Received April 30, 2021, accepted May 10, 2021, date of publication May 21, 2021, date of current version June 2, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3082755

Towards an Unsupervised Feature Selection Method for Effective Dynamic Features

NAIF ALMUSALLAM[1], ZAHIR TARI[2], JEFFREY CHAN[2], ADIL FAHAD[3], ABDULATIF ALABDULATIF[4], (Member, IEEE), AND MOHAMMED AL-NAEEM[5], (Member, IEEE)

[1]School of Computer Science, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh 13318, Saudi Arabia
[2]School of Computer Science and Information Technology, RMIT University, Melbourne, VIC 3001, Australia
[3]School of Computer Science and Information Technology (CS&IT), Albaha University, Al Bahah 65731, Saudi Arabia
[4]Computer Science Department, Qassim University, Buraydah 51452, Saudi Arabia
[5]Department of Computer Networks and Communications, College of Computer Science and Information Technology, King Faisal University, Al Ahsa 31982, Saudi Arabia

Corresponding author: Naif Almusallam (nyalmusallam@imamu.edu.sa)

Prof Zahir Tari would like to thank the Australian Department of Industry, Innovation and Science for supporting this work under the Australia-China Science and Research Fund grant on the topic of "Resource Allocation for High-Volume Streaming Data in Data Centres".

ABSTRACT Dynamic features applications present new obstacles for the selection of streaming features. The dynamic features applications have various characteristics: a) features are processed sequentially while the number of instances is fixed; and b) the feature space does not exist in advance. For example, in a text classification task for spam detection, new features (e.g. words) are dynamically generated and therefore need to be mined to filter out the spams rather than waiting for all features to be collected in order to do so. Traditional feature selection methods, which are not designed for streaming features applications, cannot be used in such an environment, as they require the full feature space in advance in order to statistically determine the representative features. Existing methods that address feature selection in dynamic features applications require the class labels in order to select the representative features. However, most of the real-life data is unlabeled and it is costly to apply manual labeling. In this paper, an efficient unsupervised features selection method is proposed for streaming features applications where the number of features increases while the number of instances remains fixed. In particular, unsupervised Feature Selection for Dynamic Features (UFSSF) is developed to determine the representative streaming features without requiring prior knowledge about data class labels or representative features. The UFSSF extends the k-mean clustering to cumulatively determine whether the newly-arrived feature can be selected as a representative streaming feature, or discarded. Experimental results show significant accuracy results and efficient execution time compared to those of other benchmark methods.

INDEX TERMS Feature selection, streaming features, unsupervised learning.

I. INTRODUCTION

The high-dimensional data decreases the performance of machine learning algorithms in dynamic features applications. Non-representative features: 1) burden the run time of machine learning algorithms with extra processing time; and 2) decrease the prediction accuracy of trained algorithms. Therefore, feature selection methods have been applied to identify the representative features of data streams to eliminate obstacles related to data dimensionality. The methods for feature selection reported in the literature (see Section II) take into consideration only the *static* features to determine the set of representative features effectively. Therefore, the current feature selection methods cannot be applied effectively for streaming features applications when features are arriving sequentially.

There are two categories of data streams: *streaming data* and *streaming features* [1]. In the category of streaming data, there is a dynamic number of instances and there is a fixed number of features. In this paper, the focus is on the streaming features category where there is a fixed number of instances and a dynamic number of features. These features are processed one-by-one upon their arrival. One real-world application that can be categorized as streaming features

The associate editor coordinating the review of this manuscript and approving it for publication was Zhigao Zheng.

application is the spam emails classification where every single email is an instance that contains features/words [2]. Twitter is another example where features, in this case slang words, are created in a dynamic manner and it is processed cumulatively upon the arrival of features rather than waiting for a complete set of features to be processed, which is required by traditional methods of feature selection. Given the overwhelming number of generated features, it is infeasible to wait for the aggregation of all features before being able to start the feature selection process as the number of streaming features is not known in advance, and new features are generated over time. In streaming features applications, the selection process consists of two stages [3]: 1) in the first stage, the newly-arrived feature is inspected to determine whether or not it is a representative feature based on a pre-defined criterion (e.g. correlation of the features); 2) in the second stage, the established set of features is evaluated to determine their representativeness for the complete dataset. This process ensures that only the representative features are selected and the non-representative features are discarded.

Traditional unsupervised feature selection methods [4]–[6] are inefficient for streaming features applications because they require the complete feature space to be identified in advance, which is impractical for streaming features. Additionally, we would need to store large windows of the data streams, which is infeasible due to the tremendous size produced by the streams. Traditional feature selection methods have greater computational complexity, which makes them inappropriate for high-dimensional streaming features as they require fast and real-time processing. Moreover, in streaming features applications, algorithms should read the data only once due to the finite amount of storage space, and then non-representative features should be removed to allow storage. Finally, traditional feature selection methods *static* cannot dynamically update their selected predictive features, thereby negatively affecting the process of feature selection. Therefore, it is necessary to consider the specific characteristics of streaming features when designing a feature selection method.

Several feature selection methods [7]–[9] have been developed for streaming features applications (see Section II for more details about these methods). However, these methods require data class labels to identify the representative features. To the best of our knowledge, the only unsupervised method for the selection of dynamic features was proposed by Li *et al.* [3]. However, it has some limitations as it requires link information to be identified (i.e. at the relationship of friendship between users in Twitter application). Moreover, it is assumed that the link information is stable, which certainly is not so in cases where it could dynamically change.

The proposed approach is an extension of our previously published work [10]. This approach extends the k-mean clustering to cumulatively determine whether a newly-arrived feature can be selected as a representative streaming feature. It uses any of linearly dependent similarity measures (see Section III) as single distance parameter for UFSSF to cumulatively identify the dependency of the arrived streaming features to determine whether or not the feature can be added to the representative features. The features are processed in a real-time one-by-one upon their arrival. Linearly-dependent measures are applied because they are insensitive to the order and distribution of features. Furthermore, the developed model cumulatively updates the cluster centroids because, due to the dynamic nature of the data stream, one feature might be representative only for a finite period of time. The problem of the dynamic change in data distribution over time is called concept drift [11]. The mean of each cluster is updated after adding a feature to it and the similarity of the newly-arrived feature is compared with the current cluster's representative feature.

Extensive experiments have been conducted. Two well-known unsupervised feature selection methods, SPEC [12] and the method proposed in [13], are used for evaluation and comparison with the proposed approach in terms of their performance regarding prediction accuracy and execution time. The conducted experiments fall into two parts. First, the developed model built the streaming features environment and ensured the following: a) there are no identified features in advance; and b) the simulation process takes place in a real-time environment. Second, to investigate the consistency of results, it is assumed that the entire feature space is available. To do so, the number of selected features is varied from the entire features stream, by selecting 10, 15, 30, from the whole stream. In the experiments, the proposed method outperformed the other selected benchmark methods in terms of both run time and classification accuracy. In summary, our contributions are:

- Development of an unsupervised dynamic/streaming feature selection method for applications by working without the need to data class labels, features size or information about the link between the users, e.g., in Twitter.
- Adapting the k-mean algorithm to work in dynamic features applications incrementally where features are not known in advance, and considering data stream properties such as one pass over data.
- Dynamically assigning(adding) and updating (replacing) the set of selected representative features.
- Empirical experiments have been conducted with extra datasets and evaluation metrics.

This paper is organized as following. The next section discusses the related work followed by the similarity measures applied in the experiments. Section IV provides full details of the UFSSF method followed by an algorithm. In Sections V and VI, the experimental setup and the experimental results are described in detail. We conclude with our findings in Section VII.

A. PRELIMINARIES AND PROBLEM STATEMENT

We formally introduce the problem of *unsupervised feature selection for streaming features*. We assume a stream of feature vectors, $F = \{f_1, f_2, \ldots\}$ (possibly infinite in their number and have no set of order), where each f_i is a vector of the feature values for n instances. Let F_t be the features observed up to time t. E.g., if F represents a stream of tweets from Twitter, then the features are individual words, and each post is an instance, and a feature vector would represent the frequency with which that word (feature) appears in each of the tweets. F_t is the feature/word vectors observed up to time t. Each feature vector in F arrives one-by-one; there are no restrictions on the order in which they arrive, and they do not have class labels.

We wish to maintain a representative set of features that approximates the feature stream seen so far.

As the feature stream is potentially infinite in length, and the relevant set of features could change with time due to concept drift, it is not feasible to wait for all the features to be collected.

Let $R_t = \{f_1^R, f_2^R, \ldots, f_k^R\}, f_i^R \subset F_t$, $1 \leq i \leq k$, denote the set of k representative features at time t. k can range from 1 to k_{max}, the maximum number of representative features.

As features arrive one-by-one, the problem of unsupervised feature selection for streaming features involves maintaining a set of representative features R_t, such that R_t approximates the features F_t observed up to time t. Each representative feature f_j^R of R_t represents a subset/cluster of features in F_t. For each incoming feature f_i, the problem we are addressing in this paper is related to two issues:

1) How to determine which existing representative feature and associated feature cluster f_i must be assigned?
2) How to update the feature cluster and representative feature?

II. RELATED WORK

Extensive research has been conducted in the field of feature selection in attempts to reduce the high-dimensionality of data. The generated features are evaluated for their representativeness based on a specific evaluation criterion. Broadly speaking, there are two main types of feature selection approaches: *filter* and *wrapper*. The distinction between these two is based on whether or not a data mining algorithm is used for the evaluation of the selected features [14]. The filter approach as in methods [4], [5], [15] depends on the overall data characteristics to assess the quality of candidate features without applying data mining algorithms. This includes distance, correlation, consistency and theoretic information measures [9]. Filter-based techniques are considered to have faster processing time compared to wrapper-based techniques because they do not require data mining algorithms to assess the quality of the selected features. Hence, filter-based techniques are more efficient for processing a data stream [16]. Conversely, the wrapper-based techniques, as in methods [6], [17], [18], requires specific data mining algorithms, such as clustering algorithms, during the evaluation of the generated candidate features [19]. Although the wrapper-based techniques can ensure better quality of candidate features compared to the filter-based approach, this comes with costly computational overheads [20]. The aforementioned work assumes that the complete feature space is static as the complete set of features is required in advance. However, in streaming features applications, features arrive sequentially and they do not pre-exist. Therefore, these traditional feature selection methods cannot work efficiently and accurately for streaming features applications [21].

Few studies have been conducted on streaming features applications in the feature selection field. Hsu *et al.* [7] proposed a method which determines the subset of streaming features that have been collected for use as one of the integral parts of the learning process. It gradually builds up the subset of selected features along with the training of the predictive model. The proposed method can effectively adapt to the dynamic nature of the streams due to the its incremental learning approach. *Alpha-investing* [8] assesses the representativeness of each newly-arrived feature based on a per-defined dynamic error reduction threshold (called p-value). More precisely, the pre-defined p-value is presented to identify the acceptability of a new feature for inclusion in the selected features set. Although *Alpha-investing* is able to evaluate the undefined size of streaming features, no selected features can be removed from the set. In [9], *an online Streaming Feature Selection(OSFS)* is developed to determine the representative features in real time while disregarding redundant features. The OSFS measures the features dependency of newly-arrived features based on their class labels and then the selected features are added if they meets a specific criterion, upon which it becomes the best candidate feature. Redundant features are removed dynamically by the OSFS using the Markov Blanket. *OFS-Destiny* [22] does not require any parameters in advance as it relies on neighborhood relationships. It decides feature redundancy based on fuzzy equal constraint. *BP-SFS* [2] selects features by deploying Bayesian regularization in a penalized model. It estimates the regularization parameter based on the coefficients of current state. The characteristics of these methods are summarised in Table 1.

The abovementioned methods have one essential requirement: the data class label is needed to identify the set of representative features. However, most real-life applications have un-labeled data, and data labeling can incur a performance overhead. To the best of our knowledge, there is only one method in [3] that does not require data class labels for the feature selection process and is convenient for applications that require streaming features selection. However, the proposed method has the drawback that it requires link information to be established initially. Moreover, it is assumed that there is stable link information which is certainly not the case where it can dynamically change.

III. SIMILARITY MEASURES

In this section, we present the similarity measures that are applied to determine the dependency of streaming features in order: 1) assign a new streaming feature to its corresponding relevant cluster; 2) identify if the feature is included to the set of representative features or not; and 3) update the set of the

TABLE 1. Characteristics of existing feature selection methods for streaming features.

Method/characteristic	Streaming features	Unsupervised (i.e no class labels)	Constrains (if applicable)
Grafting	Yes	No	NA
Alpha-investing	Yes	No	NA
OSFS	Yes	No	NA
OFS-Density	Yes	No	NA
BP-SFS	Yes	No	NA
Li et al.	Yes	Yes	Link information must be identified
UFSSF (proposed)	**Yes**	**Yes**	**NA**

selected features by discarding the features that are no longer representative dynamically. Linearly-dependent measures are used due to their effectiveness in the feature selection process where they are sensitive to the location or to the scatters of the distribution of the features data [13]. Therefore, they show promise for data stream applications that are dynamic rather than static. These linearly-dependent measures are illustrated below. For all similarity measures, x denotes a cluster centroid and y denotes a feature arriving from a stream.

- **Pearson Correlation Coefficient (PCC)**
  PCC [23] is a coefficient that measures the linear dependency of two variables. This can be calculated between two features or between a feature and a class label. Unlike the latter, which calculates the extent to which the features are correlated to their class labels, the former is applied to calculate the relationship between the cluster centroids and the streaming features to assign each feature to its corresponding cluster. It is convenient to use a feature and a feature correlation approach in our case as we focus on unsupervised learning where we assume there is no class label. In general, the dependency coefficient is faster and more capable of determining suitable representative features without the need for pairwise correlation computation. Pearson Correlation Coefficient can be computed as follows.

$$\text{PCC(x, y)} = \frac{n(\sum xy) - (\sum x)(\sum y))}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}} \tag{1}$$

  where n is the number of instances. The result of the dependency correlation between any two variables x and y is between 0 and 1. 0 indicates that there is no dependency correlation between the feature and the cluster centroid, and 1 indicates that there is a complete dependency correlation between the two variables.
- **Least Square Regression Error (LSRE)**
  LSRE [24] calculates the degree of the dependency relationship between each feature and a set of corresponding cluster centroids. The error is determined by calculating the distance between the model data and the actual data using the following equations:

$$\text{LSRE(x, y)} = y_n - (ax_n + b) \tag{2}$$

  where a and b variables are given by minimising the mean square error. n variable denotes the total number of features which is always (1) since we process a single feature each time. a variable represents the slope of variable x and can be computed as follows:

$$a = \frac{\sum xy - \frac{\sum x \sum y}{n}}{\sum x^2 - \frac{(\sum x)^2}{n}} \tag{3}$$

  While the variable b is the y-Â-intercept and can be computed as follows:

$$b = \frac{\sum y - (a\sum x)}{n} \tag{4}$$

  The result of the previous equations is the degree of linear dependency correlation between the features from a stream and a cluster centroid based on the value obtained with equation 2. $LSRE = 0$ indicates that there is a complete linear dependency correlation.
- **Maximal Information Compression Index (MICI)**

MICI [13] measures the degree of similarity between two variables, such as a feature and a cluster centroid. MICI can be defined as $MICI(x, y) =$ the smallest eigenvalue of $\sum$,i.e.,where $\sum$ represents the covariance matrix for random features.

$$\begin{aligned} &MICI(x, y) \\ &\quad = (var(x) + var(y)) \\ &\qquad - \sqrt{(var(x) + var(y))^2 - 4var(x)var(y)(1 - \rho(x, y)^2))} \end{aligned} \tag{5}$$

linearly-dependent relationship increases as much as the amount of dependency $MICI$ decreases. There is a linearly-dependent between a feature and a cluster centroid when $MICI$ value is 0.

IV. THE UFSSF METHOD

This section presents the proposed UFSSF method. We start by defining two main concepts of *representative feature* and *cluster centroid.*

Definition-1 (Representative Feature): any feature is considered to be *representative* feature of a cluster if the feature has maximum similarity to all other features that are assigned to the cluster. f_r is considered to be a *representative feature* for a given a cluster centroid c_r, namely $f_r \in R_t$, if and only if one of the following properties is present:

$$PCC(f_r, c_r) > PCC(f_j, c_r) \quad (6)$$

$$LSRE(f_r, c_r) < LSRE(f_j, c_r) \quad (7)$$

$$MICI(f_r, c_r) < MICI(f_j, c_r) \quad (8)$$

where R_t is the representative features set and f_j represents a feature of c_r. A feature is discarded if it is not representative. This leads to efficient usage in a dynamic feature space and better filtering of non-representative features.

Definition-2 (Cluster Centroid): a feature cluster is represented by a cluster centroid that is the weighted mean of the features assigned to the cluster centroid. The weights are the smallest for features that arrived in the distant past, and are the largest for the most recently-arrived features.

A. THE PROPOSED UFSSF METHOD

This section presents the proposed UFSSF method and demonstrates how it identifies the set of representative features. The UFSSF method involves two main stages: 1) identification of possible features that can be added to the representative features set; and 2) removal from the set of representative features all the features that are no longer representative. We integrated the measures for similarity that are mentioned in Section III to the UFSSF as a distance measure parameter. Linearly-dependent measures are considered to be more effective for the feature selection process because they are insensitive to the order of features and their scattered distribution. The three measures, namely PCC, LSRE and MICI, are proven to be effective for the feature selection purpose as shown by the experiments conducted by in [13]. Hence, those measures are applied to calculate the dependency correlation between the features and the cluster centroids by using the k-mean clustering algorithm. Furthermore, the proposed method uses a cluster-based approach that can determine the representative features without the need for class labels. The k-mean clustering algorithm [25] is a convenient solution that can work with multidimensional data and is suitable for streaming features. The process whereby the UFSSF method determines a representative set of a stream of features is depicted in Figure 1. The features are processed upon arrival on a first-in-first-out basis. The first step is the initialisation of the representative features and clusters:

- The newly-arrived k features form the initial centroids of k clusters (e.g., if the value of k is 10, the first collected 10 features are the first 10 initial clusters centroids as shown in Algorithm 1, Line 1.
- The initial centroid of each cluster is set as the initial representative feature as shown in Algorithm 1, Line 2).

Whenever a feature f_j arrives:

The following steps show how to update the representative features set:

- To compute the similarity between the current feature f_j and the clusters centroid of each cluster, the similarity measure needs to be specified; either PCC, LSRE or MICI as shown in Algorithm 1, Line 4. The equations for these similarity measures are given in Section 3. The feature f_j is assigned to a cluster C_{lus} if the feature f_j has the maximum level of similarity to a cluster centroid C_{lus} as shown in Algorithm 1, Line 5. Then, the centroid C_r of C_{lus} is updated incrementally. Formally, $C_r + f_j/2$. The feature f_j is assigned to the cluster's centroid C_{lus} as shown in Algorithm 1, Line 6.
- In C_{lus}, the similarity (say S) of the feature f_j is compared with the representative feature (i.e. f_r) with C_{lus}'s centroid c_r as shown in Algorithm 1, Lines 7-8. In case $S(f_j, c_r) > S(f_r, c_r)$, the feature f_j is the representative feature of C_{lus} and the feature f_r is removed as shown in Algorithm 1, Lines 9-12.
- The representative feature from each cluster comprises the representative features set as shown in Algorithm 1, Line 14.

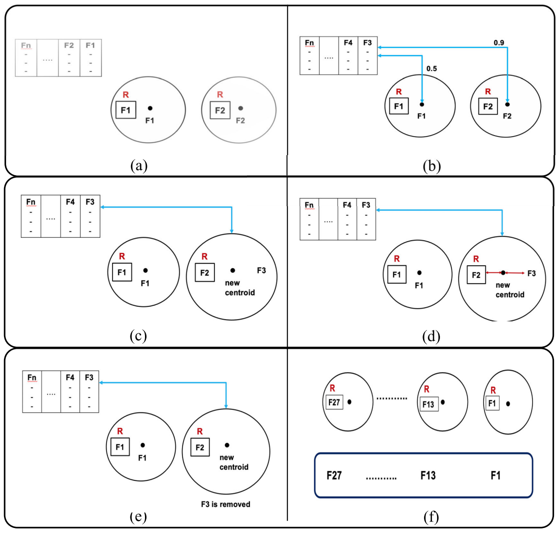


FIGURE 1. The process of how UFSSF method determines a representative set of a stream of features: (a) initialization of clusters and representative features, (b) when F3 arrives, UFSSF finds relevant cluster by computing similarity, (c) updating the cluster centroid when F3 has been allocated, (d) checking if F2 is still a representative feature, (e) F2 is kept if it has max similarity to the centroid compared to F3, (f) the representative feature of each cluster comprises the set of representative features.

The UFSSF method reads the data stream only once, which is a single pass over the data. Moreover, UFSSF method gradually updates the cluster's mean: i) to ensure the representativeness of the selected features (e.g., a selected feature f_r is consider as a representative at time t and it is not representative at time $t + 1$); and ii) to overcome the drift concept of the cluster's selection process due to the stream's dynamic nature. This can improve the accuracy of the classifier's prediction process.

To sum up, for each cluster, the developed UFSSF method stores a centroid and a single representative feature with the greatest similarity to the cluster centroid, as all other features are discarded. Hence, the UFSSF requires less storage as it does not need to retain other features assigned to clusters. Actually, it only computes their statistics by updating the clusters centroids incrementally. Then, they are removed if they are not representative. We believe that the UFSSF method is able to achieve most of the streaming applications requirements, and therefore it can be efficiently deployed for applications that have streaming features similar to those presented in the experimental section.

V. EXPERIMENTAL SETUP

This section presents the experiment specifications and the results of the proposed UFSSF method. The method evaluation is going to answer the following questions:

- How accurate is the proposed UFSSF method in identifying a representative features set?
- How efficient is the proposed UFSSF method in terms of execution time?

The experiments are conducted in two phases in the following to the experimental settings given in [3] [9].

- The first part simulates the environment of streaming features while ensuring the following: 1) the streaming features are unknown in advance; and 2) the streaming features are processed in real time [9]. There are four subsets of feature space: 20%, 40%, 60% and 80%. First, 20% subset of streaming features is picked and then the following 40% and then the feature space is increases sequentially until the 80% arrival of features set is simulated. UFSSF, SPEC [12] and [13] are applied for each subset to determine the representative features set. To ensure a fair comparison of the performance of the three methods, all of them choose the same number of features.
- The second part evaluates the feature space using sets with different numbers of features from a complete feature stream, that is, 100% of the features. This evaluation helps to determine the consistency of the results, and prevents randomness.

The performance of the proposed UFSSF method is compared with the proposed method in [13] and SPEC [12], which are two well-known classical unsupervised feature selection methods. To the best of the author's knowledge, no methods for unsupervised feature selection have been proposed without the need to link information. The setting of the experiment evaluation ensures that the comparison criteria are fair, although the selected benchmark methods have not been designed specifically for the purpose of streaming features applications. UFSSF, SPEC [12] and [13] are applied to each feature space subset separately and the same number of features are chosen. Moreover, the dataset is examined using different sized sets of representative features for each method in order to compare the performances of the non-streaming features benchmark methods.

The extracted sets of representative features obtained by the three selected methods are examined by calculating the average of J48 Decision Tree [26], Naive Bayes [27] and Lazy Nearest Neighbour [28], which are three well-known classifiers. Furthermore, k-fold-cross validation is performed on the complete set of selected features to produce improved results by eliminating over-fitting data. The set of features that is selected is first split into equal size subsets depending on the pre-defined k folds. Second, one of the k folds is applied as a selected testing subset and the remaining subsets are used for training purposes. Finally, the average result is determined by averaging the values of the complete set of folds. In the evaluation settings, the k value is pre-set at 10 to demonstrate the efficiency of the proposed method, as suggested by in [29]. The benchmark methods are implemented in a MATLAB programming environment. They are run on a Mac operating system OS X EI Capitan with 2.4 GHz Intel Core 2 Duo and 8 GB RAM.

A. DATASETS

Three datasets are used to evaluate the performance of the proposed UFSSF method and those of the selected benchmark methods. The selected datasets are used due to their popularity for data mining algorithms and are collected from diverse data domains where they have been extracted from UCI Machine Learning Repository as shown below. The input datasets have been chosen mainly for clustering and classification purposes because clustering is one of the main processes in the proposed UFSSF method which is applied to determine the representative features. A brief description of each dataset is given below.

- **Spambase**[1] dataset: it contains a multivariate dataset of both spam and non-spam email classes. Each email has 57 real data type features. There are 4601 records of emails in the Spambase dataset.
- **Waveform**[2]: is a multivariate data set that consists of 40 continuous features, some of which are noise. These features have 5000 instances.
- **Ionosphere**[3]: This dataset is collected from a radar that is provided by a system in Goose Bay, Labrador. The system consists of 16 antennas with 6.4 kilowatts power transmission. The aim is to investigate whether electrons exist in the ionosphere. The radar returns two classes: either good or bad. The dataset consists of 351 instances that are described by 34 features.

B. EVALUATION METRICS

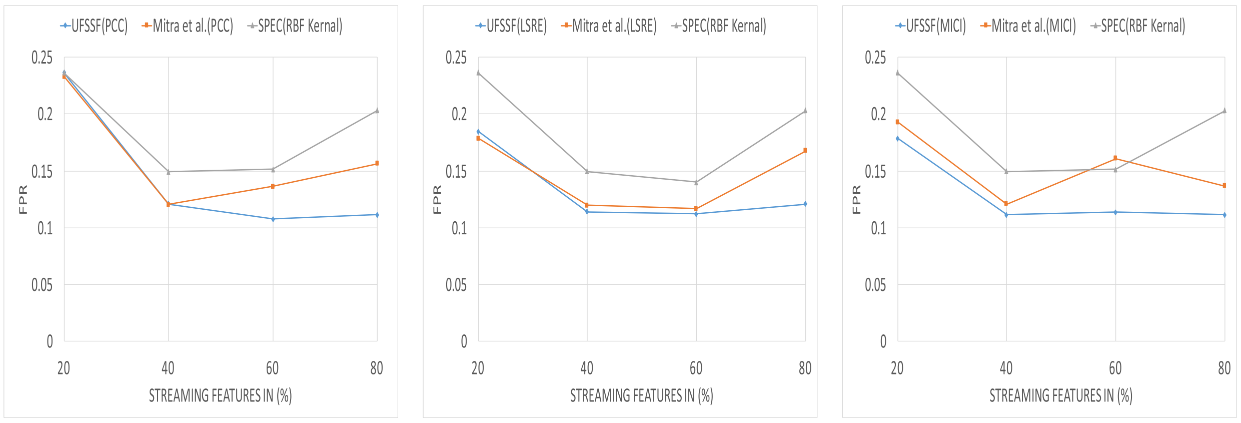
The proposed UFSSS method aims to determine a representative subset of streaming features. The selected representative features should improve the accuracy of the classifier. The selection of the streaming features should be performed within a reasonable execution time. Hence, the metrics considered for the evaluation are grouped according to classification accuracy metrics and running time.

1) **Classification accuracy metrics**
Three classification accuracy metrics are adopted, namely Precision, False Positive Rate (FPR) and

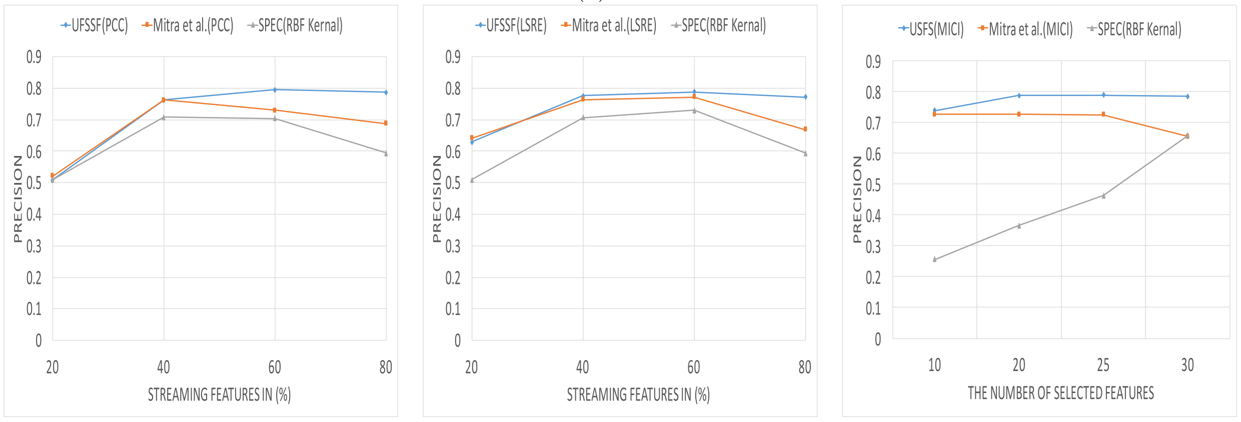
[1]https://archive.ics.uci.edu/ml/datasets/Spambase

[2]https://archive.ics.uci.edu/ml/datasets/Waveform+Database+Generator+(Version+2)

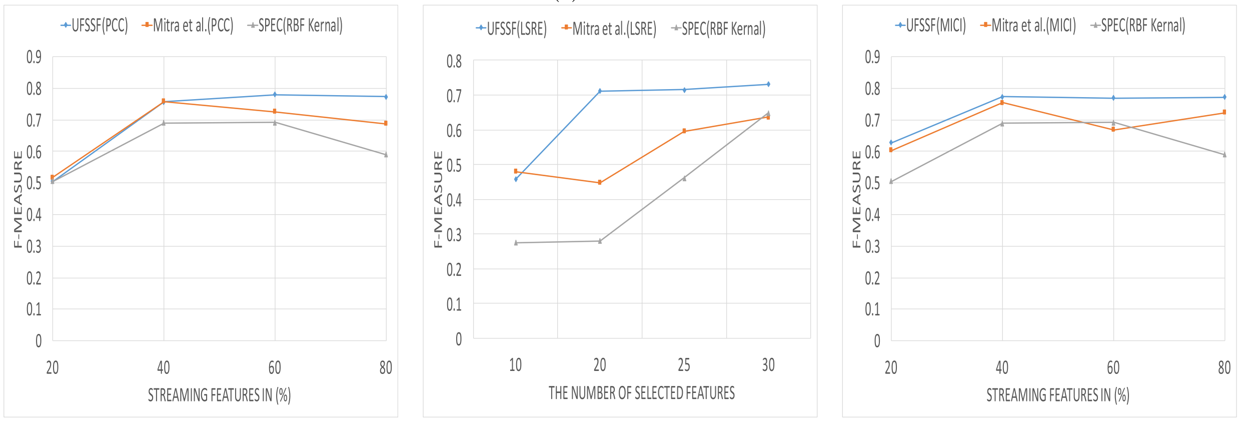
[3]https://archive.ics.uci.edu/ml/datasets/Ionosphere



$(a) FPR$



$(b) Precision$



$(c) F - measure$

FIGURE 2. Comparison of classification accuracy metrics of different methods applied to the Waveform dataset. The average results of Naive Bayes, IB1 and J48 decision tree classifiers are computed. The first, second and third rows of the figure show the results of the FPR, Precision and F-measure evaluation metrics, respectively. The columns the results when different similarity measures are applied. The x-axis denotes the percentage of arrived streaming features, while the y-axis denotes the corresponding accuracy metric. The reduction rate of each percentage is set to 30%.

F-measure. These metrics determine whether the selected stream features subset is competitively improving the accuracy of the classifiers.

2) **Running time**
UFSSF's running time is compared with Mitra's method [13] and SPEC [12] to evaluate their efficiency. To precisely measure the running time, we apply these methods to different percentages of arrived streaming features and count the time that each method takes to select a subset of features.

VI. EXPERIMENTAL RESULTS

This section presents the analysis results of experiments carried out for the UFSSF and the benchmark methods. Three evaluation criteria are applied to determine the accuracy of the feature selection process, namely FPR, Precision

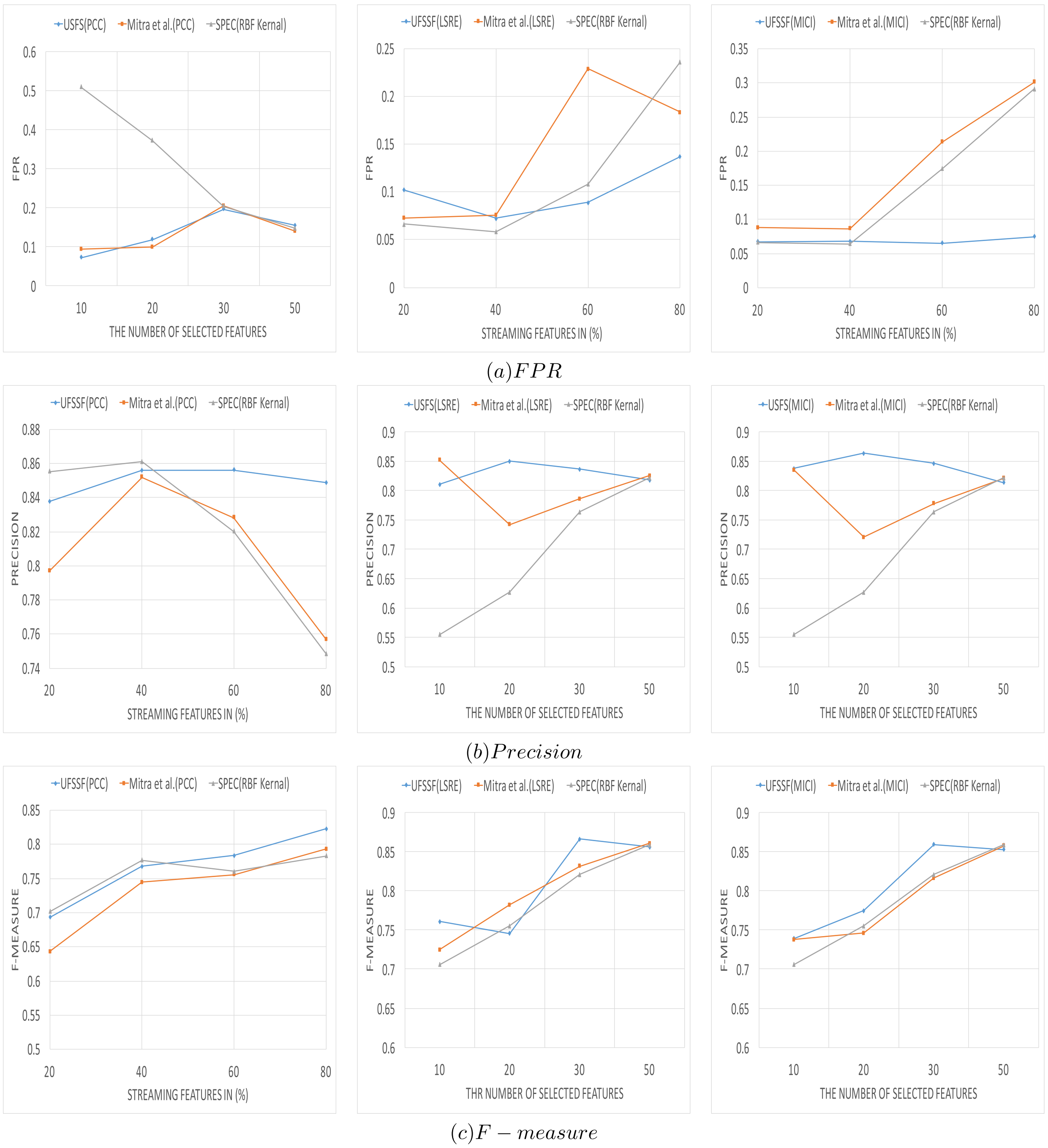


FIGURE 3. Comparison of classification accuracy metrics of different methods applied to the Spambase dataset. The average results of Naive Bayes, IB1 and J48 decision tree classifiers are computed. The first, second and third rows of the figure show the results of the FPR, Precision and F-measure evaluation metrics, respectively. The columns show the results when different similarity measures are applied. The x-axis denotes the percentage of arrived streaming features, while the y-axis denotes the corresponding accuracy metric. The reduction rate of each percentage is set to 30%.

and F-measure. Furthermore, the execution time is measured in seconds. First, we show the analysis results of the features stream where the features are unknown initially and arrive sequentially. Second, we show the results that are related to the use of the complete data stream by conducting an investigation using varying numbers of selected features to show the consistency of the analysis results. Finally, we present results showing the efficiency of the UFSSF in terms of execution time, along with the results for other two benchmark methods. Similarity measures are applied to each method to determine its analysis accuracy for each dataset. Mitra's method [13] involves three similarity measures, specifically LSRE, PCC and MICI), while SPEC can work with the RBF Kernel similarity measure.

Figures 2, 3 and 4 present the streaming features results when the features arrive sequentially and are unknown in

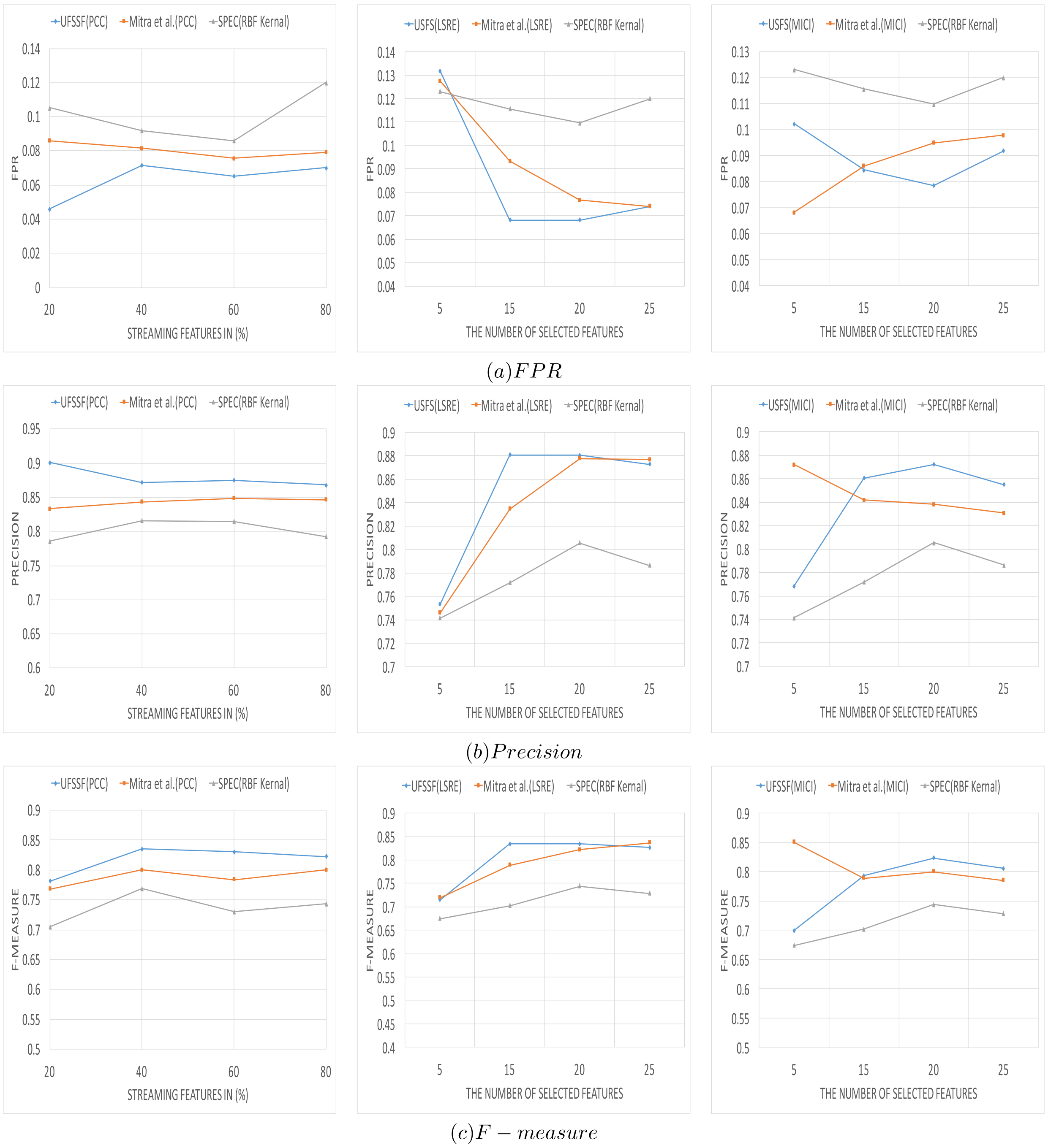


FIGURE 4. Comparison of classification accuracy metrics of different methods applied to the Ionosphere dataset. The average results of Naive Bayes, IB1 and J48 decision tree classifiers are computed. The first, second and third rows of the figure show respectively the results of the FPR, Precision and F-measure evaluation metrics. The columns show the results when different similarity measures are applied. The x-axis denotes the percentage of arrived streaming features, while the y-axis denotes the corresponding accuracy metric. The reduction rate of each percentage is set to 30%.

advance. The proposed UFSSF outperforms other methods in [13] and SPEC [12] with all percentages of streaming features according to the similarity measure results for the Waveform dataset as shown in Figure 2. Generally speaking, the UFSSF has the highest precision and F-measure, and the lowest FPR. Although both the UFSSF method and the method proposed in [13] have similar accuracy levels in the early stage when 20 and 40 percent streaming features have arrived, the accuracy of the UFSSF method increases significantly for other percentages of arriving sets of streaming features, such as 60% - 80%.

Similarly, for the Spambase dataset, as shown in Figure 3, the UFSSF tends to consistently have the lowest FPR and the highest F-measure and precision from 60% until the arrival of 80% of the features. Again, the accuracy of the UFSSF is similar to that achieved by the

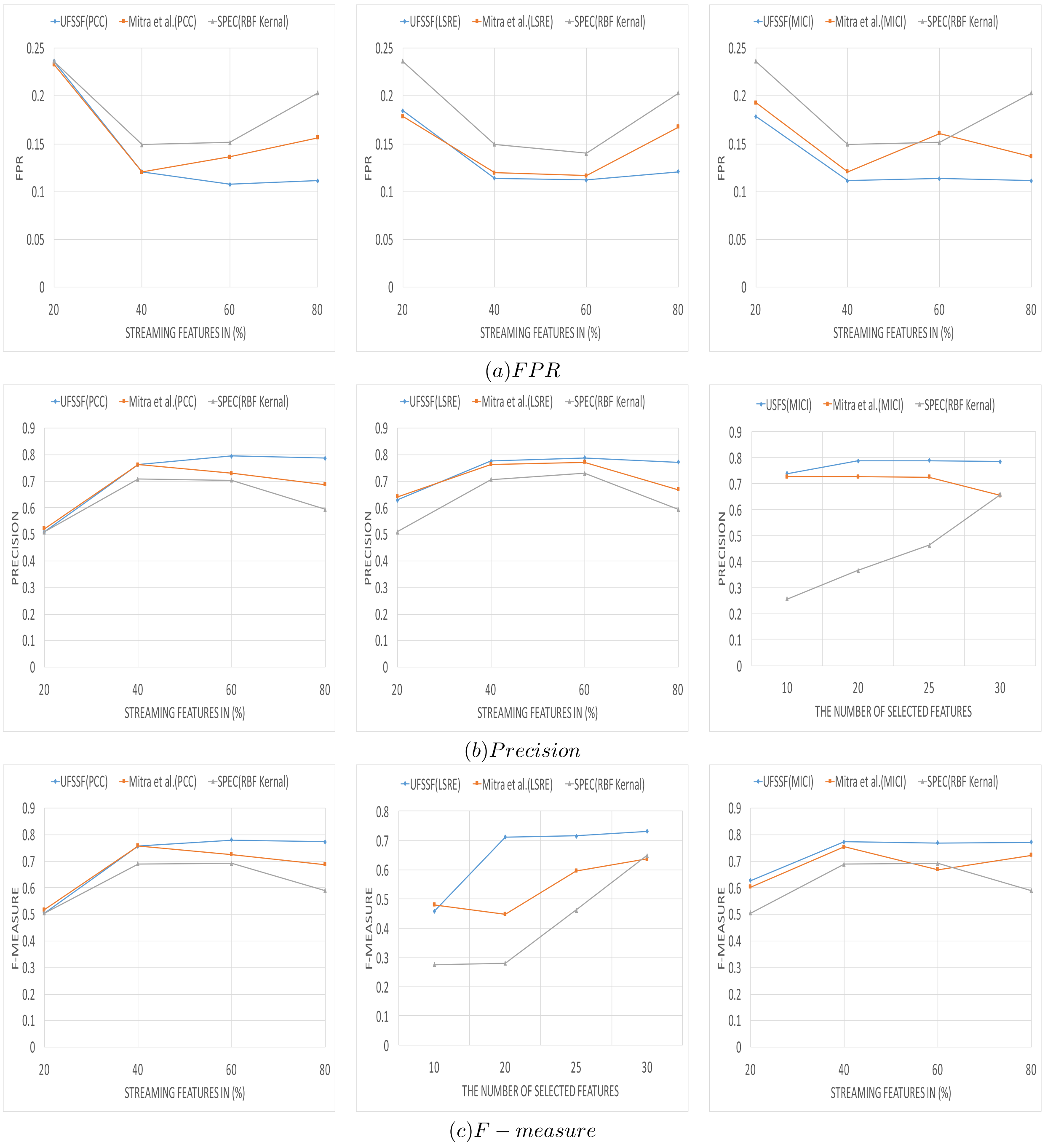


FIGURE 5. Comparison of classification accuracy metrics of different methods applied to the Waveform dataset. The average results of Naive Bayes, IB1 and J48 decision tree classifiers are computed. The first, second and third rows of the figure show the results of the FPR, Precision and F-measure evaluation metrics, respectively. The columns of the figure show the results when different similarity measures are applied. The x-axis denotes various numbers of selected features, while the y-axis denotes the corresponding accuracy metric.

two baseline methods in the early stages of feature arrivals (i.e., 20% and 40%). The analysis accuracy performs significantly better when more features arrive from Spambase and Waveform datasets. In fact, the UFSSF develops the model gradually because of the incremental update of clusters which in turns affects the process of representative features selection. UFSSF incrementally processes a stream of features one-by-one and determines the representative feature seen so far from a cluster. Hence, in a limited scenario, the UFSSF is forced to select the maximum number of representative features that have just arrived where there are no good representative features. Therefore, with the arrival of more features, the accuracy is improved. In contrast, the other two methods search the complete subset of streaming features that have arrived in order to determine the best features.

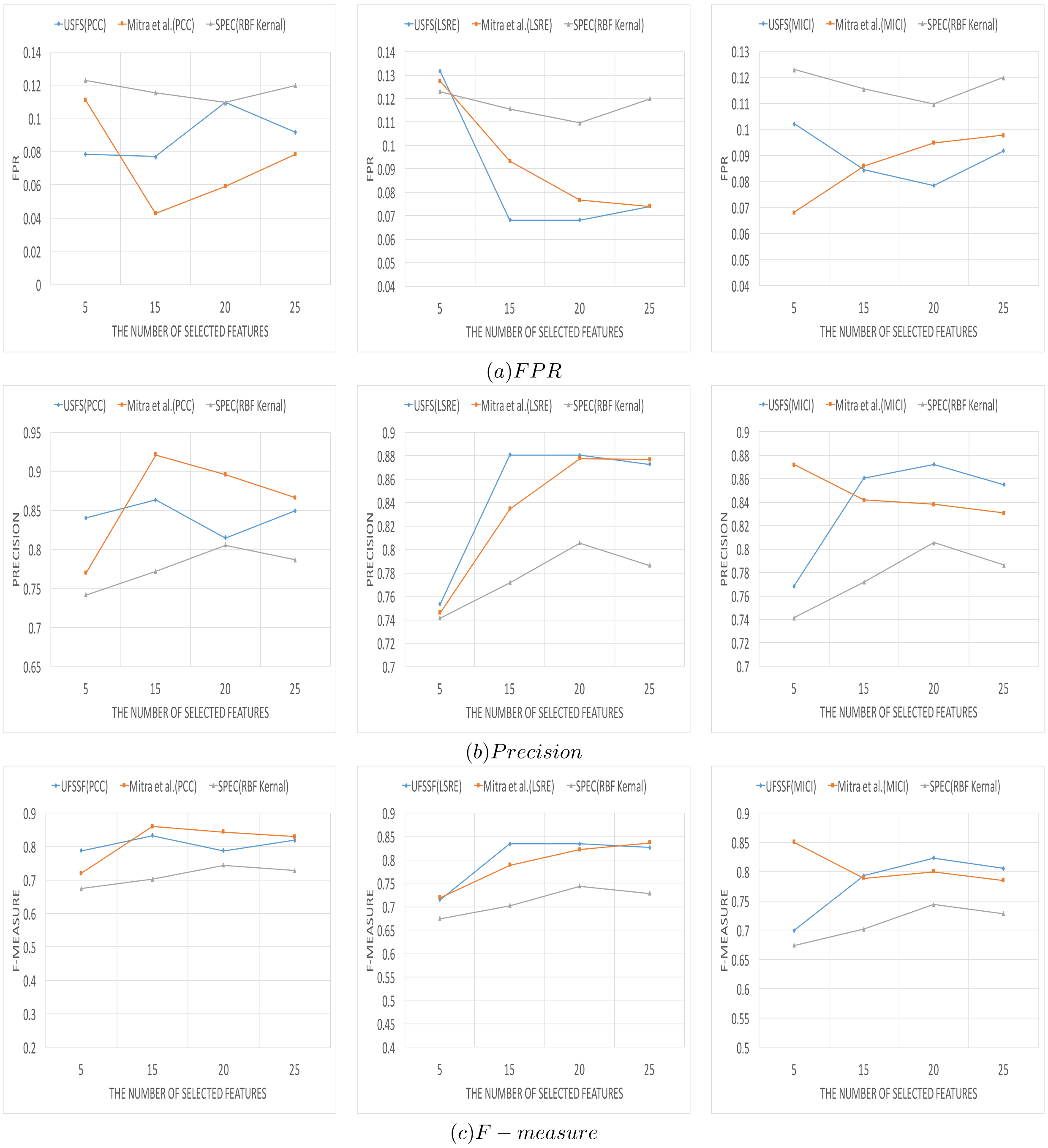
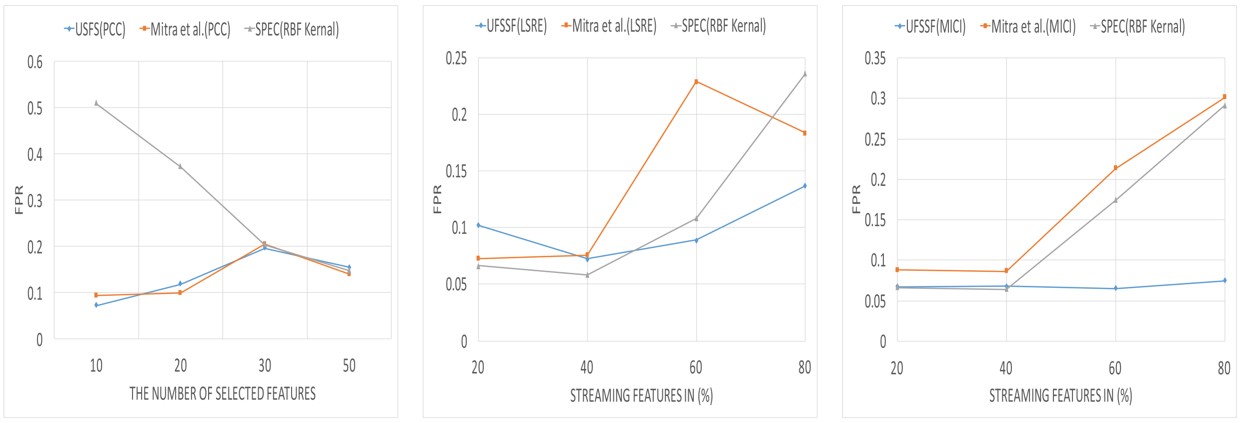


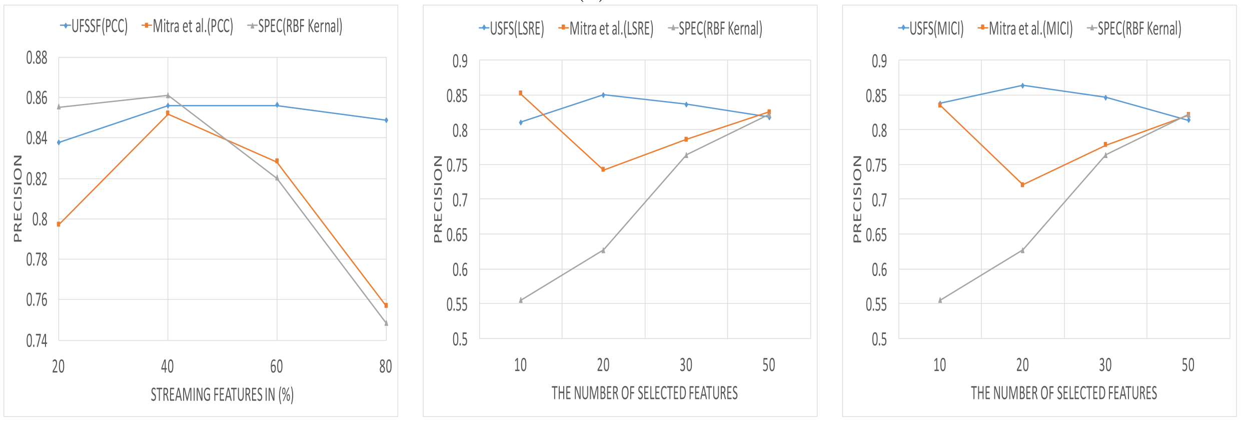
FIGURE 6. **Comparison of classification accuracy metrics of different methods applied to the Ionosphere dataset. The average results of Naive Bayes, IB1 and J48 decision tree classifiers are computed. The first, second and third rows of the figure show the results of the FPR, Precision and F-measure evaluation metrics, respectively. The columns show the results when different similarity measures are applied. The x-axis denotes various numbers of selected features, while the y-axis denotes the corresponding accuracy metric.**

Furthermore, the UFSSF has the significant lowest FPR and the highest F-measure and precision for all varying streaming features percentages compared to the other methods applied to the Ionosphere dataset, as shown in Figure 4. This is valid when using either LSRE, PCC or MICI as the similarity measure for UFSSF and [13].

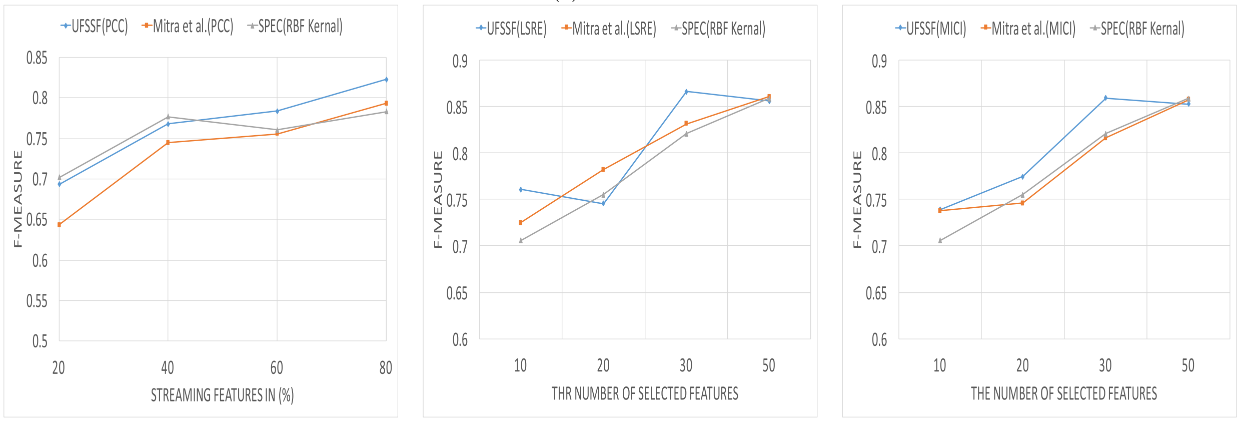
Figures 2, 3 and 4 show that that UFSSF method achieves better accuracy with an average of 5% than the benchmark methods, specifically with the arrival of 60% and 80% of the features. SPEC [12] and Mitra's method [13] do not incrementally update their models to overcome the problem of the dynamic nature of the stream. Any stream feature can be representative at a specific time period because of its dynamic nature. In contrast, the UFSSF updates its clusters in an incremental manner to ensure that the newly-arrived feature can be included in the current selected



$(a) FPR$



$(b) Precision$

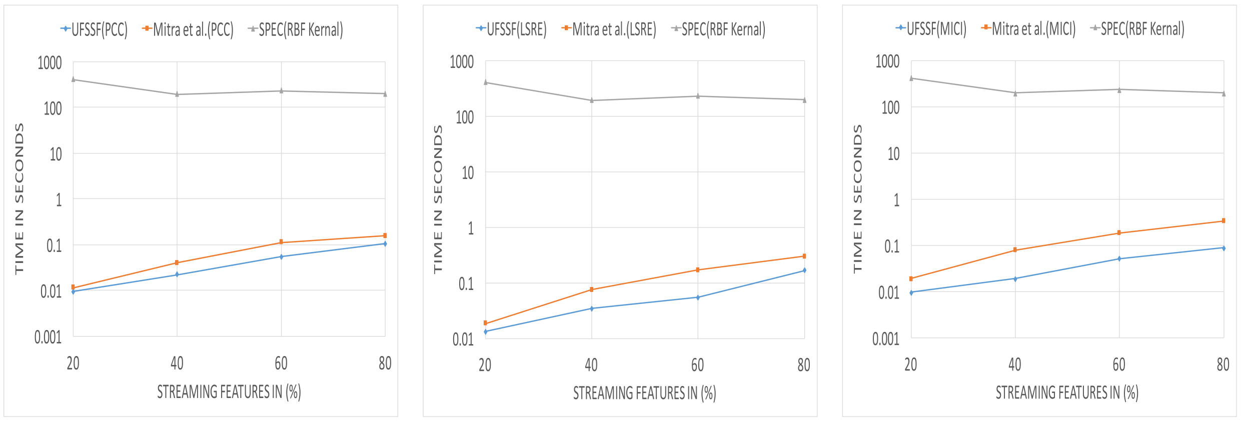


$(c) F - measure$

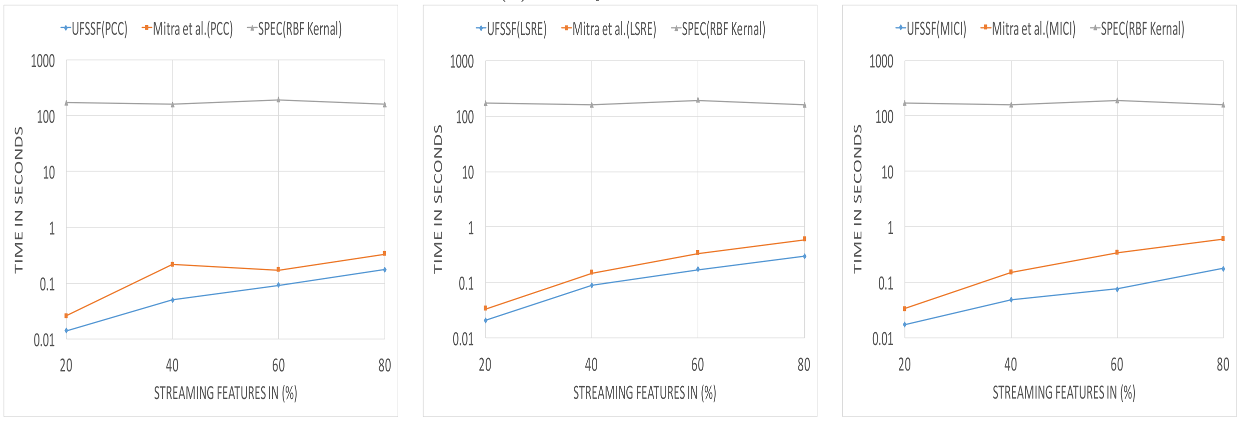
FIGURE 7. Comparison of classification accuracy metrics of different methods applied to the Spambase dataset. The average results of Naive Bayes, IB1 and J48 decision tree classifiers are computed. The first, second and third rows of the figure show, respectively, the results of the FPR, Precision and F-measure evaluation metrics. The columns show the results when different similarity measures are applied. The x-axis denotes the various numbers of selected features, while the y-axis denotes the corresponding accuracy metric.

representative features. Therefore, the UFSSF demonstrates that its feature-selection process outperforms those of other methods in the classification process.

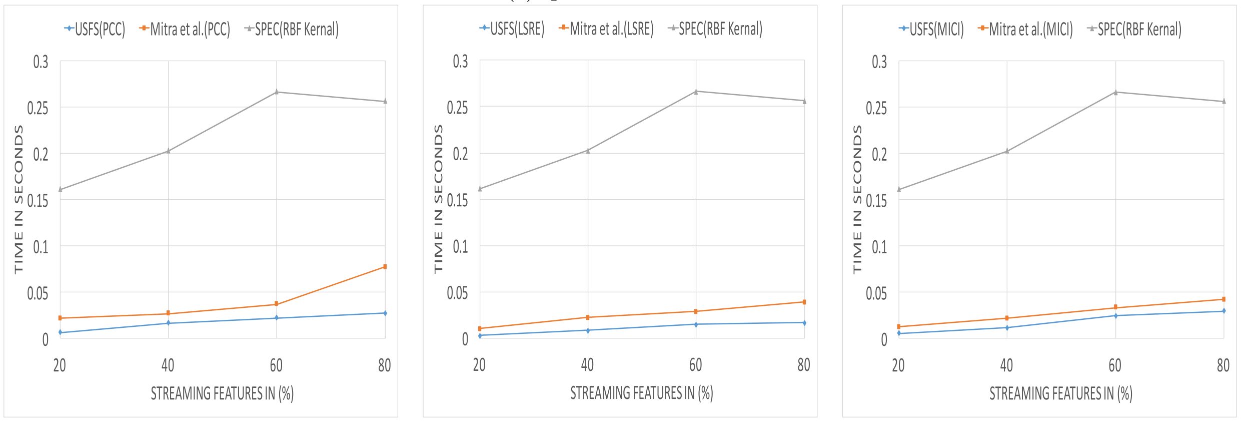
Figures 5, 6 and 7 demonstrate the accuracy of the UFSSF and that of the baseline methods when selecting varying sizes of features sets while taking into consideration the complete set of feature space as a stream (i.e. 100%). Figure 5, UFSSF remarkably outperforms other methods (lowest FPR and highest precision and F-measure) for the Waveform dataset, followed by SPEC [12]. This is the case for all varying sizes of selected features sets and for all similarity measures. Moreover, the UFSSF and [13] show almost similar accuracy analysis results when LSRE and MICI are applied as similarity measures for a set of 10 features. In Figures 6 and 7, the UFSSF has either a slightly better or a competitive prediction accuracy compared to [13] and



$(a) Waveform Dataset$



$(b) Spambase Dataset$



$(c) Ionosphere Dataset$

FIGURE 8. Comparison of the running times of the various methods for three datasets. The first, second and third rows of the figure show the running times of the Waveform, Spambase and Ionosphere datasets respectively. The columns show the results in terms of different similarity measures. The x-axis denotes the percentage of arrived streaming features, while the y-axis denotes the time (in seconds) that each method takes to select a set of features.

SPEC [12] for the Ionosphere and Spambase datasets. This holds for varying numbers of selected features sets and for all the applied similarity measures. The selected benchmark methods have been developed to be applicable to statistical datasets. However, our UFSSF method is intended for stream-based applications where the selected set of features is unknown in advance and features arrive sequentially one-by-one. Although, the UFSSF shows lower analysis accuracy levels compared to the benchmark methods for a few sets of selected features, these differences are negligible.

It is important to mention that the developed UFSSF, which is the method applied for selecting the representative set of features, is insensitive to the order in which features are received, With the proposed method, each cluster ensures that only the most representative features with maximum similarity to the cluster centroid are retained. Hence, the order in which features are received is not an influencing factor since the similarity ratio is the only value that is taken into account when determining a representative feature.

The execution time for the selected methods is shown in Figure 8. The results show that the UFSSF method has the lowest execution time in all applied similarity measures for the selected datasets. Furthermore, the UFSSF method outperforms the baseline selected methods on all the varying percentages of the streaming features. The method in [13] is competitive with the UFSSF while SPEC [12] has a higher execution time. [13] depends on a K-Nearest Neighbour (K-NN) search to split the arrived features subset. This results in a higher execution time beacsue of the computation of the similarity between features. The SPEC [12] performance is the worst in terms of execution time because of the time required to construct the Laplacian matrix. The UFSSF method achieves the best execution time because it does not need to search the entire subset of arrived features in the selection feature process compared with other methods. Instead, the UFSSF method processes the features sequentially one-by-one by calculating each feature's dependency on the cluster's centroids, which are few compared with the number of streaming features.

VII. CONCLUSION

This paper developed an unsupervised feature selection method for effective dynamic features which can reduce the dimensionality of streaming features applications, known as the dynamic feature space. In these applications, traditional features selection methods are not practical as all features must be available in advance; rather, they arrive sequentially one-by-one for the learning machines. The proposed UFSSF method, unlike existing streaming features methods that require class labels, can efficiently determine a set of representative features without requiring any advance knowledge about class labels. Therefore, it is appropriate for a wide range of applications. With the UFSSF method, the k-mean clustering algorithm is adapted for applications that involve streaming features. k-mean algorithms can cluster a stream of features that are unknown in advance. It applies LSRE, PCC and MICI similarity measures in order to: a) assign a newly-arrived feature to a relevant cluster; b) determine whether a feature can be added to the representative features; and c) determine whether to dynamically update a set of selected features by removing non-representative features. The experiments considered: 1) the streaming features settings where features are unknown in advance and arrive in varying percentages; 2) the entire features space as a stream with varying numbers of selected features, to ensure the stability of the findings; and 3) the time taken by every method to generate its selected features. Experimental results show that UFSSF generates a set of representative features with the lowest execution time compared with other methods. The selected set of representative features achieved superior prediction accuracy based on the precision, FPR and F-measure evaluation metrics.

REFERENCES

[1] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, p. 94, 2017.
[2] X.-T. Wang and X.-Z. Luan, "Bayesian penalized method for streaming feature selection," *IEEE Access*, vol. 7, pp. 103815–103822, 2019.
[3] J. Li, X. Hu, J. Tang, and H. Liu, "Unsupervised streaming feature selection in social media," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 1041–1050.
[4] G. Nandi, "An enhanced approach to Las Vegas filter (LVF) feature selection algorithm," in *Proc. 2nd Nat. Conf. Emerg. Trends Appl. Comput. Sci.*, Mar. 2011, pp. 1–3.
[5] G. Doquire and M. Verleysen, "Feature selection with missing data using mutual information estimators," *Neurocomputing*, vol. 90, pp. 3–11, Aug. 2012.
[6] M. S. Sainin and R. Alfred, "A genetic based wrapper feature selection approach using nearest neighbour distance matrix," in *Proc. 3rd Conf. Data Mining Optim. (DMO)*, Jun. 2011, pp. 237–242.
[7] S. Perkins, K. Lacker, and J. Theiler, "Grafting: Fast, incremental feature selection by gradient descent in function space," *J. Mach. Learn. Res.*, vol. 3, pp. 1333–1356, Mar. 2003.
[8] J. Zhou, D. Foster, R. Stine, and L. Ungar, "Streaming feature selection using alpha-investing," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2005, pp. 384–393.
[9] X. Wu, K. Yu, W. Ding, H. Wang, and X. Zhu, "Online feature selection with streaming features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 5, pp. 1178–1192, May 2013.
[10] N. Almusallam, Z. Tari, J. Chan, and A. AlHarthi, "Ufssf-an efficient unsupervised feature selection for streaming features," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, 2018, pp. 495–507.
[11] J. Han, J. Pei, and M. Kamber, *Data Mining: Concepts and Techniques*. Amsterdam, The Netherlands: Elsevier, 2011.
[12] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, 2007, pp. 1151–1157.
[13] P. Mitra, C. A. Murthy, and S. K. Pal, "Unsupervised feature selection using feature similarity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 301–312, Mar. 2002.
[14] R. Wald, T. M. Khoshgoftaar, and A. Napolitano, "How the choice of wrapper learner and performance metric affects subset evaluation," in *Proc. IEEE 25th Int. Conf. Tools with Artif. Intell.*, Nov. 2013, pp. 426–432.
[15] N. N. R. R. Suri, M. N. Murty, and G. Athithan, "Unsupervised feature selection for outlier detection in categorical data using mutual information," in *Proc. 12th Int. Conf. Hybrid Intell. Syst. (HIS)*, Dec. 2012, pp. 253–258.
[16] S. Jiang and L. Wang, "Unsupervised feature selection based on clustering," in *Proc. IEEE 5th Int. Conf. Bio-Inspired Comput., Theories Appl. (BIC-TA)*, Sep. 2010, pp. 263–270.
[17] C.-N. Hsu, H.-J. Huang, and S. Dietrich, "The ANNIGMA-wrapper approach to fast feature selection for neural nets," *IEEE Trans. Syst., Man Cybern., B, Cybern.*, vol. 32, no. 2, pp. 207–212, Apr. 2002.
[18] H. Zhou, J. Wu, Y. Wang, and M. Tian, "Wrapper approach for feature subset selection using ga," in *Proc. Int. Symp. Intell. Signal Process. Commun. Syst.*, pp. 188–191, Nov./Dec. 2007.
[19] C. Freeman, D. Kulić, and O. Basir, "An evaluation of classifier-specific filter measure performance for feature selection," *Pattern Recognit.*, vol. 48, no. 5, pp. 1812–1826, May 2015.
[20] Y. Hong, S. Kwong, Y. Chang, and Q. Ren, "Consensus unsupervised feature ranking from multiple views," *Pattern Recognit. Lett.*, vol. 29, no. 5, pp. 595–602, Apr. 2008.
[21] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "Recent advances and emerging challenges of feature selection in the context of big data," *Knowl.-Based Syst.*, vol. 86, pp. 33–45, Sep. 2015.
[22] P. Zhou, X. Hu, P. Li, and X. Wu, "OFS-density: A novel online streaming feature selection method," *Pattern Recognit.*, vol. 86, pp. 48–61, Feb. 2019.
[23] A. J. Onwuegbuzie, L. Daniel, and N. L. Leech, "Pearson product-moment correlation coefficient," *Encyclopedia Meas. Statist.*, pp. 751–756, 2007.
[24] C. R. Rao, *Linear Statistical Inference and Its Applications*, vol. 22. Hoboken, NJ, USA: Wiley, 2009.

[25] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, ''An efficient k-means clustering algorithm: Analysis and implementation,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, Jul. 2002.

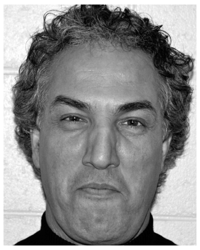
[26] J. R. Quinlan, *C4. 5: Programs for Machine Learning*. Amsterdam, The Netherlands: Elsevier, 2014.

[27] G. H. John and P. Langley, ''Estimating continuous distributions in Bayesian classifiers,'' in *Proc. 11th Conf. Uncertainty Artif. Intell.*, San Mateo, CA, USA: Morgan Kaufmann, 1995, pp. 338–345.

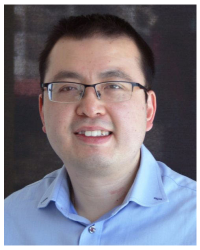
[28] D. W. Aha, D. Kibler, and M. K. Albert, ''Instance-based learning algorithms,'' *Mach. Learn.*, vol. 6, no. 1, pp. 37–66, Jan. 1991.

[29] H.-L. Chen, B. Yang, J. Liu, and D.-Y. Liu, ''A support vector machine classifier with rough set-based feature selection for breast cancer diagnosis,'' *Expert Syst. Appl.*, vol. 38, no. 7, pp. 9014–9022, Jul. 2011.

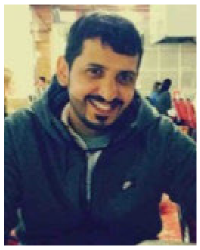
NAIF ALMUSALLAM received the B.S. degree in computer science from King Faisal University, Hofuf, Saudi Arabia, in 2009, the M.S. degree from Monash University, Melbourne, VIC, Australia, in 2013, and the Ph.D. degree in computer science from RMIT University, Melbourne, in 2019. He is currently an Assistant Professor with Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia. His research interests include machine learning, data science, and security.



ZAHIR TARI received the bachelor's degree in mathematics from the University of Algiers, Algiers, Algeria, in 1984, and the M.Sc. degree in operational research and the Ph.D. degree in computer science from the University of Grenoble, Grenoble, France, in 1985 and 1989, respectively. From 1990 to 1992, he was a Researcher with the Database Laboratory, Swiss Federal Institute of Technology, Zurich, Switzerland, where he was involved in various aspects of distributed database systems. He joined the Royal Melbourne Institute of Technology (RMIT), Melbourne, VIC, Australia, as a Senior Lecturer, in 1996, where he is currently a Professor, where he leads the Distributed Systems and Networking Discipline, School of Computer Science and IT. He is currently a Full Professor of distributed systems with RMIT. He is an expert in the areas of system performance (e.g., Web servers, P2P, an cloud) and system security (e.g., SCADA systems and cloud). He has coauthored six books (John Wiley and Springer) and has edited over 25 conference proceedings. He was a recipient of over 5M$ in funding from the Australian Research Council and lately part of a successful 7th Framework Australia to European bid on Ä Authorization and Authentication for Entrusted Unions. He is an Associate Editor of the IEEE Transactions on Computers, the IEEE Transactions on Parallel and Distributed Systems, and *IEEE Magazine on Cloud Computing*.



JEFFREY CHAN is currently a Senior Lecturer with RMIT University, Australia. He is an Honorary Research Fellow with the University of Melbourne. He has published over 35 articles on machine learning, graph analysis, and social computing.



ADIL FAHAD received the B.S. degree in computer science from King Abdul Aziz University, Jeddah, Saudi Arabia, in 2003, and the M.S. degree (Hons.) from the Royal Melbourne Institute of Technology, Melbourne, VIC, Australia, in 2008, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Information Technology. He is also an Assistant Professor with Albaha University. His research interests include wireless sensor networks, mobile networks, and ad hoc networks with emphasis on data mining, statistical analysis/modeling, and machine learning.

ABDULATIF ALABDULATIF (Member, IEEE) received the B.Sc. degree in computer science from Qassim University, Saudi Arabia, in 2008, and the M.Sc. and Ph.D. degrees in computer science from RMIT University, Australia, in 2013 and 2018, respectively. He is currently an Assistant Professor with the School of Computer Science and IT, Qassim University, Saudi Arabia. His research interests include applied cryptography, cloud computing, data mining, and remote healthcare.

MOHAMMED AL-NAEEM (Member, IEEE) received the B.Sc. degree in CIS from the College of Management Science and Planning, King Faisal University, in 2005, and the M.Sc. degree in networks and communications (specializing in network security) and the Ph.D. degree in networks and communications (specializing in wireless networks) from Monash University, Australia, in 2009 and 2015, respectively. He is currently an Assistant Professor with King Faisal University (KFU), Al Ahssa, Saudi Arabia. He is also the Chairman of the Computer Networks and Communications Department, CCSIT, KFU. He is interested in wireless networks, network security, machine learning, artificial intelligence, and pattern recognition.