KINGDOM OF SAUDI ARABIA
MINISTRY OF EDUCATION
ALBAHA UNIVERSITY

January – June 2020 Volume 4 Issue 1

ALBAHA UNIVERSITY JOURNAL OF
BASIC AND APPLIED
SCIENCES

Fuzzy Sub-Clusters: a Novel Class Detection Approach for Multi Data Streams

Adil Fahad

Department of Computer Science,
College of Computer Science and Information Technology,
Albaha University, Albaha, Saudi Arabia

p-ISSN:1658-7529
e-ISSN:1658-7537

A Refereed Academic Journal Published by Albaha University

ALBAHA UNIVERSITY JOURNAL OF

BASIC AND APPLIED SCIENCES

January- June 2020 Volume 4 Issue 1

CONTENTS

Editorial

Review Article

Research

Author Guidelines

https://portal.bu.edu.sa/web/jbas
Published by BUJBAS, Albaha University,
65451 Albaha, Kingdom of Saudi Arabia

All scientific articles in this issue are refereed.

All articles published in the Journal represent the opinion of the author(s) and do not necessarily reflect the views of the journal.

SCOPE

Albaha University Journal of Basic and Applied Sciences (BUJBAS) publishes English language, peer-reviewed papers focused on the integration of all areas of sciences and their application. Supporting the concept of interdisciplinary BUJBAS welcomes submissions in various academic areas such as medicine, dentistry, pharmacy, biology, agriculture, veterinary medicine, chemistry, mathematics, physics, engineering, computer sciences and geology.

BUJBAS publishes original articles, short communications, review articles, and case reports.

The absolute criteria of acceptance for all papers are the quality and originality of the research.

CHIEF OPERATING OFFICER

Papers for publication should be addressed to the Editor, via the website:
https://portal.bu.edu.sa/web/jbas
E-mail: bujs@bu.edu.sa

ONLINE SUPPORT

BUJBAS is published by Albaha University. For queries related to the journal, please contact
https://portal.bu.edu.sa/web/jbas
E-mail: bujs@bu.edu.sa

L.D. No: 1438/2732

p-ISSN: 1658-7529

e-ISSN: 1658-7537

Article available at Albaha University Journal of Basic and Applied Sciences

Journal of Basic and Applied Sciences

Journal homepage: https://portal.bu.edu.sa/web/jbas/

Fuzzy Sub-Clusters: a Novel Class Detection Approach for Multi Data Streams

Adil Fahad[a,*]

[a] *Department of Computer Science, College of Computer Science and Information Technology, Albaha University, Albaha, Saudi Arabia*

ARTICLEINFO

Article history:
Received 16 September 2019
Received in revised form
19 November 2019
Accepted 26 November 2019

Keywords:
Safe driving system
GPS
Intelligent systems
Mobile application

ABSTRACT

A few classification frameworks have focused on a new challenge in data streams, namely, concept-evolution, which occurs when a totally new class/concept appears in the stream. However, the recent class detection techniques dealing with this challenge focus on a single data stream and numeric data, and they have high false detection rates in many scenarios. This paper proposes a new classification technique, called FUzzy sub-clusters for Novel class Detection in multi data streams (FUND), that maintains a set of representatives of each class label, a set of clusters features, as well as extending the boundaries of the sub-clusters to allow the fuzziness nature of the data streams. In particular, FUND applies two similarity metrics, namely density and distance, to overcome the uncertainty issue in data streams and it distinguishes the instances that belong to an existing class from the potential novel class instances. The improvement in the detection relates to the classification accuracy by making sure that an instance of a novel class does not end up being classified as an existing class instance and vice versa. Experimental results show that FUND outperforms state-of-the-art data stream classification techniques in both synthetic and challenging real datasets. The average accuracy improvement is around 30% for FUND-C compared to MineClass and DNTC techniques.

1. Introduction

Recently research in mining of data stream environments has attracted attention [1-4]. In addition to those that consider a data stream at a time, more and more emerging applications are involved in monitoring multiple data streams concurrently. Currently, the most superior security surveillance systems, with advances in micro-electro-mechanical system (MEMS) and wireless communication technology, routinely gather data from multiple sources on multiple data streams.

For several applications, such as sensor networks and network monitoring, the data is in the form of streams, each of which is an infinite sequence of data points with explicit/implicit timestamps. The data also has special characteristics, such as dynamic data distribution (or concept-drift), concept-evolution and uncertainty. Nevertheless, the characteristics of data streams make classification for multi evolving data more challenging than that for a regular single stream, especially when they happen concurrently. In a multiple data streams application, it is not appropriate to analysis and process each stream individually for classification purpose as this will not help in overcoming the data stream challenges and leads to less accurate classification. If a new class (concept) evolves in the dataset, its detection will not possible if there are no enough cohesive instances that could form this new class. Processing instances from multiple evolving data streams helps discovering new concepts that may appear at any time in the streams, as instances from one stream could satisfy the cohesion and separation property with other instances from the other streams in the dataset. Furthermore, concept-drift of the target class could occur globally at the dataset level, whereas there is no concept-drift at the stream level.

Existing solutions however focus only on discovering the cross-relationships among streams, and they are not dealing with classification or/and novel class detection. The identification of a new or unknown instance/object that a (machine learning) system is not aware of during training refers to novelty detection [5]. This detection is one of the essential requirements of an efficient classification model because it overcomes the concept-evolution problem, as the dataset could have new information about instances that may not be known at the time of training the model. The main important aspect in novel class detection is that any class label of a dataset has the following property “instances belonging to the class will be far from the existing class instances and will be close to other novel class instances” [6].

This paper proposes a new approach; called FUzzy sub-clusters for Novel class Detection (FUND) that uses the advantages of the ensemble learning techniques for classifying upcoming data instances. The main objective of FUND is to guarantee the classification accuracy by maintaining an implicit concept description in the form of a set of Class Features. Here, outliers filtering is an intermediate step in novel class detection in the proposed model. FUND overcomes the concept-evolution problem by maintaining a set of representatives of each class

* Corresponding author: Department of Computer Science, College of Computer Science and Information Technology, Albaha University, 65451Albaha, Saudi Arabia.

Tel.: +966 53 939 2235.

E-mail address: afalharthi@bu.edu.sa (A. Fahad).

label, set of clusters features, as well as allowing the fuzziness nature of the data streams by extending the boundary of each sub-cluster. Novel class detection for a single data stream compares a data point with respect to the history data points, which used to build the classification model, in order to detect whether the data point is an outlier. In case of multiple data streams, a data point can be detected as an outlier either by comparing it to the history data points from the same stream or comparing it to the data points from the other streams in the dataset. The opportunity of having multiple data streams to compare facilitates better accuracy and richer semantics across the data streams. Therefore, the proposed research work takes the problem into another dimension, which is classification and novel detection on multi evolving data streams.

The rest of the paper is organized as follows. Section 2 then surveys the related work and puts our work context. Section 3 provides an overview of the proposed model, and describes the algorithms for FUND. Sections 4 provides details of datasets used in the benchmark and shows the various experimental results. The concluding remarks are given in Section 5.

2. Related Work

This work is related to classification and novel class detection, and mining multi data streams. This section briefly reviews previous research in these areas and point out the differences from the proposed work.

Research on mining multi data streams can be divided into two main kinds. One kind of works focuses on clustering the data streams and another kind of works discusses pattern discovery in multiple streams. However, all these works focus on performing clustering on the entire data set and do not consider classifying the data points as they detect patterns in a group of streams or cluster the streams not the data points.

The method of clustering is a very effective way to summarize multiple streams [7-9], through this method the similar patterned streams are placed together while the dissimilar ones are separated from the main stream. There has been a lot of research work done to identify and effectively differentiate between the streams. There is a new method that was proposed by Cao et al. [10] according to which the clusters were formulated based on their density. It helped discovering arbitrary-shapes and structures for evolving data streams with noise. The cluster method was used along with micro clusters, the clusters that do not have fixed shape are all grouped together and then the clusters that have definite shapes are all grouped together. This micro cluster system helps to differentiate between the fixed and non-fixed clusters.

In addition, an artificial system proposed by Nasraoui et al. [11], which used as immune system (TECNO-STREAMS), the system helps and supports the clustering approach. The sample data is extracted from the noisy Web click stream data [12]. There was an experiment designed by Aggarwal et al. [13,14] that turned out to be successful one where the data stream was clustered, it came to be known as the HPStream method. Through this application, the streams can be identified effectively and efficiently. COD, a framework of clustering multiple data streams, was proposed by Dai et al. [15]. COD dynamically clusters multiple data streams and minimizes the problems that occur in data mining. It allows adjustment of the data mining to meet flexible mining requirements.

To identify and summarize streams, the main attention is paid to identify the possible patterns that exist in the streams [16,17]. Today, mathematical tools are utilized to change the model that comes out as a result of the patterns obtained. The patterns are compressed or expanded as required to fit the model. The application SPIRIT [16] takes into account the variables that have been excluded when it calculates and develops the model. The program Dynammo [17] creates and recreates patterns by completing the missing values that exist within the patterns, to give it completeness.

Concept drift is a special case of class evolution [13,18-20]. Number of studies has been dedicated to class evolution. These studies assume the number of classes is changing when class evolution happens. Thus, to identify that of the novel classes accurately in streaming data, they proposed models which are adapted with the distribution of existing classes and disappeared classes. However, the recent class detection techniques dealing with this challenge focus on a single data stream and numeric data, and they have high false detection rates in many scenarios.

3. Proposed Method

In this paper, the author propped a new approach, namely FUND, to address a novel class detection problem on multi evolving data streams. In particular, the proposed FUND approach deals with multi streams datasets by processing group of chunks across multiple streams to build the classification model and declare novel classes from instances of different streams in a dataset. Nevertheless, FUND approaches achieve the classification accuracy by maintaining a set of Class Features of each class label's sub-clusters and introducing a novel policy to overcome the concept-evolution problem. FUND uses a new mechanism for outliers filtering by extending the actual boundary of each sub-cluster taking into account the natural fuzziness of the data because of the fact that boundaries between classes are sometimes not clearly defined due to noise and curse of dimensionality.

Section 3.1 gives a brief overview, and Section 3.2 presents the classification process in detail, and in Section 3.3 devises the new FUND filtering mechanism.

3.1 Overview

This section describes the main characteristics of the proposed model. Our essential goal is to build a classification model from data streams that can detect new concepts as well as minimizing the prediction error. Before explaining the general idea, the author gives informal definitions of multi data streams classification problem. Let $T=\{S_1, S_2, \ldots, S_n\}$ be a set of n streams, where S_i is the i^{th} stream. This work assumes that at each timestamp, data points from individual streams arrive simultaneously and also it is assumed that a data strea, m is continuous and that data arrive in chunks, a data stream S_i can be represented as $D_i^1 = \{x_i^1, \ldots, x_i^s\}$; $D_i^2 = \{x_i^{s+1}, \ldots, x_i^{2s}\}$; $D_i^r = \{x_i^{r-1}, \ldots, x_i^{rs}\}$, where s is the chunk size, D_i^j is the j^{th} data chunk of data stream S_i, D_i^r is the latest data chunk and x_i^p is the p^{th} instance in the stream represented by m mixed numerical and categorical attributes $\{A_{i,1}^p, A_{i,2}^p \ \ldots, x_{i,m}^p\}$. Dividing a large data stream into equal-sized chunks helps to handle the infinite length problem in data streams. We cannot save all the incoming data objects since data streams are infinite and the storing of all data streams is impossible due to limited memory. The problem is to predict the class labels of the latest unlabeled chunk D_i^r . Let c_i and $\widetilde{c_\imath}$, be the actual and predicted class labels of x_i, respectively. If $c_i = \widetilde{c_\imath}$, then the prediction is correct; otherwise it is incorrect.

The proposed model is defined as follows, where the flow of streams is analyzed chunk by chunk. The model handles multiple data streams by using either a centralized or a decentralized approach. The former combines chunks (one chunk per stream) in one global chunk before proceeding to next steps, but the latter one handles multiple data streams chunk by chunk (one chunk per stream) before proceeding to the merging phase to combine the results of all processed chunks. An initial ensemble model, denoted as E, is built with the first int num1 labeled chunks, which are divided into groups, one for each class label then each group is clustered by k-prototypes++ [21] clustering algorithm which combines the two techniques, Kmeans++ [22] and K-prototypes [23]. The former one proposed a way to initialize k-means by choosing starting centers and the latter one presented a combined dissimilarity measure to deal with both numeric and categorical attributes. The author applied the clustering technique within each class label because that makes a significant effect in discovering hidden patterns and relevant features amongst the class label (concept) objects. Hence, each class label is represented by a set of sub-clusters corresponding to different features inside this class. Consequently, this will lead to higher classification performance. Now our sub-clusters are pure because it contains instances from only one class label. A summary of each sub-cluster is saved as a cluster feature (CF). A union of all cluster-features of all classes in a chunk form a new classifier model M_i, and the collection of all classifier's models *{M₁, M₂, …, M_L}* forms the decision boundary of the ensemble model *E*.

Once the initial ensemble model *E* is built, this is used to classify new instances as well as to detect new concepts in recent unlabeled chunks. Each incoming instance in the dataset is first examined by the ensemble model to check whether it is an outlier or not. If it is not an outlier, then the instance is classified as an existing class, and a label is assigned based on the majority vote of classifier models in the ensemble E. The classification process is applied using a combination of two different similarity measures, namely distance and density. Applying various measures during classification rather of using a single one is vital to producing more accurate classification results [24]. If a test instance does not belong to any existing class, then this instance is stored in a temporary buffer and *novelClassFlag* is set to 1. If *novelClassFlag* equals to 1 and there are enough filtered outliers in the buffer that have a strong cohesion among them, then a novel class will be detected. In the process of novel class detection, the outliers in the buffer is analyzed to decide whether they could form a new class or not. That is performed by computing the cohesion between outliers and separation of the outliers from the existing classes. If a novel class is detected, the instances of the class is labelled accordingly.

When instances in a chunk are fully labelled by either the proposed ensemble model *E* or human experts, the chunk is used for training. Training a new classifier model on the latest labelled chunk is important because it maintains the ensemble model and keeps it up-to-date by applying a removal/insertion policy for retaining the best L classifiers in the ensemble model *E*. By removing the classifier model that yields the highest error rate and insert the newly trained classifier model on the latest labelled chunks, the model E is updated continuously so that it represents the most recent concepts in data streams. By doing this, we make sure that we have exactly *L* classifier models in the ensemble at any given point of time, and only a constant amount of memory is needed to store the ensemble which help in addressing the infinite length problem.

3.2 Classification and Outlier Filtering

Building an accurate classifier ensemble and outlier filtering are the two main stages in the proposed classification process. These are detailed in the following subsections.

1) Building the Decision Boundary: Initially, this work examines each test instance in the recent unlabeled chunk using the ensemble of classifier models. It is classified normally using majority voting of the classifier models *Mi* Є *E* if it falls inside the decision boundary of the ensemble *E* or outside the decision boundary but inside the fuzziness space around a sub-cluster and give gain to that sub-cluster. Otherwise, it is detected as an outlier. To keep track of the decision boundary of the classifier ensemble E, labelled chunks are divided into groups, one for each class label, then cluster each group using k-prototypes++ [21] then use Cluster Feature to store summary information of each sub-cluster. Cluster Feature (CF) is a method in stream clustering, which is used to record summary information about a cluster [25].

Definition 1 (Cluster Feature): CF_j is a summary of a sub-cluster ($clust_j$), and includes the centroid ($cent_j$), the number of data points ($|clust_j|$), the radius (rad_j), which is the maximum distance from the centroid to any point in that sub-cluster, the sum of the distance between all points and the sub-cluster centroid ($SumD_j$), and the density (Den_j).

$$CF_j = \{cent_j, |clust_j|, rad_j\,, SumD_j, Den_j\} \tag{1}$$

To compute the distance between two objects, the mixed dissimilarity measure is used. Let us consider two points *x* and *y*, where each of them is represented by *m* attribute values (i.e. $x=[x_1, x_2, \ldots, x_m]$ and $y=[y_1, y_2, \ldots, y_m]$), combined with the first *p* numeric attribute values and the last *m–p* categorical attribute values. The mixed dissimilarity measure between *x* and *y* is defined in Eq. (2).

$$Dis(x,y) = \sum_{j=1}^{p} (x_j - y_j)^2 + \gamma \sum_{j=p+1}^{m} \delta(\, x_j, x_j) \tag{2}$$

where,

$$\delta(x_j, x_j) = \begin{cases} 0 \; if \; (x_j = y_j) \\ 1 \; if \; (x_j \neq y_j) \end{cases}$$

The first part is the Euclidean distance measure, and the second part represents the simple matching dissimilarity measure. The weight γ is used to avoid favoring type of attribute. The work proposed in [26] showed that when γ Є [1.5, 2.5], clustering performance was favorable. In the experiments of this work is set to 2. Each sub-cluster *j* of a class label *i* in the data space has its own density. The density of a sub-cluster is simply considered as the distribution of the data points (this study focuses on normal distribution) into the sub-cluster and computed using Eq. (3).

$$Den_j = \frac{clust_j}{AvgDist_j} \; AvgDist_j = \frac{\sum_{i=1}^{f_j} Dis(p_i, cent_j}{f_j} \tag{3}$$

where *p* is the data point, $cent_j$ is the center and f_j is the number of points in clustj without considering the center point because the distance between the center and itself is always zero (i.e., $f_j = |\, clust_j| - 1$).

The union of all cluster features, which belong to the class label *i* in the chunk being processed, form a Class Feature set of a class label i (CF_{all}^{I}) in short). After computing the sub-clusters summaries, the raw data are discarded. The (CF_{all}^{I}) of all class labels constitute a classification model *M*, and the union of all cluster features of all classifier models form the decision boundary of the ensemble $E = \{M_1, M_2, \ldots, M_L\}$. Then *E* is used for classification and novel class detection.

2) Outlier Filtering using Fuzzy Sub-clusters: Recall that *CF* (Cluster Feature) was used to saving summary information about a sub-cluster, and the radius of a *CF* is defined by the distance from the centroid to the furthest point in the sub-cluster. As the boundaries between classes are sometimes not clearly defined due to noise and curse of dimensionality, thus if a test instance is outside the actual boundary of a sub-cluster, however very close to its surface, it will be an outlier, which increases the false alarm rate (i.e., detecting an existing class as novel). Therefore, the author extends the actual boundary of each sub-cluster by taking into account the natural fuzziness of the data.

Definition 2 (Fuzzy Sub-Cluster): Let $cent_j$ be the centroid and rad_j be the radius of $clust_j$ of class c_i and *q* is a positive small number. *clust*$_j$ is a fuzzy sub-cluster if all its points have the same class label (i.e., a pure cluster) and his actual boundary is extended by *q* to (*q* + *rad*$_j$).

It should be noted that the larger the value of *q*, the greater the confidence with which we can decide whether a test instance *x* belongs or not to *clust*$_j$. However, if *q* is set either too small or too large, then the false alarm rate will increase. Therefore, the author experimentally finds an optimal value of *q*. After extending the sub-clusters boundaries, these may intersect and it is more likely to happen between sub-clusters of the same class label ci. However, sub-clusters of different class labels could not intersect as the instances of different class labels are well separated from each other into well separated sub-clusters.

The classification process is applied by using combination of two different similarity measures, namely distance and density. The classification results will be more accurate if we apply different measures instead of relying on only one measure because it helps in handling the uncertainty problem in data streams. As shown in Fig. 1, after increasing the boundary of each sub-cluster (represented here by the dotted circles), we may consider instance x as a member of the small class A because it is closer. However, taking into consideration another similarity measure, which is the density of the small and big classes, will lead us to the right classification decision. Applying more than one similarity measure in the classification phase gives more accurate prediction by considering not only the distance metric, but also the distribution of the candidate class.

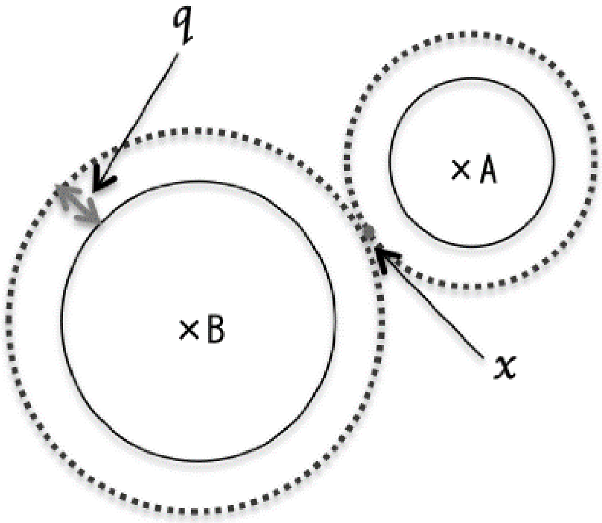


Fig. 1 Illustration of classification using the combination of two similarity measures and Fuzzy sub-clusters.

A test instance *x* is classified using formula (4).

$$Classify(x) = \begin{cases} label\ j, if\ Dis(x, cent_j \leq rad_j) \\ label\ j, if\ rad_j < Dis(x, cent_j) \\ \leq (rad_j + q)\ AND \\ x\ gives\ gain\ to\ class\ i \\ otherwise, outlier \end{cases} \tag{4}$$

In words, a test instance x is classified using a model M_i Є E as follows: the distance between this instance and the centroid of each sub-cluster using Eq. (2) is computed first. Then calculate the density metric when *x* falls in the fuzzy space around each sub-cluster. A test instance may cause either a density gain or loss when joining a sub-cluster. Accordingly, the selected sub-cluster is the one, which attains the most density gain among all sub-clusters when the test instance joined it. The effect on the sub-cluster density if the test instance joined is calculated for each sub-cluster using Eqs. (5-7).

$$Den_j(x) = ExpDen_j(x) - Den_j \tag{5}$$

$$ExpDen_j(x) = \frac{|clust_j|+1}{newAvgDis_j} \tag{6}$$

$$newAvgDis_j(x) = \frac{SumD_j + Dis(x, cent_{j)}}{f_j+1} \tag{7}$$

Let CF_{all}^{B} Є M_i be the Cluster Feature set of class label *B* whose some of his Cluster Features CF_i Є CF_{all}^{B} satisfy the following two conditions: its centroid is nearest from *x* and attains the most density gain among all sub-clusters when the test instance *x* added to it, otherwise *x* will be considering as a local outlier (will be defined shortly). The predicted class label of *x* is the class that this *CF* belongs to, which is *B*. The data point *x* is classified using the ensemble *E* by taking a majority vote among all classifier models M_i Є E.

3.3 FUND: Procedure and Algorithm

This section describes the algorithm of the proposed FUND approach. In particular, FUND handles multiple data streams by combining their chunks, one chunk per stream, in one global chunk before proceeding to following steps. This reduces duplication and leads to more stable classification model. Fig. 2 shows the flow chart of the general process of FUND and algorithm 1 in Fig. 3 shows the corresponding processing details.

- Merging phase: when the initial classification model is built, the procedure starts with latest unlabeled chunks one from each stream in the dataset D_r, which then are combined into one big chunk DATA with respect to the instance's timestamps (Lines 4-5).
- Class-Based grouping phase: when the combined chunk DATA is fully labelled, then it is divided into groups based on class labels, one for each class, that is all instances belong to the same class label will be in one group (Line 15).

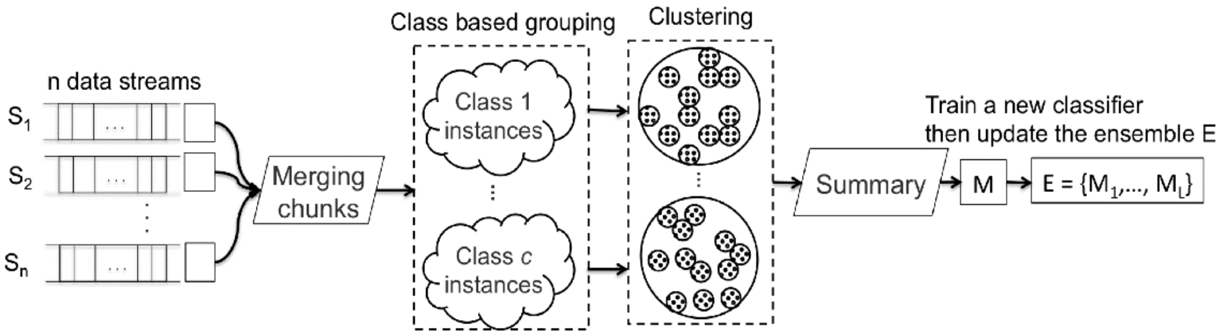


Fig. 2 The FUND approach.

```
Algorithm 1: FUND($D^r, E$)
Input: $D^r = \{D_1^r, D_2^r, \dots \dots D_n^r\}$: /* Set of latest unlabeled
      chunks on form each stream in a dataset */
$E = \{M_1, M_2, \dots \dots M_n\}$: /* Current ensemble of bet
      classifier */
Output: Updated the ensemble model $E$
 4 DATA ← null
 5 foreach chunk $D^r \in D$ do
 6     DATA ← DATA ∪ $\{D^r\}$
 7 foreach instance $x \in DATA$ do
 8     DATA ← DATA ∪ $\{D^r\}$
 9     if label == null then
           /* x is an outlier to all classifier models
           $M_i \in E$ */
10         $Fbuffer$ ← $Fbuffer \in E$
11         novelClassFlag = 1
12     else
13         assign (label) to x
14 if novelClassFlag == 1 then
15 DetectNovelClass(Fbuffer)
           /* Assuming that DATA is now fully labelled */
16 ClassSets ← GroupClasses(DATA)
17 foreach classSet i = 0 to c do
18     $SC_i$ ← cluster($classSet_i$)
19     foreach $clast_j \in SC_i$ do
20         $CF_j$ ← $GenerateClusterFeature(clast_j)$
21         $CF_{all}^i$ ← $\{CF_j\}$
22     CCF ← ∪ $\{CF_{all}^i\}$
23  $M^*$ ← $TrainNewClassifier(CCF)$
           /* Train on the most recent labelled chunks */
24  $E$ ← $UpdateTheEnsemble(E, M^*, DATA)$
25 return $E$
```

Fig. 3 FUND Algorithm.

- Clustering phase: after that, total k sub-clusters per group is generated from the previous step (Line 17) using k-prototypes++ clustering algorithm [21]. Now the sub-clusters SC_i are pure, as they contain instances from one class only.
- Summarizing phase: a cluster feature *CF* is created for each sub-cluster generated in the previous step (Lines 18- 20). CF_{all}^i is the union of all cluster features of all sub-clusters that belong to a class label i. Then a classifier model *M`* is created by combining all the $CF_{all}^i$ all of all class labels in the DATA (Line 22). Finally, the model *M`* is used to update the classifier ensemble *E* (Line 23).

4. Evaluation

FUND is implemented in Eclipse Java EE IDE version Juno Service Release 2. The carried experiments were performed on a 64-bit Windows-based system with an Intel core (i7), 3.40 GHz processor machine with 8 Gbytes of RAM. The code for C4.5 is adopted from Weka3. Weka 3 is also used as Library, open source data mining software written in Java [27]. Weka3 is a collection of machine learning algorithms for data mining tasks, which contains tools for data preprocessing, classification, regression, clustering, association rules, and visualization. These learning algorithms can be either applied directly to a dataset or called from author's own coding.

FUND is evaluated on a number of real and synthetic datasets. This section describes the used datasets, experimental environment and parameters setting then present the evaluation results.

4.1 Datasets

Table 1 describes the various datasets used in the experiments, which include the datasets from UCI machine learning repository [28], ADFA-LD [29-32] and synthetic datasets. These datasets represent a wide range of domains and data characteristics including: **KDDCup 99 network intrusion detection** dataset, **EMG physical action** dataset, **EMG physical action** dataset, and **Synthetic multi stream dataset** (SynMS).

Table 1 Datasets summarized characteristics.

Dataset	# stream	# instance	# attribute	# classes
KDDcup 99	1	490,000	42	23
ADFA-LD	1	5206	180	60
Auslan	9	6650	15	95
EMG	4	10000	8	20
SynMS40	4	1,000,000	10	40

4.2 The Baseline Techniques

FUND is compared against two well-known baseline techniques in novel class detection, namely MineClass (Mining novel Classes in data streams) [6] and DTNC (Detecting Novel Classes in Data Streams) [21]. These two approaches use C4.5 and VFDTc respectively as base learners. Because the categorical attributes cannot be processed by k-means, the k-means and k-modes are combined for MineClass. Both MineClass and DTNC were extended to be applicable to deal with multi streaming data in FUND so we could compare it with the proposed model. This work applied the both versions on the four real datasets.

4.3 Parameters Settings

The parameter settings are as follows: *K* (number of cluster feature per chunk) =20, *S* (window size) is 400 for ADFA-LD dataset and 4000 for KDD99 dataset and synthetic dataset *L* (ensemble size) =6 and N (minimum number of Global outliers to declare a novel class) = 50. (Used to avoid favoring type of attribute in the mixed dissimilarity measure) =2. *q* (Small positive length added to the radius of a sub-cluster to extend its boundary) =0.4. These parameters are chosen either according to the default values used in [6] or by trial and error to get an overall satisfactory performance. In particular, to evaluate the process of each novel classis's detection in data streams approaches (*M*=5 Є *N*=10) cross-validation strategy is used. The 10-fold cross-validation is repeated *M*=5 times, with the order of the instances of the dataset being randomized each time. This is because many of the algorithms are biased by the data order that is certain orderings dramatically improve or degrade performance.

The procedure for the experiments is shown in Fig. 4. For each novel class detection approach, this work evaluated its corresponding runtime, detection rate, falsely identified as existing class rate and falsely identified as novel class rate are obtained for each dataset. The average detection rate is defined in Eq. (8).

$$ERR = \frac{(E_p + E_n + E_e) * 100}{N} \qquad (8)$$

E_n denotes the total novel class instances misclassified as existing class, E_e denotes the total existing class instances misclassified as novel class, E_p the total existing class instances misclassified as another existing class, and *N* denotes the total instances of the traffic data. The percentage of the total novel class instances misclassified is also calculated as existing class as in Eq. (9).

$$M_{new} = \frac{E_n * 100}{V} \qquad (9)$$

Where *V* denotes total number of novel class instances. The percent of existing class instances misclassified as novel class has been calculated as Eq. (10).

$$F_{new} = \frac{E_p * 100}{N - V} \qquad (10)$$

The percent of existing class instances misclassified as another existing class (other than E_p) has been calculated as in Eq. (11).

$$F_e = \frac{E_e * 100}{N - V} \quad (11)$$

Algorithm 2: Experimental Procedure

```
Input:
M = 10;
Tech = {$FUND, DNTC, MineClass$};
DATA = {$KDD, EMG, \dots, D_n$};
Output:
PerfoMetrics = {$ERR, Mnew, Fnew, Runtime$};
foreach $Tech_i \in (1, ST)$ do
    foreach $D_i \in DATA$ do
        foreach times $\in (1, M)$ do
            randomise instance-order for $D_i$;
            generate N bins from the randomized $D_i$;
            foreach fold $\in (1, N)$ do
                $Test_{Data}$ = bins[fold];
                $Train_{Data}$ = $Trans_{Data}$ - $Test_{Data}$;
                $Train'_{Data}$ = selectSubset from $Train_{Data}$;
                $Test'_{Data}$ = selectSubset from $Test_{Data}$;
                foreach $Tech_i \in Tech$ do
                    $Tech_i$ = learner($Train'_{Data}$);
                    $Results_i$ = apply $Tech_i$ to ($Test'_{Data}$);
            OverallResult = OverallResult ∪ $Results_i$;
        PerfoMetrics = average(OverallResult);
```

Fig. 4 Experimental Procedure.

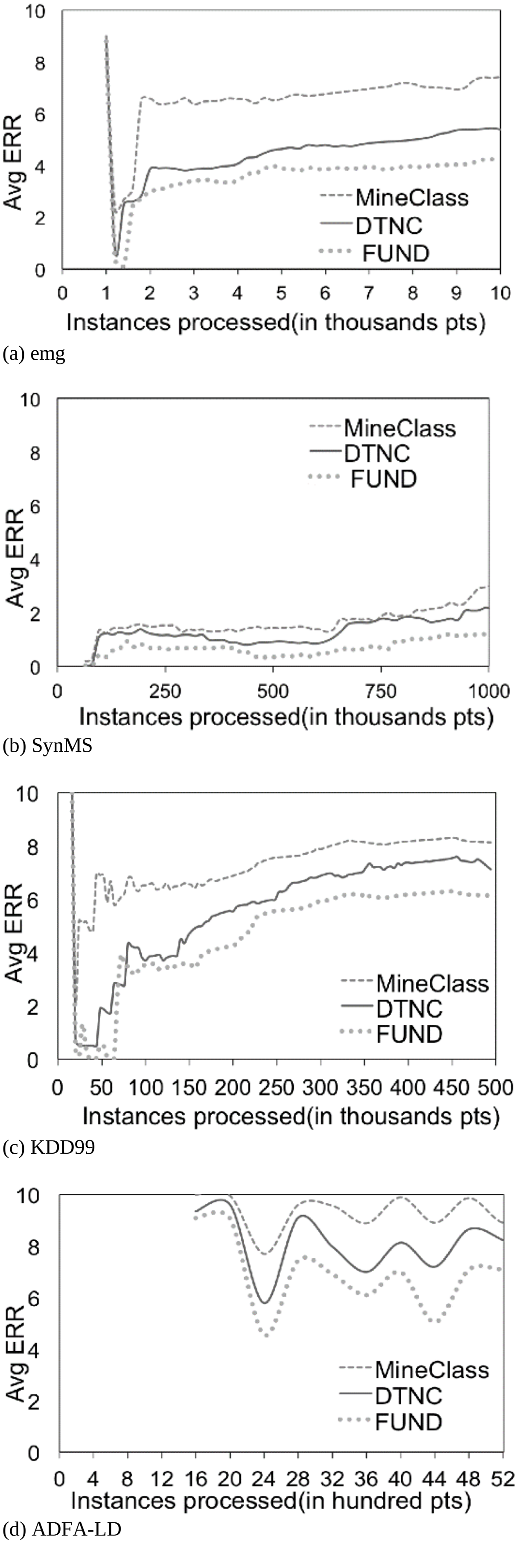
4.4 Experimental Results

There are different ways to evaluate the performance of data stream mining classifiers. The performance can simply be measured by counting the proportion of correctly classified instances in an unseen test dataset. Fig. 5 summarizes the results of the average detection error rate for the FUND approach and baseline techniques.

In particular, this work builds the initial models in each method with the first int num = 3 chunks. From the chunkNo=int num + 1 onward, the author first evaluates the performances of each method on that chunk, then use that chunk to update the ensemble. The performance metrics for each chunk for each method are saved and averaged for producing the summary result. Fig. 5 shows the ERR for each model using FUND and baseline approach throughout the streams in different multi-streams datasets. The Y values show the average ERR of each model from the beginning of the data streams to instance number 6000. It can be seen that the proposed approach scored the lowest average detection rate compared to the baseline techniques on all datasets. Table 2 summarizes the error metrics for each of the techniques in all dataset for each approach. The columns headed by ERR, Mnew and Fnew report the average of the corresponding metric on an entire dataset. The error rate values for each classifier model shown in the above three columns are produced using Eqs. (7-9).

The results indicate that the proposed approach shows great robustness to learn and classify new class labels (new concepts) in each dataset. Also, the overall ERR rate of FUND is much lower than other methods in all datasets. For example, the ERR of FUND, DTNC, and MineClass on KDD99 dataset are 4.71%, 7.68% and 8.53%, respectively. Also, on KDD99 dataset, Fund misses only 5.41% of novel class instances, whereas DTNC, and MineClass misses 8.43% and 9.42% instances, respectively. Likewise, Fund misclassified existing class instances as novel class only 1.83%, whereas DTNC, and
MineClass misses 2.12% and 3.57% instances, respectively.

The ERR of FUND, DTNC, and MineClass on ADFA-LD are following: 6.97%, 8.04%, and 9.30 %, respectively. Also, on ADFA-LD dataset, Fund misses only 8.11% of novel class instances, whereas DTNC, and MineClass misses 8.83% and 9.87% instances, respectively. Likewise, Fund misclassified existing class instances as novel class only 2.71%, whereas DTNC, and MineClass misses 4.01% and 4.48% instances, respectively.



(a) emg

(b) SynMS

(c) KDD99

(d) ADFA-LD

Fig. 5 Error comparison on multi-streams datasets using FUND.

Table 2 Performance comparison.

Dataset	ERR			Mnew			Fnew		
	FUND	MineClass	DNTC	FUND	MineClass	DNTC	FUND	MineClass	DNTC
KDD99	4.71	8.53	7.68	5.41	9.42	8.43	1.83	3.57	2.12
ADFA-LD	6.97	9.30	8.04	8.11	9.87	8.83	2.71	4.48	4.01
EMG	3.21	6.59	4.53	3.95	6.81	5.14	0.85	1.9	1.15
SynMS40	0.74	1.82	1.51	0.0	0.0	0.0	0.5	1.2	0.9

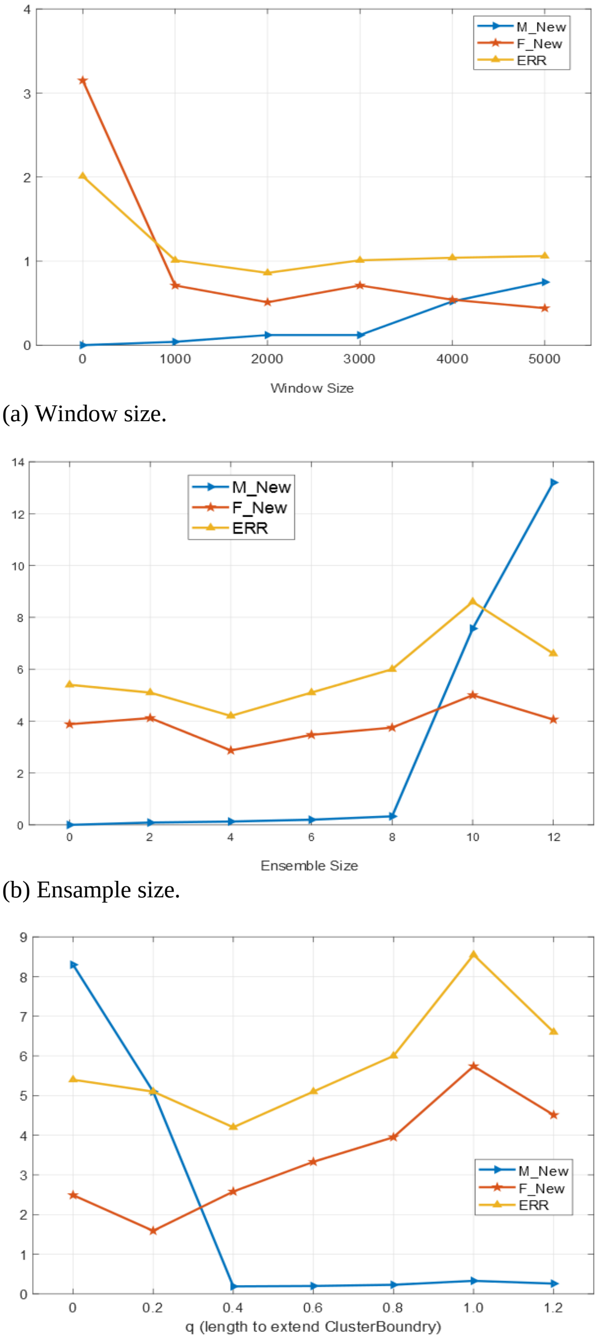
The ERR of FUND, DTNC, and MineClass on EMG are 3.21%, 4.53% and 6.59%, respectively. Also, on EMG dataset, Fund misses only 3.95% of novel class instances, whereas DTNC, and MineClass misses 5.14% and 6.81% instances, respectively. Likewise, Fund misclassified existing class instances as novel class only 0.85%, whereas DTNC, and MineClass misses 1.15% and 1.9% instances, respectively.

The ERR of FUND, DTNC, and MineClass on SynMS40 are 0.74%, 1.51% and 1.82%, respectively. Also, on SynMS40, Fund, DTNC and MineClass have correctly classified all novel class instances with Zero misclassification, this is due to the noiseless presented in the SynMS40 dataset. On the other hand, Fund misclassified existing class instances as novel class only 0.5%, whereas DTNC, and MineClass misses 0.9% and 1.2% instances, respectively.

In general, FUND outperforms the baseline techniques in the average of error rate and novel class detection. This is due to the ability of FUND in filtering outliers during the process of novel class detection. Recall that FUND allows the natural fuzziness of dataset by extending the boundary of each sub-cluster and uses the density metric in addition to the distance metric to identify outliers. Nevertheless, the baseline techniques are more sensitive to concept-drift than FUND. Thus, due to concept-drift some existing class instances are misclassified as novel class. Similar characteristics are observed for other data sets. Also, it can be seen from Fig. 6, the waiting time to build the next classifier is increased as window size increasing. This result in high error rates as an ensemble model is not frequently updated, and remains outdated for longer time.

Figure 6(b) shows the effect of ensemble size (M) on error rates. It can be seen that as the number of ensemble size (M) is increased, the value of the ERR and also kept decreasing. This is due to the reduction in error variance. On the other hand, it can be observed that there is increasing in Mnew rate as the number of ensemble size (M) is increased. This is due to the fact that a larger ensemble means more restriction on identification of the arrival of novel classes. Thus, to choose the default parameters of FUND, we need to be assured that both the overall error (ERR) and Mnew rates are as low as possible.

Figure 6(c) shows the effect of q (Length to Extend Cluster Boundary) on the overall error (ERR), Mnew rates and Fnew rates. It can be seen from Fig. 6(c) that the influence of q value is similar to the effects of the number of ensemble size (M). In particular, the outliers are detected accurately as the q value increased, this results in low Mnew rate. A general observation also indicates a large value of q can enhance the confidence in emerging of novel class. However, also it can be seen from Fig. 6(c) that too large value of q results in high M_{new} and ERR rates. To address this issue, the author found the best setting of q value can be between 0.2-0.8.



(a) Window size.

(b) Ensample size.

(c) q length to extend cluster boundary.

Fig. 6 The performance of FUND under different parameters setting.

Table 3 Comparison of Running Time.

Dataset	Time (Sec.)			Points/Sec.			Speed Gain	
	FUND	DNTC	MineClass	FUND	DNTC	MineClass	FUND over DNTC	FUND over MineClass
KDD99	576.03	540.54	950.01	934	975	4190	0.94	1.65
ADFA-LD	4.06	4.42	6.47	51	70	59	1.09	1.59
EMG	7.26	8.49	12.43	1438	1270	870	1.17	1.71
SynMS40	1070.42	1147.73	1385.05	1009	953	816	1.07	1.29

The running time of the novel class detection is an important criterion in streaming data. Table 3 shows performance of the FUND and the other two baseline techniques on the four datasets in terms of the running time. In particular, the author has evaluated each approach with respect to the average training and testing times, the average processed instances per second (instances/sec) and the ratio of the speed of FUND approach to the other two baseline techniques (speed gain). It can be seen that from Table 3, the FUND approach outperforms the baseline techniques on all datasets except for KDD dataset. For example, the FUND is 1.71, 1.65, 1.59 and 1.29 times faster than MineClass on EMG, KDD, SynMS40 and ADFA-LD datasets, respectively. Also, the FUND is 1.17, 1.09 and 1.07 faster than DNTC on EMG, ADFA-LD and SynMS40 datasets, respectively. However, the author notes that the performance time of the FUND approach is 1.06 time slower than DNTC on KDD dataset. This is because FUND approach creates new cluster as new class emerge, and then proceeds to validate the created clusters. Thus, the processing time of the FUND becomes slower than DNTC as novel classes observed frequently in KDD dataset. From Fig. 6, this work examined the effects of parameter settings on KDD dataset on the performance of FUND. This includes the effect of window size on ERR, Fnew, and Mnew rates for default values of other parameters. We can notice that Fnew rates decrease up-to a certain point then increases. This is due to increment of the window size, which mean more training data during the training phase. Such parameters have similar effects on other data sets and FUND approach.

Figure 7 shows the scalability of FUND approach and the other two baseline techniques with varying number of instances in the dataset. Note with ever-looming page limitations, the author uses only KDD dataset. The size of the KDD dataset is 490K flows, the author plots them on the graph with flows varying from 50K to 490K. Fig. 7 indicates that the FUND approach has the better scalability performance in comparison to the MineClass, whereas the scalability of FUND may seem like the same as DNTC. It is clear from the trend that DNTC and FUND almost scale linearly with respect to the number of instances. However, to improve the efficiency of FUND approach future work using 1) parallel computing or 2) GPU environment.

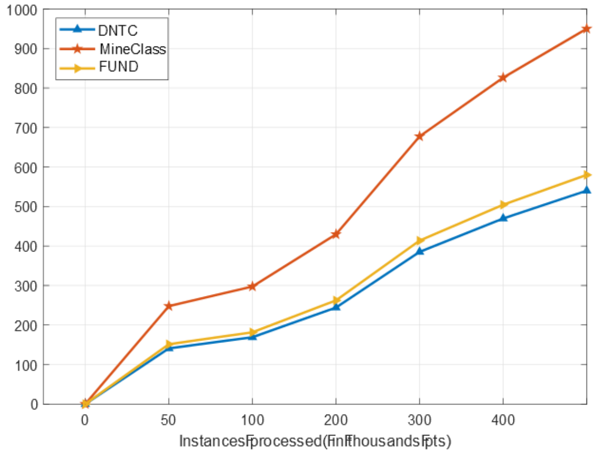


Fig. 7 The scalability of FUND approach and the other two baseline techniques.

5. Conclusion

This paper has introduced an ensemble model for classification and novel class detection in multi streams dataset. The main objective of this research is to minimize the total misclassification error (ERR) in classification and novel class detection tasks. This paper develops a new mechanism for novel class detection (the critical aspect in this context) by maintaining a set of representatives of each class label, set of clusters features, as well as, extended the actual boundary of the sub-clusters to allow the fuzziness nature of the data streams and using different similarity measure, density and distance, to overcome the uncertainty problem. As a result, novel class instances in data streams can be automatically detected using the proposed model FUND. This work addresses challenging issues in multi data streams classifications such as infinite length, concept-drift, and concept-evolution and data uncertainty. The proposed ensemble model generally continuously updates itself with newly arrived instances chunks so that it represents the most recent concepts in data streams. The author tested the performance of the ensemble model on five real and synthetic datasets. The experimental results proved that this ensemble classifier efficiently detects the arrival of novel class instances and also greatly improves the classification accuracy rates under different circumstances.

Acknowledgement

This research is a part of a project entitled *"SMART-PHS: A Smart Card for Automated Patient Health Service: Medical Documents Analysis, Recognition and Data-storage"*. This project was funded by the *Deanship of Scientific Research, Albaha University, KSA (Grant No. 1439/33)*. The assistance of the deanship is gratefully acknowledged.

References

[1] Al-Kaff A, García F, Martín D, et al. Obstacle detection and avoidance system based on monocular camera and size expansion algorithm for UAVs. Sensors. 2017;17(5):1061.

[2] Xie M, Trassoudaine L, Alizon J, et al. Road obstacle detection and tracking by an active and intelligent sensing strategy. Machine Vision and Applications. 1994;7(3):165-177.

[3] Ngai EW, Hu Y, Wong YH, et al. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. Decision support systems. 2011;50(3):559-569.

[4] Zhang D, Lu G, editors. A comparative study of Fourier descriptors for shape representation and retrieval. Proc. 5th Asian Conference on Computer Vision; 2002: Citeseer.

[5] Russell SJ, Norvig P. Artificial Intelligence A Modern Approach; PearsonEducation. Artificial Intelligence: A Modern Approach: Pearson Education. 2003.

[6] Jain P. Wind energy engineering. New York: McGraw-Hill; 2011.

[7] Shah VR, Maru SV, Jhaveri RH. An obstacle detection scheme for vehicles in an intelligent transportation system. International Journal of Computer Network and Information Security. 2016;8(10):23.

[8] Rodrigues PP, Gama J, Pedroso J. Hierarchical clustering of time-series data streams. IEEE transactions on knowledge and data engineering. 2008;20(5):615-627.

[9] Yeh M-Y, Dai B-R, Chen M-S. Clustering over multiple evolving streams by events and correlations. IEEE transactions on knowledge and data engineering. 2007;19(10):1349-1362.

[10] Shinzato PY, Wolf DF, Stiller C, editors. Road terrain detection: Avoiding common obstacle detection assumptions using sensor fusion. 2014 IEEE Intelligent Vehicles Symposium Proceedings; 2014: IEEE.

[11] Nasraoui O, Uribe CC, Coronel CR, et al., editors. Tecno-streams: Tracking evolving clusters in noisy data streams with a scalable immune system learning model. Third IEEE International Conference on Data Mining; 2003: IEEE.

[12] Nasraoui O, Cardona C, Rojas C, et al., editors. Mining evolving user profiles in noisy web clickstream data with a scalable immune system clustering algorithm. Proc. of WebKDD; 2003.

[13] Alsharif N. Connectivity-Aware Routing in Vehicular Ad Hoc Networks. 2017.

[14] Xie D, Xu Y, Wang R. Obstacle detection and tracking

method for autonomous vehicle based on three-dimensional LiDAR. International Journal of Advanced Robotic Systems. 2019;16(2):1729881419831587.

[15] Dai B-R, Huang J-W, Yeh M-Y, et al., editors. Clustering on demand for multiple data streams. Fourth IEEE International Conference on Data Mining (ICDM'04); 2004: IEEE.

[16] Papadimitriou S, Sun J, Faloutsos C. Streaming pattern discovery in multiple time-series. 2005.

[17] Sun S, Huang R, editors. An adaptive k-nearest neighbor algorithm. 2010 seventh international conference on fuzzy systems and knowledge discovery; 2010: IEEE.

[18] Wang S, Minku LL, Yao X. A systematic study of online class imbalance learning with concept drift. IEEE transactions on neural networks and learning systems. 2018;29(10):4802-4821.

[19] Sun Y, Tang K, Zhu Z, et al. Concept drift adaptation by exploiting historical knowledge. IEEE transactions on neural networks and learning systems. 2018;29(10):4822-4832.

[20] ZareMoodi P, Siahroudi SK, Beigy H. Concept-evolution detection in non-stationary data streams: a fuzzy clustering approach. Knowledge and Information Systems. 2019;60(3):1329-1352.

[21] Miao Y, Qiu L, Chen H, et al., editors. Novel class detection within classification for data streams. International Symposium on Neural Networks; 2013: Springer.

[22] Wang H, Lou X, Cai Y, et al. A 64-line Lidar-based road obstacle sensing algorithm for intelligent vehicles. Scientific Programming. 2018;2018.

[23] Discant A, Rogozan A, Rusu C, et al., editors. Sensors for obstacle detection-a survey. 2007 30th International Spring Seminar on Electronics Technology (ISSE); 2007: IEEE.

[24] Abdallah ZS, Gaber MM, editors. Kb-cb-n classification: Towards unsupervised approach for supervised learning. 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM); 2011: IEEE.

[25] Zhang T, Ramakrishnan R, Livny M. BIRCH: an efficient data clustering method for very large databases. ACM sigmod record. 1996;25(2):103-114.

[26] Wang J, Zhu Y. Research on the Weighting Exponent in the Fuzzy K-Prototypes Algorithm. Journal of Computer Applications. 2005;25(2):348-351.

[27] Sivaraman S, Trivedi MM. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. IEEE transactions on intelligent transportation systems. 2013;14(4):1773-1795.

[28] Bendjaballah M, Graovac S, Boulahlib MA. A classification of on-road obstacles according to their relative velocities. EURASIP journal on image and video processing. 2016;2016(1):41.

[29] Hong T-H, Legowik S, Nashman M. Obstacle detection and mapping system. US Department of Commerce, Technology Administration, National Institute of …; 1998.

[30] Umakirthika D, Pushparani P, Rajkumar MV. Internet of Things in Vehicle Safety–Obstacle Detection and Alert System. International Journal of Engineering and Computer Science. 2018;7(02):23540-23551.

[31] Yi X, Song G, Derong T, et al. Fast road obstacle detection method based on maximally stable extremal regions. International Journal of Advanced Robotic Systems. 2018;15(1):1729881418759118.

[32] Xie M, Hu J, editors. Evaluating host-based anomaly detection systems: A preliminary analysis of adfa-ld. 2013 6th International Congress on Image and Signal Processing (CISP); 2013: IEEE.

ALBAHA UNIVERSITY JOURNAL OF BASIC AND APPLIED SCIENCES

Dar Al Manar for Printing +966 17 7223212