpose of EPM is the extraction of the differentiating characteristics among classes or the description of emerging trends with respect to a property of interest. In recent years the use of fuzzy logic within SDRD is gaining special relevance with the development of different algorithms such as SDIGA [34], FEPM [35], NMEEFSD [36] and MOEA-EFEP [11], amongst others. The use of fuzzy logic allows us to handle numeric variables by means of fuzzy linguistic labels (LLs), avoiding the loss of information produced by a discretised representation and a knowledge description closer to human reasoning [37]. In addition, the majority of proposals are combined with evolutionary algorithms (EAs) [38], well known throughout the literature as EFSs [39]. These kinds of systems have been applied in several real-world applications [40], [41], [42], [43], [44]. For EPM, a detailed review where the most important EPM algorithms are classified according to the approach used to mine EPs is presented in [4].

In EPM, the determination of the descriptive characteristics of a pattern is based on the number of examples covered or not covered by the patterns which belong or do not belong to the class of the pattern. Thus, several quality measures can be used from the SDRD framework for the determination of a wide range of aspects. The most widely used quality measures in EPM are outlined in Table I [4] where p are examples correctly covered, n are incorrectly covered examples, P are examples of the class and N are examples that do not belong to the class.

TABLE I
QUALITY MEASURES USED IN EPM FOR THE DETERMINATION OF THE QUALITY OF A PATTERN

Name	Abbreviation	Formula
Confidence [45]	Conf	$\frac{p}{p+n}$
Unusualness [46]	WRAcc	$\frac{p+n}{P+N}\left(\frac{p}{p+n} - \frac{P}{P+N}\right)$
Growth Rate [3]	GR	$\frac{p \cdot N}{P \cdot n}$
Support Difference [46]	SuppDiff	$\frac{p}{P} - \frac{n}{N}$
True Positive Rate [47]	TPR	$\frac{p}{P}$
False Positive Rate [48]	FPR	$\frac{n}{N}$

III. FEPDS: Fuzzy Emerging Patterns Extraction from Data Stream

In this section, the algorithm for the extraction of fuzzy EPs (fEPs) in data stream mining, called FEPDS, and its main characteristics are presented.

Briefly, the main ideas of the algorithm are: it is considered for the learning algorithm that instances are collected from the stream by means of batches of a predetermined size. The algorithm employs as learning method a multi-objective EFS for the extraction of fEPs. This learning method allows the extraction of high-quality fEPs with a good trade-off between generality and reliability. In addition, the use of fuzzy logic within the learning process produces robustness against slight changes, so a good adaptation to gradual drift can be achieved. FEPDS follows a blind strategy, where the model is updated using adaptive learning. Following this strategy, the learning method is triggered for each batch of data. This learning method also employs a specific memory structure in order to use previously extracted knowledge to bias the current learning process based on the history. This memory is based on a sliding window which allows the storage of the previous models extracted. In this way, a good adaptation to gradual drift can be achieved as the method is regularly executed. Finally, the evaluation of the model follows a test-then-train approach which allows the continuous evaluation of the current model.

Below, the main elements of FEPDS are depicted. Firstly, the memory component of FEPDS is presented in Section III-A. Next, in Section III-B the main concepts of the learning adaptation process for the algorithm are described. After that, the evolutionary learning approach of the FEPDS algorithm for extracting fEPs and its main characteristics are shown in Section III-C. Finally, the connections between the different components are presented in the operational scheme of FEPDS in Section III-D.

A. Memory component

In FEPDS it is assumed that data arrive in the system in the form of batches of data of a fixed length. For this purpose, a process is carried out in the algorithm for collecting these batches of data, as can be observed in Fig. 1.

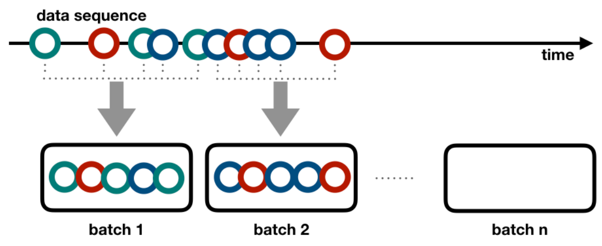


Fig. 1. Online phase of the FEPDS algorithm where data are collected in batch with a fixed length.

FEPDS uses a memory structure in order to store each set of fuzzy patterns (fPS) extracted by the learning algorithm when processing the different batches of data. It is a sliding window (SW) of a fixed size. In this way, the memory structure employs a first-in-first-out (FIFO) policy when the capacity of the window is full. In particular, the size of the SW is initialised for the storage of m fPSs, and it is assumed that knowledge extracted more than m data batches ago is old enough to be discarded for the complete evolutionary process. Therefore, the algorithm forgets old knowledge while it is able to remember the most recent for its use in the current learning process. In this way, the recommended size of SW is between 3 and 5, according to the execution time restrictions. Nevertheless, it is recommended to set this size to 5 whenever it is possible as recent knowledge with relevant information could be lost otherwise. It is important to remark that the patterns of the fPS included in (SW_t), together with the associated quality information for each fEP are queued as a single element in SW.

The information in SW is employed within the learning process. In particular, it is employed within the reward on the token-competition-based procedure which is detailed in Section III-C3. In this way, the procedure is able to bias the current learning in order to provide a good adaptation to the stream as it is influenced by good previously extracted patterns.

B. Learning adaptation process

The adaptation to the drift in FEPDS is based on a blind strategy with an adaptive learning approach where the model is updated for each batch. The employment of a blind scheme allows a good adaptation to gradual changes as it is continuously adapting to the changes. The algorithm initialises fPS and the SW structure as empty. Once the first batch is completed, data are continuously received for use in the following stages. Meanwhile, the evolutionary learning method obtains knowledge which is based on the current batch together with the previous insights stored in SW, if available, and this new fPS is incorporated into the SW structure. In this way, the SW is incrementally updated by means of adding the previous knowledge in the current evolutionary process. This is because these insights contain a valuable summary of the stream. In this way patterns that appeared frequently will have more weight in the evolutionary process. The application of these mechanisms to previous knowledge avoids the inclusion of non-relevant information within the current evolutionary process.

Finally, it is very important to highlight that despite the fact that the algorithm employs adaptive learning without the detection of changes (blind strategy), the use of the SW brings the algorithm some insights about past knowledge and the behaviour of FEPDS with respect to the stream.

C. Multi-objective evolutionary learning approach

The learning algorithm within FEPDS is a multi-objective EFS (MOEA) for obtaining fEPs. The determination of the quality of the fEPs is necessary to properly guide the search process. This is determined by several objectives, calculated using quality measures such as those presented in Table I. In particular, the objectives employed in the evolutionary process are WRAcc and SuppDiff, as they provide a good trade-off between generality and reliability for EPM. Moreover, these measures allows us to properly deal with input space drift as they contain specific elements that determine the posterior distribution of class memberships and the distribution of examples covered for the pattern class, respectively, which are the main elements of real and virtual drift [49], [20].

An elite population is employed in order to store the best fPS found so far for the current batch of data. However, the addition of information about the evolution of the stream is key to fitting the knowledge to the underlying phenomena which generate the data in order to handle issues related to the concept drift. In this way, when the elite population is being updated, a token-competition-based procedure is applied [50]. This procedure employs the information stored in SW to take into consideration the evolution of the stream in order to properly guide the search process. In the following subsection, the main elements of the evolutionary learning algorithm of FEPDS are presented.

1) Pattern representation: The EFS uses a "chromosome = pattern" approach [51] whereby each individual in the population represents a potential pattern. In FEPDS, fuzzy logic is used for the representation of numeric variables by means of linguistic labels (LLs). In the absence of expert knowledge, by default, the shape of these LLs is predetermined by means of uniform fuzzy partitions with the same number of LLs for all the variables, as it is one of the most commonly used strategies in these cases [52]. The memberships functions defined in these cases are usually based on classic membership functions such as triangular, trapezoidal, gaussian, and so on [53]. For FEPDS, triangular membership functions have been chosen as they are easy to define and to compute. An example of this predefined LLs is shown in Fig. 2, where five LLs are defined. Moreover, FEPDS allows the user to defined their own LLs by means of introducing a Java class with the desired membership function.

The EFS extracts patterns following a disjunctive normal form (DNF) representation because it is a good knowledge representation for descriptive EPs [11]. DNF patterns are codified in the evolutionary algorithm by means of a bit-vector genotype whose length is equal to the total number

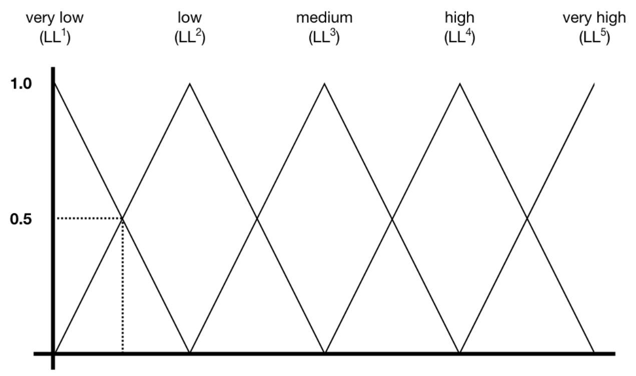


Fig. 2. Representation of a continuous variable with five linguistic labels.

of elements. The number of elements is determined by the number of possible categories for nominal variables, while for numerical variables it is the number of LLs used. In addition, the consequent part is represented by an integer value which allows the extraction of patterns for all classes in a single execution. A fEP and its representation in FEPDS can be observed in Fig. 3.

$$\begin{array}{c} \textit{Genotype} \\ \begin{array}{|c||c||c||c||c|} X_1 & X_2 & X_3 & X_4 & Class \\ 1\ \emptyset\ 1 & 1\ 1\ 1 & 1\ \emptyset\ \emptyset\ \emptyset & \emptyset\ \emptyset\ \emptyset & 2 \end{array} \\ \Downarrow \\ \textit{Phenotype} \\ IF(X_1 = (Low \vee High)) \wedge (X_3 = Arts) \\ THEN\ (Class = Negative) \end{array}$$

Fig. 3. Representation of a fuzzy DNF pattern with continuous and categorical variables in FEPDS.

2) Genetic operators: Following the adaptive learning approach, the evolutionary algorithm employs an initialisation procedure which incorporates the current fPS model in order to update it. After that, the remaining individuals are randomly initialised until the population is filled.

The genetic operators employed are the classical binary tournament selection [54], the multi-point crossover operator [55] and an oriented mutation operator. This mutation operator removes a variable of a pattern or randomly changes a gene with the same probability.

Finally, a reinitialisation operator is employed in order to avoid falling into a local maxima. In this procedure, the memory SW is employed when the Pareto front does not evolve, i.e. it is not able to cover new examples, during at least 25% of the total evaluations; and during the final stage it is used to obtain the final fPS model. With this procedure the elite population is updated by means of the reward following a token-competition-based procedure which is described below.

3) Reward in the token-competition-based procedure: FEPDS introduces a function in order to take into consideration the underlying phenomena in the stream to bias the search process. These underlying phenomena are known by the patterns extracted in previous batches of the stream. In this way, areas of the search space close to recently extracted patterns are potentially interesting. Therefore, the idea of FEPDS is to positively reinforce recently extracted patterns in the stream as it provides relevant information about its evolution. However, it is important to remark that usually the older the pattern, the lesser the accuracy in the description of the current state of the stream it has as concept drift can occur. In this way, if a pattern was good a few moments ago, the probability of its being a good pattern again, or the area close to it, is very high. Nevertheless, this probability decays as time goes by. Therefore, gradually less interest should be given to older patterns. Reward in the token-competition-based procedure is applied for the complete main population (P_g) when the token-competition-based procedure is applied to the reinitialisation process, in order to move to promising areas of the search space. The main idea is to reward the patterns extracted in previous stages, as they contain not only relevant insights but the capacity to simplify the model as fewer patterns are needed to cover the example space. Specifically, the operator incorporates SW with the function $SW\ (fEP_i, j)$ that returns a value of one if the pattern fEP_i is in the set of patterns fPS_j, or zero otherwise. A value is obtained for each individual as follows:

$$Div_t(fEP_i) = WRAcc_t(fEP_i) + \sum_{j=t-n}^{t} SW(fEP_i, j) \cdot 2^{-(t-j)} \cdot WRAcc_j(fEP_i) \qquad (2)$$

where $WRAcc_t(fEP_i)$ is the value of the unusualness measure in the current batch t for the pattern i, and $WRAcc_j(fEP_i)$ is the unusualness of the pattern on a previous batch j. It can be observed that the accumulated reward decreases by means of an exponential decay factor in order to gradually forget old knowledge.

The complete population is ordered with the Div_t values (from high to low) and the token competition operator is applied afterwards [40]. In this way, individuals with the highest diversity values will exploit its niches by seizing as many tokens as they can. The operator allows us to obtain a fPS where all patterns cover at least one example of the dataset not yet covered by other stronger patterns, i.e., with a better Div_t value. In order to complete the population of the next generation (P_{g+1}) the new individuals are generated through the guided reinitialization procedure [11], with the objective of searching into other promising areas of space so that the algorithm can cover all the examples.

The reward component incorporated into the algorithm allows an adaptation for the possible drift changes.

D. Operational scheme of FEPDS

In Fig. 4 a graphical representation of the working scheme of FEPDS is presented. In addition, Alg. 1 presents the pseudocode of FEPDS in order to ease its understanding. Firstly, FEPDS uses a test-then-train evaluation approach (lines 6-8). The evolutionary learning algorithm updates fPS for each batch of data while the test-then-train approach allows us the evaluation of the previous insights gained against the current

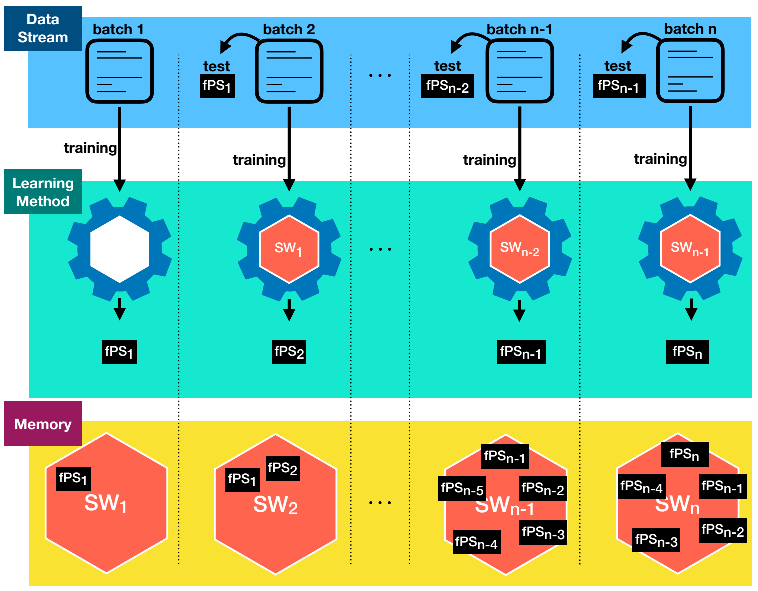


Fig. 4. General working schema of the FEPDS algorithm.

batch of data. Therefore, fPS_1 is evaluated with the second batch, fPS_2 with the third, and so on.

After that, following the blind adaptive learning strategy, the algorithm is executed for each batch of data. Once the batch is completed (lines 3-5), and the current fPS has been tested against this batch (line 7), the evolutionary learning process is triggered for updating the current fPS following the adaptive learning approach (lines 9-27). The evolutionary learning algorithm starts the initial population P_0 using the initialisation operator that includes the current fPS model (line 10). Next, P_0 is evaluated to determine its quality (line 11). Then the evolutionary process begins (lines 13-24), until a maximum number of generations is reached or a new batch is available. Within this process, P_g creates an offspring population Off_g by means of the application of the genetic operators (line 14), while Off_g is evaluated afterwards (line 15). These populations are joined together and the fast non-dominated sorting algorithm [56] is applied (lines 16-17). Finally, the application of the reinitialisation is checked. If the population is stagnated, P_{g+1} is filled by means of the reward in the token-competition-based procedure (line 19), which makes use of SW. Otherwise, P_{g+1} is filled by introducing the first k whole fronts or the first k individuals of a front if there is no room for the whole front (line 21). At the end of the evolutionary process, the reward in the token-competition-based procedure is applied once again to filter overlapped patterns, boosting those ones which are able to remain in time (line 25). Finally, the updated fPS is stored in SW which follows a first-in-first-out (FIFO) strategy and the updated fPS is returned (lines 26-27). It is important to remark that the whole process is repeated until the end of the data stream.

Algorithm 1 Operational scheme of the FEPDS algorithm.

```
1: $t \leftarrow 0$
2: **repeat**
3:     **while** Batch size is less than $n$ **do**
4:         Collect data from the stream.
5:     **end while**
6:     **if** $fPS_t \neq \emptyset$ **then**
7:         Test $fPS_t$ against current batch.
8:     **end if**
9:     $g \leftarrow 0$
10:    $P_g \leftarrow$ Initialisation($fPS_t$)
11:    Evaluate($P_g$)
12:    $t \leftarrow t + 1$
13:    **while** $g < maxGens$ **and** no new batch is available **do**
14:        $Off_g \leftarrow$ GeneticOperators($P_g$)
15:        Evaluate($Off_g$)
16:        $R_g \leftarrow P_g \cup Off_g$
17:        $F \leftarrow$ DominanceSorting($R_g$)
18:        **if** ParetoFront($F$) does not evolve **then**
19:            $P_{g+1} \leftarrow$ RewardInTokenCompetition($F$, $SW$)
20:        **else**
21:            $P_{g+1} \leftarrow$ FillByFronts($F$)
22:        **end if**
23:        $g \leftarrow g + 1$
24:    **end while**
25:    $fPS_t \leftarrow$ RewardInTokenCompetition($F$,$SW$)
26:    Add $fPS_t$ in $SW$
27:    **return** $fPS_t$
28: **until** The end of the stream
```

IV. Experimental study

A complete experimental study is carried out in order to determine the quality of the proposed algorithm. The aim of this study is threefold: firstly, the adaptation of FEPDS to concept drift is analysed. After that, a second study where the quality of the knowledge is analysed is carried out. Finally, the scalability of the proposal is analysed in order to test its stability in the long run. These studies are shown in this section. Firstly, the experimental framework is presented in Subsection IV-A. Then the method is analysed for concept drift adaptation in Subsection IV-B. The quality of the insights extracted is analysed in Subsection IV-C. Finally, a scalability analysis of FEPDS is presented in Subsection IV-D.

A. Experimental framework

In this section, the details of the experimental framework are shown in order to ease the reproducibility of this study. First, the datasets employed are presented. Next, the parameters employed by FEPDS are shown. Finally, the quality measures employed for determining the quality of the insights are presented. For further information and tools to reproduce this study, please refer to our GitHub repository[1].

a) Datasets: For the first study, datasets were artificially generated using the MOA software [57]. Two sets of three datasets each were generated, one set containing an abrupt drift and the other containing a gradual drift. In both cases, drifts occur every 50.000 instances. For gradual drift, the size of the change window is 10.000. All datasets were generated using 500.000 instances, so 9 changes were produced.

For the second and third study, a set of 94 artificial datasets with one million instances was generated using several streams generators from MOA. For each dataset five different streams were generated changing its random seed. In this way, results can be averaged in order to avoid biases.

b) Parameters: A good trade-off between quality and performance is key in data stream mining environments. In this way, FEPDS contains two types of parameters: the ones related to the data stream and its processing, and the ones related with the evolutionary learning method proposed. The parameters that influences the processing of the stream are the data chunk size and the SW size. Data chunk size has been set to 2500 instances, according to the results presented in Section IV-D. It is also considered an optimal SW size of 5, according to the recommendations presented previously, due to the size of this experimental study.

Parameters related to the evolutionary algorithm are crossover and mutation probabilities, population size, number of LLs and number of generations. The crossover and mutation probabilities, and the population size has been set to 0.6, 0.1 and 50 respectively. These values have been considered as a standard within the majority of experimental studies for the SDRD task using evolutionary algorithms. On the other hand, the number of LLs and the number of generations have been determined by means of a complete experimental study. The latter it is really important in this study as it significantly influences the balance between the quality of the knowledge and performance. The complete study is available at https://simidat.ujaen.es/papers/FEPDS. According to this results, the number of LLs and generations was set to 3 and 60 respectively.

According to the results presented, the number of LLs and the number of generations has been set to 3 and 60, respectively, as they provide a good balance between quality of results and execution time.

c) Quality measures: The confidence is the conditional probability of the class, given the pattern [49]. When a real concept drift occurs, the average confidence of the extracted fPS could be significantly affected. Therefore, the confidence is analysed in order to determine the behaviour and robustness of the algorithm with respect to real concept drift in the first study. In addition, it is interesting to determine the execution time in order to determine whether the concept drift affects the efficiency.

For the second study, the quality measures shown in the results tables are the ones related to the description of the main characteristics of descriptive patterns in EPM [4], i.e. the average number of patterns (n_r) and the average number of variables (n_v) needed to measure the conciseness of the model extracted; the average WRAcc needed to compute the novelty and reliability; the average GR, Conf and FPR needed to estimate the reliability; and the average TPR needed to calculate the generality. It is important to remark that the results shown are the average of five executions using different seeds in the stream generators.

Finally, for the scalability analysis, the results presented in Table III are related to the average execution time and memory consumption per data chunk on all the analysed data streams. In addition, Figure 6 presents a chart for a specific dataset in order to determine its behaviour through the execution of the method in order to determine its stability.

d) Run environment: The studies were performed on a computation cluster composed of 8 nodes with 2 x Intel Xeon CPU 3065 at 2.33GHz and 7 GB of RAM each one, running Rocks Cluster 6.1.1 Sand Boa.

B. Concept drift adaptation analysis

Fig 5 shows the performance of FEPDS in terms of average confidence of the patterns extracted on each data chunk against different concept drifts. In fact, Figs. 5a, 5c and 5e contain an abrupt drift while Figs. 5b, 5d and 5f present a gradual one. It is important to remark that each drift is marked with a dashed, vertical line on each plot.

In all the datasets analysed the performance after the first drift changes significantly. After this point the algorithm is able to recover rapidly from the drift. In general, the performance of the algorithm remains stable regardless of the concept drift. Therefore, the knowledge extracted is robust against this type of concept drift. This fact can be produced by the employment of fuzzy logic on numerical variables. The use of fuzzy logic allows us to handle some uncertainty which is useful when the concept changes as it provides a tolerance level for small or gradual changes without affecting performance. In addition,

[1]https://github.com/SIMIDAT/FEPDS

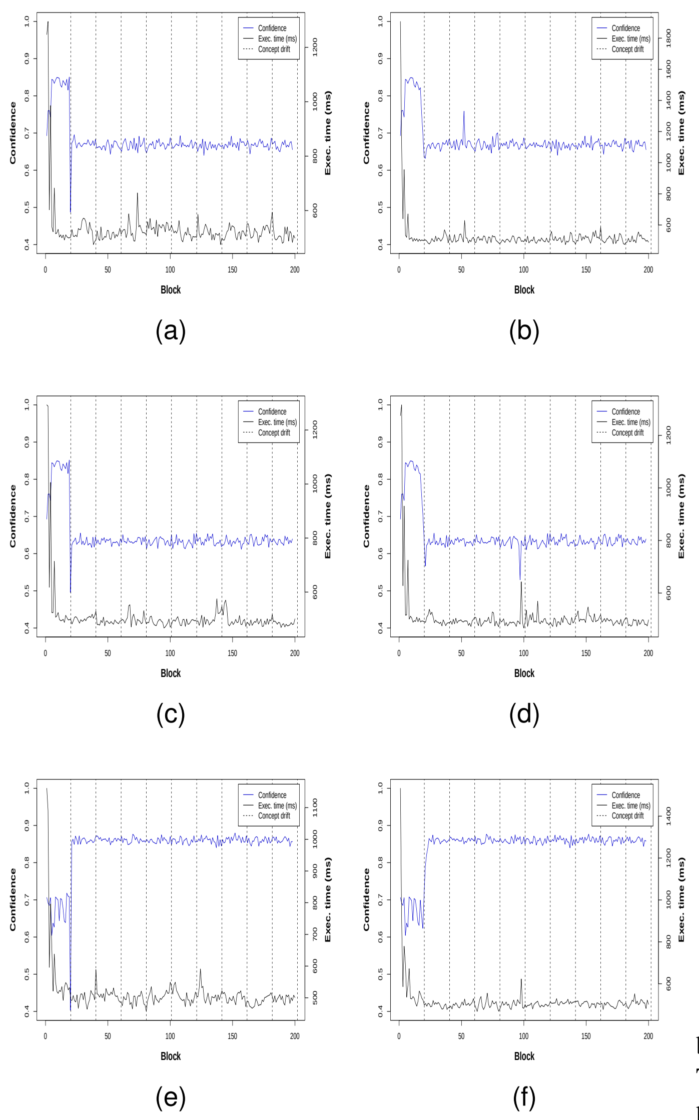


Fig. 5. Confidence and execution time performance of FEPDS on different datasets with concept drift. (a), (c), (e) are datasets with an abrupt drift, while (b), (d), (f) are datasets with a gradual drift.

the evolutionary process includes a learning stage that is able to provide a fast adaptation to the change. This tolerance level, together with the evolutionary process, allows the extraction of robust patterns with a good trade-off between their quality and their adaptability to changes.

Finally, it can be observed that the adaptive learning approach is not significantly affected by concept drifts in terms of execution time. In fact, the execution time is in general close to 600ms, which is fast enough to provide a real-time response.

C. *Quality analysis*

Table II presents the average fEP quality across the data stream, together with the average complexity of the extracted fPS. From this table, an analysis of the different aspects of EPM is depicted below:

TABLE II
AVERAGE RESULTS OF FEPDS.

n_r	n_v	WRACC	CONF	GR	TPR	FPR
2.38	3.62	0.75	0.78	0.99	0.69	0.18

- Complexity of the model. This is measured by n_r and n_v. In general, the fPS extracted is simple. In fact, it is composed of almost one fEP per class, with a low number of variables. This simplicity allows a fast analysis of the insights extracted, which is key in the data stream mining context.
- Interest. This is determined by WRAcc. The results extracted are of great interest. This is due to the high trade-off between the generality, in terms of covered instances, and the reliability, according to the amount of correctly covered instances. Therefore, the fEPs could be interesting for the experts as they are a good approximation of the underlying phenomena in the data.
- Generality. This is determined by TPR. In general the fEPs extracted contains a high generalisation capacity as each one is able to cover 69% of all positive instances. Therefore, the insights extracted by the proposed algorithm concern a high number of target instances.
- Reliability. This aspect is determined by CONF, GR and FPR. 78% of instances covered by each fEP are covered correctly. In addition, the high GR percentage indicates that almost all the patterns extracted are EPs on test data. In addition, although the amount of FPR is high, its ratio to TPR determines that the average fEP extracted by the proposed method contains a high discriminative power and it is trustworthy.

In EPM it is necessary to search for a good trade-off between three objectives: simplicity, generality and reliability. The average fPS extracted by FEPDS is simple enough to perform a fast analysis which is relevant in the data stream mining context. Although the model is simple, it does not concern its reliability as the patterns extracted contains a high discriminative power together with a high confidence. In addition, the fPS extracted is able to cover a large amount of target instances. Therefore, using the insights extracted, experts are able to extract interesting, efficient conclusions about the underlying phenomena in data.

D. *Processing speed and scalability analysis*

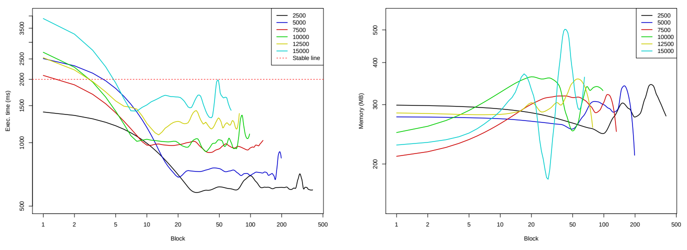
It is assumed that a data stream mining algorithm is deployed in lifelong environments. Therefore, it is desirable that the processing time of these algorithms is below the arriving rate of instances in order to assert the stability of the system. In addition, memory consumption should remain immutable throughout time.

Table III presents the average runtime and memory consumption for each data chunk on the analysed data streams. If data chunks arrive every two seconds, FEPDS is able to process instances at approximately 5 KHz. In fact, the complexity scales up linearly but slowly with respect to the instance rate. Therefore, FEPDS is fast enough to provide a

TABLE III
AVERAGE EXECUTION TIMES (IN MILLISECONDS) AND MEMORY CONSUMPTION (IN MEGABYTES) OF FEPDS WITH DIFFERENT DATA BLOCK SIZE.

Block size	Exec. Time (ms)	Memory (MB)
2500	1134.57 ± 30.76	273.41 ± 2.29
5000	1420.16 ± 40.66	299.19 ± 2.12
7500	1629.15 ± 52.36	310.16 ± 9.32
10000	2024.45 ± 88.07	324.04 ± 3.08
12500	2319.50 ± 111.02	326.85 ± 15.78
15000	2187.56 ± 130.64	329.48 ± 5.59

real-time response on a huge set of problems. Finally, it can be observed that memory consumption is low enough to be executed on many commodity hardware.



(a) (b)

Fig. 6. Scalability analysis for the *AG-c1-1* data stream with respect to different data block size. (a) processing time, (b) memory consumption.

Finally, Fig. 6 shows the performance in terms of execution time and memory consumption for the artificial data stream *AG-c1-1* employed in this study with different batch sizes. The graphs are presented using logarithmic scale with respect to the number of blocks. In terms of execution time, the algorithm remains stable, i.e., it is below the data chunk arriving rate represented as the stable line, after the processing of the first blocks of data due to memory and instructions caching. On the other hand, memory consumption remains stable with some variations due to the triggering of the Java garbage collector. In conclusion, FEPDS is an alternative for permanent deployment in lifelong environments.

V. A CASE STUDY: DETERMINATION OF PROFILES OF NEW YORK CITY CAB CUSTOMERS ACCORDING TO THEIR FARE AMOUNT

One of the most popular images of New York City (NYC) is its yellow cab. The fleet is regulated by the NYC Taxi & Limousine Commission (NYC-TLC), which is formed of 13587 medallion cabs. One of the characteristics of these vehicles is that to be picked-up by one of them the passenger must hail the driver in the street. Therefore, no previous booking is needed before travelling.

In this case study the characteristics of passengers with respect to their fare amount is analysed. The idea is to continuously analyse the behaviour of people in order to improve the service.

A. *Characteristics of data*

The data analysed consist of all the records in 2017 of the journeys made by the yellow cabs[2]. Moreover, basic data formatting and data cleaning procedures have been carried out such as removing outliers and inconsistencies, unnecessary variables, and so on. To reproduce this preprocessing, please refer to our repository.

The final dataset is composed of almost 100 million instances with 10 variables each. It contains a class value that represents the amount of money spent on each journey. As can be observed, the amount of money spent is a numerical variable, therefore it must be discretised in order to be handled by FEPDS. This process was carried out by means of an expert criterion, determining the following intervals: [0, 15), [15, 30), [30, ∞], which correspond to low, medium and high fares. Finally, a brief description of each nominal (nom) and numerical (num) variable in the dataset is presented in Table IV.

TABLE IV
DESCRIPTION OF VARIABLES USED IN NYC-TLC TAXIS DATA.

Name	Type	Description
passenger_count	num	Passengers in the journey
RateCodeID	nom	The rate in effect during the journey
store_and_fwd_flag	nom	Connection with the server
payment_type	nom	The type of payment made
extra	nom	Extra surcharges (Rush & overnight)
tip_amount	num	The amount of money spend on tips
tolls_amount	num	The amount of money spend on tolls
PUBorough	nom	The pick-up zone
DOBorough	nom	The drop-off zone
total_amount	nom	The total fare of the journey

B. *Extraction of emerging patterns in NYC-TLC data stream by FEPDS*

TABLE V
AVERAGE RESULTS OF FEPDS ON THE NYC TAXIS DATASET.

n_r	n_v	WRACC	CONF	GR	TPR	FPR
3.00	3.63	0.78	0.62	1.00	0.72	0.16

Table V presents the average quality results for each individual fEP for each data chunk analysed, except for n_r and n_v which correspond the whole set of patterns extracted for that data chunk. An analysis of these results is presented below:

- Complexity of the models. FEPDS constantly returns three fEPs, which means that the method extracts only one fEP for each class. On the other hand, the average number of variables of each pattern is low. Therefore, experts are able to perform a fast, easy analysis of the knowledge extracted.
- Interest. The results extracted show an excellent average WRAcc value of 0.78. This means that the patterns extracted contain an excellent trade-off between the coverage of the target instances and their reliability. In fact,

[2]https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

this interest can be observed in the high TPR value together with an acceptable confidence value. This fact is due to the use of the reward in the TC-based procedure evaluation function making it possible to keep those relevant patterns while the evolutionary process searches for other interesting knowledge in order to adapt to potential changes. Therefore, the patterns extracted by FEPDS are very interesting for the expert because they present an excellent trade-off between generality and reliability.
- Generality. On average, 72% of the target instances are covered for each pattern, as can be observed from the TPR. This is quite relevant for the experts as the knowledge extracted is able to provide a wide view of the general behaviour of customers, so better decisions can be carried out.
- Reliability. Firstly, it can be observed that all the patterns extracted are EPs because the GR value is 1. In addition, the ratio between the TPR and the FPR of the extracted fEPs is high, so they have a great discriminative power. On the other hand, the confidence level is high enough to trust on the knowledge extracted, especially when the level of generality is very high. Therefore, the patterns extracted present reliable descriptions which allows the experts to extract robust conclusions from them.

Additionally, the most repeated patterns extracted throughout the stream are presented together with their average quality measures. The idea is to determine recurrent patterns in the stream analysed so a general profile can be extracted.

Table VI presents the two most repeated patterns throughout the stream for each class. In general, the quality of these patterns is very good, highlighting the high levels of GR and WRACC, which means that the patterns are highly discriminative. In this way, we can trust in the reliability of the patterns extracted.

Several conclusions can be extracted from this table. For example, it can be observed that many of the cheapest journeys are produced within the working area of Manhattan. For medium fares, it can be observed that the destinations are usually residential or leisure areas of Manhattan. On the other hand, for the highest fares it can be observed that users are usually picked up in those areas where people usually work. Therefore, from these results the day-to-day lives of people can be observed, where first they go to their workplace and then they return home. Finally, it is important to remark on the presence of the payment by card on those journeys with medium and high fares.

VI. Concluding remarks

This paper presents an algorithm for extracting $fEPs$ in data stream environments. FEPDS processes stream by means of a batch strategy where instances are collected. Once a batch is completed, the learning method of the algorithm updates the fPS model using the data of the current batch. Specifically, the algorithm is based on a blind strategy with adaptive learning whereby the model is updated for each batch. In addition, FEPDS uses a learning method based on a multi-objective evolutionary algorithm able to extract pattern models with a very good trade-off between the simplicity of the model and its reliability. This learning method employs the fPSs obtained in previous stages by means of a memory structure based on a sliding window (SW) in order to reward patterns that sometimes appear in the window. Finally, the algorithm uses a test-then-train evaluation where the model is continuously evaluated against unseen instances in order to determine the quality of the knowledge extracted.

FEPDS has been tested in an experimental study. Firstly, the adaptability of the proposed algorithm to different kinds of concept drifts was analysed. The conclusion is that FEPDS is able to adapt the model properly with respect to abrupt and gradual drift without decreasing performance in terms of execution time. The quality of the insights extracted by FEPDS contains an interesting ratio between the coverage of target instances and their reliability, together with a fPS that can be analysed easily. In addition, the proposed algorithm provides a great scalability and performance in terms of execution time and memory consumption. This allows its deployment in lifelong environments.

Finally, a case study was carried out. In particular, the profile of the customers of the New York City taxis according to their fare amount was analysed. In general, the results extracted by FEPD are of a high quality. In particular, the generality of each individual pattern is highlighted. Moreover, an analysis of recurrent patterns was carried out, from which slightly recurrent behaviours such as payment by card on medium or high fares and the daily travel to work of new yorkers have been extracted.

References

[1] J. Gama, *Knowledge discovery from data streams*. CRC Press, 2010.

[2] I. Žliobaitė, M. Pechenizkiy, and J. Gama, "An overview of concept drift applications," in *Big Data Analysis: New Algorithms for a New Society*. Springer, 2016, pp. 91–114.

[3] G. Dong and J. Li, "Efficient mining of emerging patterns: Discovering trends and differences," in *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 1999, pp. 43–52.

[4] A. M. García-Vico, C. J. Carmona, D. Martín, M. García-Borroto, and M. J. del Jesus, "An overview of emerging pattern mining in supervised descriptive rule discovery: Taxonomy, empirical study, trends and prospects," *WIREs: Data Mining and Knowledge Discovery*, vol. 8, no. 1, 2018.

[5] P. Kralj-Novak, N. Lavrac, and G. I. Webb, "Supervised Descriptive Rule Discovery: A Unifying Survey of Constrast Set, Emerging Pattern and Subgroup Mining," *Journal of Machine Learning Research*, vol. 10, pp. 377–403, 2009.

[6] G. Li, R. Law, H. Q. Vu, J. Rong, and X. R. Zhao, "Identifying emerging hotel preferences using emerging pattern mining technique," *Tourism management*, vol. 46, pp. 311–321, 2015.

[7] Y. Yu, K. Yan, X. Zhu, and G. Wang, "Detecting of PIU Behaviors Based on Discovered Generators and Emerging Patterns from Computer-Mediated Interaction Events," in *Proc. of the 15th International Conference on Web-Age Information Management*, vol. 8485, 2014, pp. 277–293.

[8] J.-P. Métivier, A. Lepailleur, A. Buzmakov, G. Poezevara, B. Crémilleux, S. O. Kuznetsov, J. L. Goff, A. Napoli, R. Bureau, and B. Cuissart, "Discovering structural alerts for mutagenicity using stable emerging molecular patterns," *Journal of chemical information and modeling*, vol. 55, no. 5, pp. 925–940, 2015.

[9] C.-H. Weng and C.-K. H. Tony, "Observation of sales trends by mining emerging patterns in dynamic markets," *Applied Intelligence*, pp. 1–15, 2018.

TABLE VI
MOST REPEATED PATTERNS FOR EACH CLASS AND THEIR AVERAGE QUALITY THROUGHOUT THE NYC-TLC DATA STREAM.

Pattern	Repetitions	n_v	WRACC	CONF	GR	TPR	FPR
R_1: IF DOBorough = (Downtown, Midtown, Central Park, Uppertown) THEN LOW [0,15)	4633	1	0.76	0.86	2.29	0.93	0.42
R_2: IF DOBorough = (Downtown, Midtown, Central Park) THEN LOW [0,15)	3340	1	0.78	0.88	2.97	0.88	0.32
R_3: IF DOBorough = (Uppertown, Other, Central Park) THEN MEDIUM [15,30)	448	1	0.71	0.59	4.03	0.59	0.16
R_4: IF payment_type = credit_card & tip_amount = LL_3 THEN MEDIUM [15,30)	199	2	0.71	0.84	15.10	0.45	0.03
R_5: IF PUBorough = (Central Park, Midtown, Downtown, Other) THEN HIGH [30, ∞)	630	1	0.85	0.23	16.72	0.78	0.08
R_6: IF payment_type = credit_card & tip_amount = (LL_4, LL_6) THEN HIGH [30, ∞)	341	2	0.87	0.38	33.47	0.76	0.02

[10] A. M. García-Vico, P. González, M. J. del Jesus, and C. J. Carmona, "A first approach to handle emergining patterns mining on big data problems: The evaefp-spark algorithm," in *IEEE International Conference on Fuzzy Systems*, 2017, pp. 1–6.

[11] A. M. García-Vico, C. J. Carmona, P. González, and M. J. del Jesus, "MOEA-EFEP: Multi-objective evolutionary algorithm for extracting fuzzy emerging patterns," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 5, pp. 2861 – 2872, 2018.

[12] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 44:1–44:37, 2014.

[13] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Woźniak, "Ensemble learning for data stream analysis: A survey," *Information Fusion*, vol. 37, pp. 132–156, 2017.

[14] A. Sayuri-Iwashita and J. Papa, "An overview on concept dripft learning," *IEEE Access*, vol. 7, pp. 1532–1547, 2019.

[15] M. M. Gaber, "Advances in data stream mining," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 79–85, 2012.

[16] A. Bifet, "Adaptive learning and mining for data streams and frequent patterns," Ph.D. dissertation, Universitat Politècnica de Catalunya, 2009.

[17] M. Sayed-Mouchaweh and E. Lughofer, *Learning in non-stationary environments: methods and applications*. Springer Science & Business Media, 2012.

[18] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Wozniak, and F. Herrera, "A survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing*, vol. 239, pp. 39–57, 2017.

[19] G. I. Webb, L. K. Lee, B. Goethals, and F. Petitjean, "Analyzing concept drift and shift from sample data," *Data Mining and Knowledge Discovery*, vol. 32, no. 5, pp. 1179–1199, 2018.

[20] I. Khamassi, M. Sayed Mouchaweh, M. Hammami, and K. Ghédira, "Discussion and review on evolving data streams and concept drift adapting," *Evolving Systems*, vol. 9, no. 1, pp. 1–23, 2018.

[21] E. Lughofer, "On-line active learning: A new paradigm to improve practical useability of data stream modeling methods," *Information Sciences*, pp. 356–376, 2017.

[22] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 97–106.

[23] J. Gama and G. Castillo, "Learning with local drift detection," in *Proceedings of the Second International Conference in Advanced Data Mining and Applications, ADMA, Xi'an, China, August 14-16*, 2006, pp. 42–55.

[24] I. Khamassi and M. Sayed Mouchaweh, "Drift detection and monitoring in non-stationary environments," in *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems, EAIS 2014, Linz, Austria, June 2-4, 2014*, 2014, pp. 1–6.

[25] A. Shaker and E. Lughofer, "Self-adaptive and local strategies for a smooth treatment of drifts in data streams," *Evolving Systems*, vol. 5, no. 4, pp. 239–257, 2014.

[26] H. Toubakh and M. Sayed-Mouchaweh, "Hybrid dynamic data-driven approach for drift-like fault detection in wind turbines," *Evolving Systems*, vol. 6, no. 2, pp. 115–129, 2015.

[27] E. Lughofer, "Evolving fuzzy systems fundamentals, reliability, interpretability, useability, applications," in *Handbook on Computational Intelligence: Volume 1: Fuzzy Logic, Systems, Artificial Neural Networks, and Learning Systems*. World Scientific, 2016, pp. 67–135.

[28] I. Škrjanc, J. A. Iglesias, A. Sanchis, D. Leite, E. Lughofer, and F. Gomide, "Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A survey," *Information Sciences*, vol. 490, pp. 344–368, 2019.

[29] N. Japkowicz and M. Shah, *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.

[30] A. Wald, *Sequential analysis*. Courier Corporation, 1973.

[31] M. Pratama, C. Za'in, A. Ashfahani, Y. S. Ong, and W. Ding, "Automatic construction of multi-layer perceptron network from streaming examples," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 1171–1180.

[32] M. Pratama, W. Pedrycz, and G. I. Webb, "An incremental construction of deep neuro fuzzy system for continual learning of non-stationary data streams," *IEEE Transactions on Fuzzy Systems*, In Press.

[33] M. Pratama, M. de Carvalho, R. Xie, E. Lughofer, and J. Lu, "Atl: Autonomous knowledge transfer from many streaming processes," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 269–278.

[34] M. J. del Jesus, P. González, F. Herrera, and M. Mesonero, "Evolutionary Fuzzy Rule Induction Process for Subgroup Discovery: A case study in marketing," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 4, pp. 578–592, 2007.

[35] M. García-Borroto, J. Martínez-Trinidad, and J. Carrasco-Ochoa, "Fuzzy emerging patterns for classifying hard domains." *Knowledge and Information Systems*, vol. 28, no. 2, pp. 473–489, 2011.

[36] C. J. Carmona, P. González, M. J. del Jesus, and F. Herrera, "NMEEF-SD: Non-dominated Multi-objective Evolutionary algorithm for Extracting Fuzzy rules in Subgroup Discovery," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 5, pp. 958–970, 2010.

[37] E. Hüllermeier, "Fuzzy methods in machine learning and data mining: Status and prospects," *Fuzzy Sets and Systems*, vol. 156, no. 3, pp. 387–406, 2005.

[38] D. E. Goldberg, *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley Longman Publishing Co., Inc., 1989.

[39] F. Herrera, "Genetic fuzzy systems: taxomony, current research trends and prospects," *Evolutionary Intelligence*, vol. 1, pp. 27–46, 2008.

[40] C. J. Carmona, V. Ruiz-Rodado, M. J. del Jesus, A. Weber, M. Grootveld, P. González, and D. Elizondo, "A fuzzy genetic programming-based algorithm for subgroup discovery and the application to one problem of pathogenesis of acute sore throat conditions in humans," *Information Sciences*, vol. 298, pp. 180–197, 2015.

[41] A. Starkey, H. Hagras, S. Shakya, and G. Owusu, "A multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization," *Information Sciences*, vol. 329, pp. 390–411, 2016.

[42] O. Castillo, L. Cervantes, J. Soria, M. Sánchez, and J. Castro, "A generalized type-2 fuzzy granular approach with applications to aerospace," *Information Sciences*, vol. 354, pp. 165–177, 2016.

[43] M. Antonelli, D. Bernardo, H. Hagras, and F. Marcelloni, "Multiobjective evolutionary optimization of type-2 fuzzy rule-based systems for financial data classification," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 2, pp. 249–264, 2017.

[44] M. Babaei and M. Sheidaii, "Desirability-based design of space structures using genetic algorithm and fuzzy logic," *International Journal of Civil Engineering*, vol. 15, no. 2, pp. 231–245, 2017.

[45] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery: an overview," in *Advances in knowledge discovery and data mining*. Menlo Park, CA, USA: AAAI/MIT Press, 1996, pp. 1–34.

[46] C. J. Carmona, M. J. del Jesus, and F. Herrera, "A Unifying Analysis for the Supervised Descriptive Rule Discovery via the Weighted Relative Accuracy," *Knowledge-Based Systems*, vol. 139, pp. 89–100, 2018.

[47] W. Kloesgen, "Explora: A Multipattern and Multistrategy Discovery Assistant," in *Advances in Knowledge Discovery and Data Mining*. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1996, pp. 249–271.

[48] D. Gamberger and N. Lavrac, "Expert-Guided Subgroup Discovery: Methodology and Application," *Journal Artificial Intelligence Research*, vol. 17, pp. 501–527, 2002.

[49] M. García-Borroto, O. Loyola-González, J. F. Martínez-Trinidad, and J. A. Carrasco-Ochoa, "Evaluation of quality measures for contrast

patterns by using unseen objects," *Expert Systems with Applications*, vol. 83, pp. 104 – 113, 2017.

[50] K. S. Leung, Y. Leung, L. So, and K. F. Yam, "Rule Learning in Expert Systems Using Genetic Algorithm: 1, Concepts," in *Proc. of the 2nd International Conference on Fuzzy Logic and Neural Networks*, K. Jizuka, Ed., 1992, pp. 201–204.

[51] M. L. Wong and K. S. Leung, *Data Mining using Grammar Based Genetic Programming and Applications*. Norwell, MA, USA: Kluwer Academics Publishers, 2000.

[52] O. Cordón, F. Herrera, and P. Villar, "Analysis and guidelines to obtain a good uniform fuzzy partition granularity for fuzzy rule-based systems using simulated annealing," *International Journal of Approximate Reasoning*, vol. 25, no. 3, pp. 187 – 215, 2000.

[53] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice Hall, 1995.

[54] B. L. Miller and D. E. Goldberg, "Genetic Algorithms, Tournament Selection, and the Effects of Noise," *Complex System*, vol. 9, pp. 193–212, 1995.

[55] J. H. Holland, *Adaptation in Natural and Artificial Systems*, 2nd ed. University of Michigan Press, 1975.

[56] K. Deb, A. Pratap, S. Agrawal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[57] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: massive online analysis," *Journal of Machine Learning Research*, vol. 11, pp. 1601–1604, 2010. [Online]. Available: https://moa.cms.waikato.ac.nz/