Journal of Autonomous Intelligence (2023) Volume 6 Issue 3
doi: 10.32629/jai.v6i3.880

ORIGINAL RESEARCH ARTICLE

Stream learning under concept and feature drift: A literature survey

Abubaker Jumaah Rabash[1,*], Mohd Zakree Ahmad Nazri[1], Azrulhizam Shapii[1], Abdulmajeed Al-Jumaily[2]

[1] *Center for Artificial Intelligence Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi 43600, Selangor, Malaysia*

[2] *Department of Signal Theory and Communications, Universidad Carlos III de Madrid, 28903 Madrid, Spain*

*** Corresponding author:** Abubaker Jumaah Rabash, abj87r@gmail.com

ABSTRACT

Stream data learning is an emerging machine learning topic, and it has many challenges. One of its challenges is the dynamic behavior or changes in the environment which leads to drifts. Two types of drift occur, namely, concept drift and feature drift. This article provides a survey on stream data learning with focusing on the issues of feature drift and the methods developed for handling it. After presenting the fundamental concepts and definition in this field, it provides an overview of the various models and methods developed for detecting feature drift and maintaining the validity of the machine learning models when the drift occurs. Furthermore, the article provides the generators used for creating dataset with feature drift to provide benchmarking for approaches of detecting or handling feature drift. The article provides also taxonomy of feature selection methods in both static and dynamic environment. It concludes that reinforcement-based models are promising for this task, and it lists various open challenges and future works in this area.

Keywords: stream learning; concept drift; data stream; feature drift detection

ARTICLE INFO

Received: 4 July 2023
Accepted: 4 September 2023
Available online: 26 September 2023

1. Introduction

Majority of machine learning algorithms were developed initially under the assumption of static environment. The consideration of dynamic behaviors or changes in the environment is a recent aspect of developing the machine learning algorithm and it falls under the category of handling concept drift. This concept occurs when learning within dynamic environment and changes or diversity in the data occurs. In a more formal way, concept drift describes how the target variable's statistical characteristics, which the model is attempting to forecast, vary over time in unexpected ways[1]. Such a change results in the past data losing its significance and relevance, which ultimately produces inaccurate predictions. Concept drift can happen in a variety of applications and with a variety of notions[2]. Basically, concept drift is an existing problem for wide range of data driven systems for decision making or for early warning such as the security systems of intrusion detection. Some examples of concept drift applications are spam categorization[3], weather predictions[4], and financial fraud detection[5,6]. The issue of concept drift and how to manage learning when concept drift occurs are starting to receive more attention in the literature[1]. The concept drift under imbalanced stream data classifications[7], memory management for reoccurring concept[8,9], a specific type of concept drift named as feature drift[10]. Although there are numerous recent research surveys discussing the problem of

intrusion detection system IDS from the machine learning perspective, addressing the challenge of concept drift in IDS is not provided in thorough literature surveys. In order to give the readers an idea about the differences between latest conducted survey in IDS, we present a comparison between them from the perspective of various elements in machine learning, namely, addressing of concept drift in general, addressing of feature drift in particular, considering the curse of dimensionality and addressing, discussion the types of the models developed such as incremental vs. offline learning, supervised or un-supervised, and the specific scope that was considered from the application perspective. Recently, several literature survey articles have tackled the problem of concept drift which is regarded as associated problem with intrusion detection systems. However, considering that concept drift can occur in other type of the applications, some of them have focused on certain applications while others were general. We present a summary of them and the various elements that were addressed in **Table 1**. As it is shown, we find that Khamassi et al.[11] and that of de Barros and de Carvalho Santos[12] were general and have focused on concept drift types that occur in IDS. On the other side, we find that Al-Jarrah et al.[13] has tackled the problem of intrusion detection in the literature, but no focus was given to concept drift.

Table 1. Comparison between recent literature surveys of IDS from the perspective of machine learning.

Survey	Concept drift	Feature drift	Learning types	Dimensionality	Ensembles	Application
[13]	×	×	∠	∠	×	Intra-vehicle networks
[11]	∠	∠	∠	×	×	General
[12]	∠	×	∠	×	∠	General

Hence, survey has considered the first that will address the problem of IDS from the perspective of machine learning with handling the issues of:

1) The batch learning nature of the intrusion detection considering that the data arrives sequentially with a labelling percentage which enables updating the knowledge in an online way.

2) The statistical change in the joint probability of stream data classes and the feature over time. Hence, the challenge of concept is looked into the problem of IDS and the handling of concept drift in IDS systems in the previous approaches is considered as one attribute.

3) The relevance change of the features with respect to time which is named as feature drift is also tackled in this survey and the handling of feature drift in IDS systems in the previous approaches is considered as one attribute.

2. Background

2.1. Stream data

A data stream is a potentially infinite succession of data pieces that arrive at a high rate on a regular basis or non-regular basis. Let us define a data stream mathematically as show in Equation (1).

$$D = [(x_t, y_t)]_1^{\infty} \quad (1)$$

where,

(x_t, y_t) denotes a data item arrived at time stamp t.

$x_t \in R^n$ denotes n-dimensional feature vector.

$y_t \in Y = [c_1, c_2, \dots c_k]$ denotes the ground truth.

2.2. Concept drift

According to Žliobaitė[14], concept drift is a term used to describe an ongoing, non-stationary learning issue. In real-world issues, the training and the application data frequently don't match. They made the assumption that

a series of examples is observed, one instance at a time, not necessarily in evenly spaced time intervals, in their framework for concept drift analysis. This represents the general condition of stream data arrival. Nevertheless, the handling of the issue of concept drift can be in the special case where the data arrives in equally spaced time units. Assuming that the stream data is represented by $X_t \in \Re^p$ where $\Re^p$ is the set of real numbers with the dimension p. X_t denotes sample of feature space observed at time t. Assuming that the label of X_t is $y_t \in \mathcal{Z}^1$. Then, the tuple (X_t, y_t) represents labelled data. The historical data is stored in a buffer $X^H = (X_1, \dots, X_t)$. The goal is to predict the label of X_{t+1}. Hence, it is called as the testing data. Assuming that a training model is built based on X^H and it is denoted by $\mathcal{L}_t$. This model is then used to predict X_{t+1}. However, considering that the sources of data S_1, S_2, ... and S_{t+1} might change with respect to time or there is a moment of time $t = i$ where $S_1 = S_2 = \cdots S_i \neq S_{i+1} = S_{j+1} \dots S_{t+1}$. Then, we say that a concept drift has affected the stream data at the moment $i + 1$. The awareness of concept drift while training the model $\mathcal{L}_t$ is important for enhancing the accuracy of prediction of X_{t+1}. A conceptual diagram that represents the concept drift is represented in **Figure 1** where $i = 3$ is taken as an example. As it is stated in the study of Žliobaitė[14], the core assumption when dealing with concept drift is the uncertainty of knowing the future. Otherwise, the data can be decomposed into separate datasets and learned separately. Also, it is pointed out that periodic seasonality is only regarded as concept drift in case it is not known with certainty.

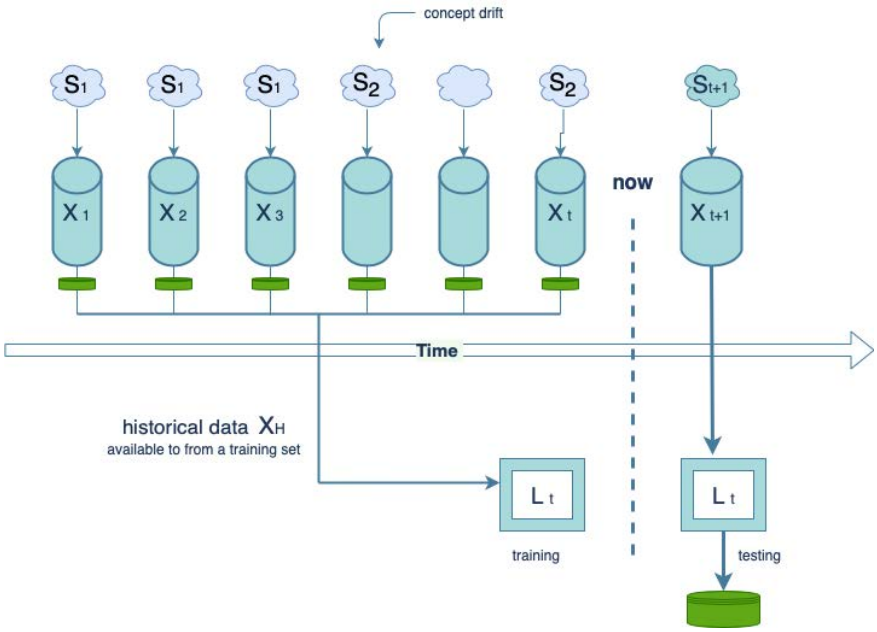

Figure 1. Conceptual diagram to represent the concept drift occurrence.

In a concise way, the concept drift is defined as a phenomena in which a target domain's statistical properties alter in an arbitrary way over time. The idea that noise data can change into non-noise information at any time was initially put forth by Schlimmer and Granger[15]. These changes could be the result of changes in unobservable hidden factors. In a more formal way, it is stated that given a time period $[0, t]$, set of samples $S_{0,t} = \{d_0, d_1, \dots d_t\}$ where $d_i = (X_i, y_i)$ is one observation combined of feature X_i and label y_i and $S_{0,t}$ follows a random distribution $F_{0,t}(X, y)$, concept drift occurred at timestamp $t + 1$ if $F_{0,t}(X, y) \neq F_{t+1,\infty}(X, y)$. Thus, the change in the joint probability of X and Y at time t is the definition of concept drift. Changes in the probabilities of features, features that are provided to a particular class, or a combination of both can set it off. The following subsections contain two examples of concept drift and feature drift.

2.3. Types of concept drift

A. Reoccurring concept drift

When the examples at time $t + \Delta$ are created from the same distribution as those at the previously observed time t, i.e., $F_{0,t1} = F_{0,t+\Delta}$, a repeating idea drift develops. That is, a concept occurs at one moment in time, then vanishes for a lengthy time before reappearing. As an example, a repeating concept drift appears when the person moves from A to B and vice versa in example 1.

B. Concept evolution

A novel class that was previously unknown in the non-stationary stream setting is referred to as concept evolution, while a novel class denotes a class that did not occur in the initial training data set for creating a learning model but does so subsequently. As an example, the person moves to a new room that he has never visited in example 1 or a factory built near the location T in example 2.

1) First example:

A real-world example of concept drift is the Wi-Fi localization given in **Figure 2**. The goal is to use the received Wi-Fi signal strength from different access points to predict the location of the moving subject. The subject might be located in area A and it uses a trained model that is valid for area A, however, when it moves to area B the trained model is not valid anymore due to the change in the physical characteristics between A and B and even the disable and activation of certain Wi-Fi points. Hence, it is important to consider detecting such concept drift in order to enable better predictions in the area B[16].

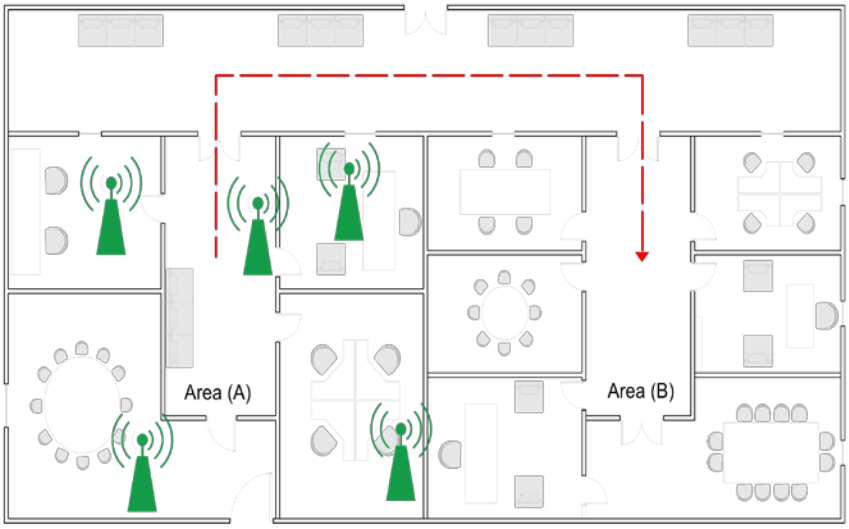

Figure 2. The concept drift example of Wi-Fi localization problem[16].

2) Second example:

Given the readings from surrounding sensors at times t and $t-1$ and, if available, the level of T at $t-1$, it is necessary to anticipate the current level of PM 2.5 (particles less than 2.5 m in diameter) at a target location T. It is difficult to deduce information about air quality because of spatial non-linearities and abrupt temporal changes. Environmental and contextual elements, such as the wind, traffic, and use of heaters in cities, as well as landmarks, have been found to have considerable dependencies on these spatio-temporal correlations[17,18].

3) Drift in IDS

Since the 1980s, intrusion detection has been an active topic of research and a cornerstone of cybersecurity. The primary objectives of an intrusion detection system (IDS) have not changed, despite early research concentrating on host intrusion detection systems (HIDS). A good IDS should have high discriminating power, be able to identify a variety of intrusions—possibly in real-time, improve itself through self-learning, and be adaptable in both design and execution. With the development and extensive use of computer networking, some research attention shifted away from HIDS and toward network intrusion detection systems (NIDS)[19].

Most AI-based IDS models now in use are trained on static data. Data is given in streams in an IDS, and as attack patterns evolve over time, the data distribution may shift over time, leading to idea drift. The IDS must also evolve over time in order to recognize new attack types for it to be effective. Because static batch learning models must be updated over time and become antiquated in certain circumstances, static data models perform poorly.

4) Data stream in classification

Data streams are categorized differently than static data, although only marginally. Data stream classifiers receive data sequentially at a high speed, in contrast to static situations where all the data is available at once for training. Furthermore, when dealing with massive amounts of online data, data stream classifiers must function with limited time and memory. Due to idea drift, a classifier trained on older data may become outdated or useless for fresh data[18]. As a result, concept-drifting learning algorithms must be flexible enough to adapt to changing conceptions. Many concept-drift learning methods have been introduced in the recent years. In this section, we examine earlier methods of IDS's data stream-based classification. An adaptive random forest classifier with an ADWIN change detector is used in the work of D'hooge et al.[19] as a suggested solution to detect change in a data stream and adapt to drift detection in streamed data, resulting in agile adaptation against unforeseen incursions and removing the need to retrain the model over time. The technique was used on the CIC-IDS 2018 and produced accuracy and recall rates of 99.5% and 99.8%, respectively. An opposing self-adaptive grasshopper optimization algorithm is used in the work of Fan et al.[20] and is based on mutation and the perceptive principle. Additionally, a support vector machine known as gain actor critic with support vector machine makes advantage of reinforcement learning to enhance detection by identifying fresh cyberattacks. Extensive tests are run on common intrusion detection datasets including NSL-KDD, AWID, and CIC-IDS 2017 to determine the effectiveness of the proposed technique. In NSL-KDD with six ideal features, the model's accuracy in AWID data was 99.23%, and in CIC-IDS 2017 data, it was 99.15%. The prospect of using machine learning classification algorithms to defend IoT devices against DoS attacks has been investigated in the work of Velayutham and Thangavel[21]. An extensive analysis is conducted on the classifiers that can aid in the advancement of the creation of anomaly-based intrusion detection systems (IDSs). On IoT-specific hardware, Raspberry Pi is utilized to assess classifier response times.

3. Formulations of dynamic features selection

The literature contains various formulation of the problem of dynamic features selection. It is stated some of them.

3.1. Markov decision process

The work of Xu et al.[22], where the feature selection problem was expressed as a part of a classification problem using Markov decision process notation, is one of the newly presented formulations of dynamic features selection as shown in **Figure 3**. Formally, the sample after feature selection defines the states, and the prediction of the sample's class membership is the action. Depending on the activity chosen, the environment sends back the reward to the agent. The incentive is designed as show in Equation (2).

$$r_t = \begin{cases} r_1 & \text{when prediction is correct} \\ r_2 & \text{when prediction is not correct} \end{cases} \quad (2)$$

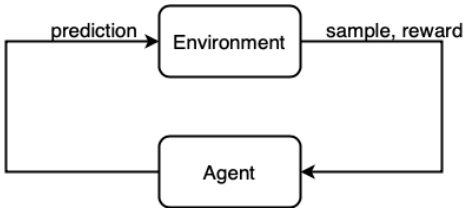

Figure 3. Markov decision process (MDP) for feature selection.

The definition of the state in this formulation is the training sample, and the action is the predicted class of the sample, and the transition probability is embedded in the stream data. Considering that this problem was formulated under MDP, it can be resolved using reinforcement learning where the goal is to build a feature ranking vector that gives the weight of each feature before using the sample in the prediction. For each class of the data, discriminant functions are built using RL. When updating the vectors of the discriminant function, the feature ranking is completed according to the vectors of all the discriminating functions for each class of the data individually.

3.2. Automated feature selection (AutoFS)

An automated feature selection (AutoFS) was suggested in the work of Fan et al.[20], Each feature *f, i* in their formulation has a corresponding agent *agt i* that is in charge of selecting or not selecting the feature based on the environment's condition. Finding an ideal feature subset *F'F* that is most suitable for the predicted accuracy in the downstream task is a challenge when the set of features is $F = f_1, f_2, \ldots f_N$, where *N* is the total number of features. After identifying assertive or reluctant actors, they present a KBest-based trainer that might urge hesitant agents to modify their original behaviour. In this issue formulation, the corresponding agent should switch from deselection to selection if the trainer thinks a hesitant feature is superior to half of the assertive features.

4. Feature drift detection

There are numerous techniques for detecting features drift in the literature. In the ensuing subsections, it lists the most well-known ones.

4.1. Quick reduct and adaptive QuickReduct for feature drift detection

QuickReduct (QR)[22], the reduct with the largest increase in dependency degree, up to its maximum in the dataset under consideration, are concatenated to form the reduct. Because it would be computationally challenging to investigate every potential feature combination, QR decides to act greedily and adds the features that have the biggest impact on the Rough Set dependency degree one at a time to the empty set until no more can be added. Even though it cannot be guaranteed to discover the ideal minimum number of features, this strategy minimizes dataset dimensionality in many real-world settings while maintaining a decent time-performance balance. Adaptive QR (AQR) was put forth in the study of Prasad et al.[23], the fundamental idea of QuickReduct must be used in a dynamic environment, which calls for both the recognition of feature drifts and the accompanying alteration of the selected features. The optimal technique should remove any previously picked features whose contribution has become irrelevant as well as introduce new characteristics that improve the selected reduct's dependency degree when new data are analyzed.

1) Framework for feature drift detection

It is framework to detect abrupt and gradual feature drift and describe the distribution changes of the important features in the data stream[10]. The framework evaluates feature drift based on different known feature importance detection methods, namely, SHAP (SHapley Addictive Explanations), LIME (Local Interpretable Model-agnostic Explanations) and PI (Permutation Importance).

2) Synthetic data generators

It is vital to analyze a learning algorithm's performance across many datasets in order to determine whether it is capable of working in various contexts. Synthetic data stream generators, in contrast to real-world data, are vital and widely utilized because of their versatility, since they allow for a specific description of drift types and places during streams.

4.2. SEA-FD

In the work of Barddal et al.[24], a data stream generator method has been proposed for extending SEA generator that was proposed in (). SEA-FD is capable of simulating streams with $d > 2$ with a uniformly distribution where $\forall D_i \in \mathcal{D}, D_i \in [0; 10]$ and only two randomly selected features are relevant to the concept to be learned. $D^* = \{D_w, D_\xi\}$, and the class y is defined based on θ which is a user defined threshold as show in Equation (3).

$$y = \begin{cases} 1, & \text{if } D_\alpha + D_\beta \leq \theta \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

4.3. BG-FD

Binary generator with feature drift[25], and it has three functions: BG1-FD, BG2-FD, and BG3-FD. For BG1-FD, from the entire set of features D, only a random sub-set $D^* \subset D$ is relevant to the concept to be learned where $|\mathcal{D}^*| = d_r$, d_r is user-given parameter. The class label is given as show in Equation (4).

$$y = \begin{cases} 1, & \text{if } \bigwedge_{D_i \in \mathcal{D}^*} D_i \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

For BG2-FD and BG3-FD, $\mathcal{D}^* = \{D_\alpha, D_\beta, D_\epsilon\}$ is defined and class label for BG2-FD and BG3-FD is given as shown in Equations (5) and (6) respectively.

$$y = \begin{cases} 1, & \text{if } (D_\alpha \wedge D_\beta) \vee (D_\alpha \wedge D_\epsilon) \vee (D_\beta \wedge D_\epsilon) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

$$y = \begin{cases} 1, & \text{if } (D_\alpha \wedge D_\beta \wedge D_\epsilon) \vee (\neg D_\alpha \wedge \neg D_\beta \wedge \neg D_\epsilon) \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

4.4. RTG-FD. the original random tree generator (RTG) builds

It builds a decision tree by randomly performing splits on features and assigning a random class label to each leaf (citation). Instances are created by generating a random valued $\vec{x}$ and traversing the tree for its corresponding label. Barddal et al.[25] has proposed an extension to this generator, namely RTG-FD, such that only a random subset of features $\mathcal{D}^* \subset \mathcal{D}$ are relevant. Assuming $\mathcal{D}_i = \mathcal{D} \setminus \mathcal{D}^*$ as the subset of irrelevant features, $|\mathcal{D}_i|$ is a user-given parameter.

4.4.1. Metrics evaluation

Considering that the problem of dynamic feature selection involves classification, all classification metrics can be calculated.

1) Accuracy:

The accuracy is given by the Equation (7).

$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{7}$$

2) Precision:

Precision denotes the predictive position value and it is given by Equation (8).

$$\text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}} = 1 - \text{FDR} \tag{8}$$

3) Recall:

Recall denotes the true positive ratio and it is given by the Equation (9).

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR} \tag{9}$$

4) G-mean:

Denotes the geometric mean of precision and recall given by the Equation (10).

$$G - \ \mathrm{Mean}\ = \sqrt{\mathrm{precision\ recall}} \tag{10}$$

5) F-measure:

It combines the precision and recall based on the Equation (11).

$$F_1 = 2 \cdot \frac{\mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} \tag{11}$$

5. Taxonomy of feature selection methods

The goal of this section is to review various approaches in the literature for solving the problem of feature selection. It starts with presenting the traditional feature selection methods. Next, it provides a review of the meta-heuristic approaches for feature selection. Next, it presents a literature survey for approaches of solving the online feature selection. A diagram of the categories of the reviewed approaches is given in **Figure 4**.

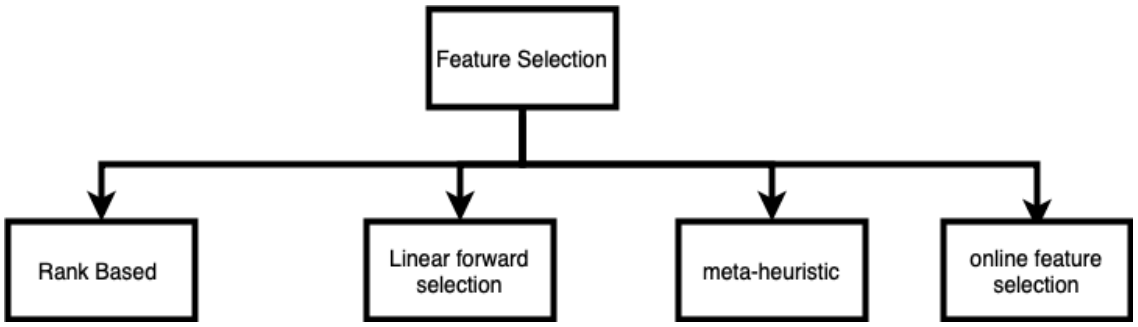

Figure 4. Diagram of the categories of the reviewed approaches.

5.1. Rank based methods

The term "Rank Search" refers to a group of algorithms that can generate a list of attributes that are ranked according to certain criteria such as Bayesian rate error, entropy, symmetric uncertainty, information gain. They all were presented in section 2.3. In the work of Thaseen and Kumar[26], chi-squared has been used for feature ranking. In addition, a process of ranking has been suggested as follows. One input feature is selected from the sample at a time, and the resulting sample is then used for model training and testing. The most significant features are classified using a collection of performance-based rules. The following is a description of the procedure:

1) Excluding one of the data's inputs attributes (training and testing).
2) The classifier is trained and evaluated using the resulting results.
3) The output metrics are used to measure the classifier's results.
4) The rules are used to rate the attribute according to its degree of significance.
5) Repeat steps 1 through 4 for each attribute.

Another example of ranking based methods is the work of Prasad et al.[23]. Their feature selection computed core features and ranked them based on estimated probability.

5.2. Linear forward selection

This approach is indicated in the study of Di Mauro et al.[27]. The goal of their approach is to limit the computational complexity resulted from sequential selection from $O(N^2)$ to $O(k^2)$ where the limit of the

number of features to be selected. It is pre-defined constant according to the available performance. For this step, choosing the top *k*-ranked algorithm based on ranking mechanism as follows:

1) Start with empty set of features.
2) Sequentially add one feature at a time.
3) Repeat the approach for 2 features, 3 and so on.
4) The number of all possible cases $\frac{k(k+1)}{2}$.

5.3. Meta heuristic

The algorithm of meta-heuristic searching provides random searching for the best candidate features with adding some heuristic. They are divided into various classes: swarm algorithm considers the population as swarm and enables mobility of the swarm inside the searching space based on some operators such as the mobility operator of particle swarm optimization. Evolutionary algorithm considers the population as a generation of solutions and perform crossover among elites to generate off-springs until reaching a convergence or pre-defined number of generations. When using meta-heuristic searching for feature selection, the searching is regarded as binary searching as each component of the solution indicates to either selecting the corresponding feature or non-selecting it.

A summary of various meta-heuristic searching algorithms is presented for feature selection in **Table 2**. As it is presented in the table, in the work of Selvakumar and Muneeswaran[28], firefly algorithm was used for feature selection with deployment of wrapper and filter and the classification was based on C4.5 and Bayesian Networks (BN). Similarly, in the work of SaiSindhuTheja and Shyam[29], an integration between crow search algorithm (CSA) and opposition based learning (OBL) was proposed. The classification is done based recurrent neural network (RNN) classifier. A method for anomaly-based detection has been developed in the work of Sarvari et al.[30], using an improved Cuckoo Search Algorithm (CSA) called Mutation Cuckoo Fuzzy (MCF) for feature selection and an Evolutionary Neural Network (ENN) for classification. They use mutation in their search method to more thoroughly scan the search space and enable candidates to escape local minima.

The goal function and Fuzzy C Means (FCM) clustering tool are also utilized to create the fuzzy membership search domain, which comprises all potential compromise solutions. These tools are used to produce the best results for the overlapping dataset. A method for anomaly-based detection has been developed in the work of Raman et al.[31], using an improved Cuckoo Search Algorithm (CSA) called Mutation Cuckoo Fuzzy (MCF) for feature selection and an Evolutionary Neural Network (ENN) for classification. They use mutation in their search method to more thoroughly scan the search space and enable candidates to escape local minima. The goal function and Fuzzy C Means (FCM) clustering tool are also utilized to create the fuzzy membership search domain, which comprises all potential compromise solutions. These tools are used to produce the best results for the overlapping dataset. The MOEA/D (Multi-objective Evolutionary Algorithm based on Decomposition) framework was used in the work of Nguyen et al.[32], to manage feature selection by offering a decomposition approach with two mechanisms (static and dynamic) based on numerous reference points. The static mechanism reduces the decomposition's reliance on the Pareto front shape and the discontinuity's impact. The dynamic one can identify areas where aims are more incompatible, and it devotes more processing power to those areas.

Table 2. Comparing various meta-heuristic-based feature selection.

Article	Algorithm	Improvement	Classification	Dataset	Reached performance	Limitation
[28]	Firefly algorithm	Deploying wrapper and filter	C4.5 and bayesian networks (BN)	KDD CUP 99 dataset	99% with 10 selected features	It does not provide online feature selection
[29]	Crow search algorithm (CSA)	Opposition based learning (OBL)	Recurrent neural network (RNN) classifier	KDD CUP 99	94%	It does not provide online feature selection
[33]	Multi-objective estimation of distribution algorithms (EDA), for (minimizing classification error rate (ER) and minimizing the number of features (NF)	Mutual information (MI) and probabilistic model	Five classification algorithms	NSL-KDD	Up to 97%	Only offline
[30]	modified Cuckoo Search Algorithm (CSA)	Mutation Cuckoo Fuzzy (MCF)	Evolutionary Neural Network (ENN) for classification	NSL-KDD dataset	98.8%	Only off-line
[31]	Hypergraph based genetic algorithm	Kernel parameters with feature selection	Support vector machine	NSL-KDD	95.82	Only offline
[32]	Multi-objective optimization with objective decomposition	Incorporation of objective decomposition and static and dynamic mechanism for searching	KNN with 10-fold cross-validation	Wine australian vehicle german wbcd sonar hill valley musk1 arrhythmia madelon isolet5 multiple features	-	Computational time for repairing duplicated features subsets and revaluating the repaired solutions

5.4. Online feature selection for handling feature drift

In the review of Hu et al.[34], it has been mentioned that traditional feature selection assumes that all candidate features are available before learning starts. However, in many real-world applications, features are generated dynamically, and arrive one by one or group by group. It is hence not practical to wait until all features have been generated before feature selection begins. This poses great challenges to traditional feature selection approaches, called online feature selection with streaming features. This section reviews them. In another review[34], the problem of online feature selection has been reviewed in the literature. The authors have stated some of the challenges such as the generalization to multi-class datasets, the enabling to deal with noisy data and the need of distributed online feature selection approach to handling the computational cost. In the work of Fahy and Yang[35], a dynamic feature mask for clustering high dimensional data streams has been proposed. Redundant features are masked and clustering is performed along unmasked, relevant features. If a feature's perceived importance changes, the mask is updated accordingly; previously unimportant features are unmasked and features which lose relevance become masked. In the work of Li and Cheng[36], an approach based on dynamic sliding windows and feature repulsion loss was proposed. Firstly, within dynamic sliding windows, candidate streaming features that are strongly related to the labels in different feature groups are selected and stored in a fixed sliding window.

Then, the interaction between features is measured by a loss function inspired by the mutual repulsion and attraction between atoms in physics. Specifically, one feature attraction term and two feature repulsion terms are constructed and combined to create the feature repulsion loss function. Finally, for the fixed sliding window, the best feature subset is selected according to this loss function. In the work of Zhou et al.[37], the problem of online streaming feature selection for class imbalance is formulated. In addition, an efficient online

feature selection framework to handle the dependency between condition features and decision classes. Also, the algorithm named as online feature selection based on the Dependency in *K* nearest neighbors was proposed which uses the information of nearest neighbors to select relevant features. In the work of You et al.[38], online learning algorithm named OSFAS was proposed. It uses self-adaption sliding-window and discards the irrelevant and redundant features by conditional independence. In the work of You et al.[39], an algorithm about online streaming feature selection was developed named ConInd that uses a three-layer filtering strategy to process streaming features. Through three-layer filtering, i.e., null-conditional independence, single-conditional independence, and multi-conditional independence, approximate Markov blanket with high accuracy and low running time was obtained.

In the work of Ni et al.[40], an incremental mechanisms of information measure was proposed. In addition, a key instance set containing representative instances to select supplementary features when new instances arrive was proposed. As the key instance set is much smaller than the whole dataset, the proposed incremental feature selection mostly suppresses redundant computations. In the work of Liyanage et al.[41], two algorithms were proposed, namely, ETANA, an on-the-fly fEature selecTion and clAssificatioN and the fast version of ETANA (F-ETANA) for stream data based feature selection. This algorithm does not consider the features dependency between the features. In the work of Wei et al.[42], Dynamic Feature Importance-based Feature Selection (DFIFS), which dynamically selects features according to their Dynamic Feature Importance (DFI) index in the selection process is proposed. DFI is defined by both feature redundancy and feature importance. Further, Gini Importance (GI) of random forest (RF) is used for the feature importance, and Maximum Information Coefficient (MIC) is used for feature redundancy. In the work of Sahmoud and Topcuoglu[43], a framework using a dynamic multi-objective evolutionary algorithm called Dynamic Filter-Based Feature Selection (DFBFS) algorithm was proposed for dynamic feature selection. The framework enables the usability of non-dominated sorting, crowding distance for dynamic feature selection. It incorporates a feature drift detection which is responsible of triggering the dynamic feature selection. In addition, it uses neural network for classification as shown in **Table 3**.

Table 3. Summary table of online feature selection for handling feature drift.

Article	Type	Method or technique	Limitation
[35]	Unsupervised	Masking/unmasking based searching	Concern about efficiency in high dimensional data
[36]	Supervised	Dynamic sliding windows and feature repulsion loss	The manual setting of the threshold of the dynamic sliding window
[44]	Supervised	learning the covariance between feature set and label set	Requires adaptive method and more normalization
[45]	Unsurprised	Local Structure Learning and Sparse Learning (LSS FS)	It does not work for supervised or semi supervised learning
[37]	Supervised	Online feature selection based on the Dependency in *K* nearest neighbours	Limited to binary classification
[38]	Supervised	Self-adaption sliding-window based approach	The computation and the need to adaptively adjust the size of the sliding window
[39]	Supervised	Three-layer filtering strategy to process streaming features	It does not deal with class imbalance
[13]	Supervised	Incremental mechanisms of information measure + key instance set containing representative instances to select supplementary features	Achieve a relatively low performance on datasets with low dimensionality and several instances
[41]	Supervised	ETANA, an on-the-fly fEature selecTion and clAssificatioN And the fast version of ETANA (F-ETANA)	Lacking of exploiting feature dependency

Table 3. (*Continued*).

Article	Type	Method or technique	Limitation
[42]	Supervised	Gini Importance (GI) of random forest (RF) and Maximum Information Coefficient (MIC) for feature redundancy, and it is combined with mRMR	influenced by classifiers and pre-algorithms
[43]	Supervised	Dynamic Filter-Based Feature Selection (DFBFS) based on mutual information and relevance, non-dominated sorting and crowding distance	Fixed space searching non-suitability for high dimensional data

6. Reinforcement learning based approaches

A new solution has recently emerged that uses the reinforcement learning approach to solve the feature selection problem as shown in **Table 4**. An Interactive Reinforced Feature Selection (IRFS) framework is used in the work of Fan et al.[20] to lead agents by using both self-exploration experience and different external professional trainers to speed up feature exploration learning. They specifically model two trainers adept at various searching techniques after framing the feature selection problem into an interactive reinforcement learning framework: (1) KBest based trainer and (2) Decision Tree based trainer. Then, in order to diversity agent training, two tactics were developed: (1) to identify forceful and reluctant agents; and (2) to allow the two trainers to take on the teaching role at various points in order to combine their experiences and diversify the teaching process. Interactive and closed-loop modeling of interactive reinforcement learning (IRL) and decision tree feedback is presented in the study of Fan et al.[46] (DTF).

In particular, IRL will develop an interactive feature selection loop, while DTF will feed the loop structured feature information. The DTF enhances IRL in two ways. First, state representation is enhanced by using the decision tree-generated tree-structured feature hierarchy. They specifically depict the chosen feature subset as a directed tree of decision features and an undirected graph of feature-feature correlations. They suggest a brand-new embedding technique that enables the Graph Convolutional Network (GCN) to concurrently learn state representation from the graph and the tree. Second, a novel incentive system is created by taking use of the tree-structured feature hierarchy. They specifically tailor incentive distribution for agents based on the significance of decision tree features. They also create a new reward scheme to weigh and distribute rewards depending on the chosen frequency ratio of each agent in past action records because watching agents in action can also provide feedback. In the work of Fan et al.[47], Group-based Interactive Reinforced Feature Selection (GIRFS) framework has been proposed. GIRFS framework balances single-agent RFS and multi-agent RFS for better feature selection. Specifically, based on formulating the feature selection problem into a group-based RFS problem. In this formulation, the given features were assigned into several groups based on feature similarity measurement. Then, agents for each group were created, where each agent decides to select/deselect features in its corresponding group. Moreover, to further improve learning efficiency, a hierarchical teacher-like trainer to provide external action advice for agents was proposed. This trainer provides advice by intra-group selection and inter-group selection and fuses knowledge from mRMR and decision tree to help agents explore and learn. In the work of Liu et al.[48], the feature selection problem was reformulated with the combinatorial multi-armed bandit (CMAB) framework by regarding each feature as an arm. Two novel oracles were proposed and how the super arm is formed under different oracles were investigated, and how the coordination between various features can be improved by a novel reward scheme. Two methods were proposed: (1) Combinatorial Multi-Armed Bandit Generative Feature Selection (CMAB-GFS) and (2) Combinatorial Upper Confidence Bounds Based Feature Selection (CUCB-FS).

Table 4. Overview of the reinforcement learning based on methods.

Article	Agents	State	Action	Reward	Limitation
[21]	Multi agent and two trainers	Graph Convolutional Network	Selection vs. deselection advice	Equal distribution of predictive accuracy	Reward distribution is not fair
[46]	Multi agent and two trainers	Utilizing a graph convolutional network, both the graph and the tree are used to simultaneously learn state representation.	Selection vs. deselection advice	Predictive accuracy and feature correlation new personalized reward scheme to better measure agent reward assignment.	Concern about efficiency
[47]	Multi agent and two trainers using one agent for each group of features	Utilizing a graph convolutional network, both the graph and the tree are used to simultaneously learn state representation.	Selection vs. deselection advice	Predictive accuracy and feature correlation new personalized reward scheme to better measure agent reward assignment	Complex architecture

7. Open challenges and research direction

This section provides the open challenges and future research directions.

7.1. Balancing the effectiveness and efficiency of automated feature selection

Recent investigations have noted a computational problem in feature selection[46]:

1) While conventional selection techniques are typically effective, it might be difficult to pinpoint the optimal subset.

2) New reinforced selection techniques investigate the optimal subset by automatically navigating across feature space, although they are often ineffective. Do automation and effectiveness have to be mutually exclusive. In the face of automation, is it feasible to strike a compromise between efficacy and efficiency.

7.2. Concept drift understanding

All drift detection techniques can answer the question “When”, but very few techniques can also respond to the questions “How” and “Where”. Future research will focus on creating models to address this challenge[1].

7.3. Adaptive models

Recent breakthroughs in idea drift adaptation have seen a rise in the importance of adaptive models and ensemble methodologies. The development of retraining models that explicitly detect drift, however, has become a critical and promising area of research in addressing the evolving nature of data streams and maintaining model performance over time[1].

7.4. True label is available after

The majority of drift detection and adaptation algorithms in use today anticipate that verification will take a long time or that the ground truth label will be accessible after classification or prediction. The topic of unsupervised or semi-supervised drift detection and adaptation has received very little attention[1].

7.5. Real-world data streams from the concept drift aspect

Real-world data streams from the concept drift aspect, such as the drift occurrence time, the severity of drift, and the drift zones, have not been thoroughly analyzed[1].

8. Conclusion

A survey on feature selection methods for stream data with dynamical changes or feature drift is presented in this article. After providing the fundamentals and important concepts for feature and concept drift. Next, the methods for feature drift detection and models for generating stream data with feature drift were provided. In

addition, the article provided the evaluation metrics for dynamic feature selection. The article provided a taxonomy for feature selection. After reviewing various approaches, it gives a more focus on reinforcement learning based models. Lastly, the article presents the various challenges in this field, namely, balancing effectiveness and efficiency, concept drift understanding, adaptive models, true label availability, and real-world data streams from concept drift. This survey enables researchers to have an updated view of the state of the art of IDS from machine learning methods and the most recently issues that are under the focus of the literature.

Author contributions

Conceptualization, AJR and MZAN; methodology, AJR; software, AJR; validation, MZAN, AS and AAJ; formal analysis, AJR; investigation, MZAN; resources, AJR; data curation, AJR; writing—original draft preparation, AJR; writing—review and editing, MZAN; visualization, AJR; supervision, AS; project administration, MZAN; funding acquisition, MZAN. All authors have read and agreed to the published version of the manuscript.

Conflict of interest

The authors declare no conflict of interest.

References

1. Lu J, Liu A, Dong F, et al. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering* 2019; 31(12): 2346–2363. doi: 10.1109/TKDE.2018.2876857
2. Iwashita AS, Papa JP. An overview on concept drift learning. *IEEE Access* 2019; 7: 1532–1547. doi: 10.1109/ACCESS.2018.2886026
3. Gama J, Žliobaitė I, Bifet A, et al. A survey on concept drift adaptation. *Acm Computing Surveys* 2014; 46(4): 1–37. doi: 10.1145/2523813
4. Yang Z, Al-Dahidi S, Baraldi P, et al. A novel concept drift detection method for incremental learning in nonstationary environments. *IEEE Transactions on Neural Networks and Learning Systems* 2020; 31(1): 309–320. doi: 10.1109/TNNLS.2019.2900956
5. Kunlin Y. A memory-enhanced framework for financial fraud detection. In: Proceedings of 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA); 17–20 December 2018; Orlando, FL, USA. pp. 871–874.
6. Somasundaram A, Reddy S. Parallel and incremental credit card fraud detection model to handle concept drift and data imbalance. *Neural Computing Applications* 2019; 31(1): 3–14. doi: 10.1007/s00521-018-3633-8
7. Brzezinski D, Minku LL, Pewinski T, et al. The impact of data difficulty factors on classification of imbalanced and concept drifting data streams. *Knowledge Information Systems* 2021; 63(6): 1429–1469. doi: 10.1007/s10115-021-01560-w
8. Halstead B, Koh YS, Riddle P, et al. Recurring concept memory management in data streams: Exploiting data stream concept evolution to improve performance and transparency. *Data Mining Knowledge Discovery* 2021; 35(3): 796–836. doi: 10.1007/s10618-021-00736-w
9. Al-Khaleefa AS, Ahmad MR, Isa AAM, et al. Infinite-term memory classifier for Wi-Fi localization based on dynamic Wi-Fi simulator. *IEEE Access* 2018; 6: 54769–54785. doi: 10.1109/ACCESS.2018.2870754
10. Zhao D, Koh YS. Feature drift detection in evolving data streams. In: Hartmann S, Küng J, Kotsis G, et al. (editors). *Database and Expert Systems Applications*, Proceedings of 31st International Conference, DEXA 2020; 14–17 September 2020; Bratislava, Slovakia. Springer Cham; 2020. pp. 335–349.
11. Khamassi I, Sayed-Mouchaweh M, Hammami M, Ghédira K. Discussion and review on evolving data streams and concept drift adapting. *Evolving Systems* 2018; 9(1): 1–23. doi: 10.1007/s12530-016-9168-2
12. de Barros RSM, de Carvalho Santos SGT. An overview and comprehensive comparison of ensembles for concept drift. *Information Fusion* 2019; 52: 213–244. doi: 10.1016/j.inffus.2019.03.006
13. Al-Jarrah OY, Maple C, Dianati M, et al. Intrusion detection systems for intra-vehicle networks: A review. *IEEE Access* 2019; 7: 21266–21289. doi: 10.1109/ACCESS.2019.2894183
14. Žliobaitė I. Learning under concept drift: An overview. *arXiv* 2010; arXiv:1010.4784. doi: 10.48550/arXiv.1010.4784
15. Schlimmer JC, Granger RH. Incremental learning from noisy data. *Machine Learning* 1986; 1(3): 317–354. doi: 10.1007/BF00116895

16. Al-Khaleefa AS, Ahmad MR, Esa AAM, et al. Knowledge preserving OSELM model for Wi-Fi-based indoor localization. *Sensors* 2019; 19(10): 2397. doi: 10.3390/s19102397
17. Halstead B, Koh YS, Riddle P, et al. Analyzing and repairing concept drift adaptation in data stream classification. *Machine Learning* 2022; 111: 3489–3523. doi: 10.1007/s10994-021-05993-w
18. Al-Jumaily A, Sali A, Jiménez VPG, et al. Evaluation of 5G coexistence and interference signals in the C-band satellite earth station. *IEEE Transactions on Vehicular Technology* 2022; 71(6): 6189–6200. doi: 10.1109/TVT.2022.3158344
19. D'hooge L, Wauters T, Volckaert B, De Turck F. Inter-dataset generalization strength of supervised machine learning methods for intrusion detection. *Journal of Information Security Applications* 2020; 54: 102564. doi: 10.1016/j.jisa.2020.102564
20. Fan W, Liu K, Liu H, et al. Autofs: Automated feature selection via diversity-aware interactive reinforcement learning. *arXiv* 2020; arXiv:2008.12001. doi: 10.48550/arXiv.2008.12001
21. Velayutham C, Thangavel K. Unsupervised quick reduct algorithm using rough set theory. *Journal of electronic science technology* 2011; 9(3): 193–201.
22. Xu R, Li M, Yang Z, et al. Dynamic feature selection algorithm based on Q-learning mechanism. *Applied Intelligence* 2021; 51: 7233–7244. doi: 10.1007/s10489-021-02257-x
23. Prasad M, Tripathi S, Dahal K. An efficient feature selection based bayesian and rough set approach for intrusion detection. *Applied Soft Computing* 2020; 87: 105980. doi: 10.1016/j.asoc.2019.105980
24. Barddal JP, Gomes HM, Enembreck F. Analyzing the impact of feature drifts in streaming learning. In: Arik S, Huang T, Lai W, et al. (editors). *Neural Information Processing International*, Proceedings of 22nd International Conference, ICONIP 2015; 9–12 November 2015; Istanbul, Turkey. Springer Cham; 2015. pp. 21–28.
25. Barddal JP, Gomes HM, de Souza Britto A, Enembreck F. A benchmark of classifiers on feature drifting data streams. In: Proceedings of 2016 23rd International Conference on Pattern Recognition (ICPR); 4–8 December 2016; Cancun, Mexico. pp. 2180–2185.
26. Thaseen IS, Kumar CA. Intrusion detection model using fusion of chi-square feature selection and multi class SVM. *Journal of King Saud University-Computer Information Sciences* 2017; 29(4): 462–472. doi: 10.1016/j.jksuci.2015.12.004
27. Di Mauro M, Galatro G, Fortino G, Liotta A. Supervised feature selection techniques in network intrusion detection: A critical review. *Engineering Applications of Artificial Intelligence* 2021; 101: 104216. doi: 10.1016/j.engappai.2021.104216
28. Selvakumar B, Muneeswaran K. Firefly algorithm based feature selection for network intrusion detection. *Computers Security* 2019; 81: 148–155. doi: 10.1016/j.cose.2018.11.005
29. SaiSindhuTheja R, Shyam GK. An efficient metaheuristic algorithm based feature selection and recurrent neural network for DoS attack detection in cloud computing environment. *Applied Soft Computing* 2021; 100: 106997. doi: 10.1016/j.asoc.2020.106997
30. Sarvari S, Sani NFM, Hanapi ZM, Abdullah MT. An efficient anomaly intrusion detection method with feature selection and evolutionary neural network. *IEEE Access* 2020; 8: 70651–70663. doi: 10.1109/ACCESS.2020.2986217
31. Raman MRG, Somu N, Kirthivasan K, et al. An efficient intrusion detection system based on hypergraph-genetic algorithm for parameter optimization and feature selection in support vector machine. *Knowledge-Based Systems* 2017; 134: 1–12. doi: 10.1016/j.knosys.2017.07.005
32. Nguyen BH, Xue B, Andreae P, et al. Multiple reference points-based decomposition for multiobjective feature selection in classification: Static and dynamic mechanisms. *IEEE Transactions on Evolutionary Computation* 2020; 24(1): 170–184. doi: 10.1109/TEVC.2019.2913831
33. Maza S, Touahria M. Feature selection for intrusion detection using new multi-objective estimation of distribution algorithms. *Applied Intelligence* 2019; 49(2): 4237–4257. doi: 10.1007/s10489-019-01503-7
34. Hu X, Zhou P, Li P, et al. A survey on online feature selection with streaming features. *Frontiers of Computer Science* 2018; 12(3): 479–493. doi: 10.1007/s11704-016-5489-3
35. Fahy C, Yang S. Dynamic feature selection for clustering high dimensional data streams. *IEEE Access* 2019; 7: 127128–127140. doi: 10.1109/ACCESS.2019.2932308
36. Li Y, Cheng Y. Streaming feature selection for multi-label data with dynamic sliding windows and feature repulsion loss. *Entropy* 2019; 21(12): 1151. doi: 10.3390/e21121151
37. Zhou P, Hu X, Li P, Wu X. Online feature selection for high-dimensional class-imbalanced data. *Knowledge-Based Systems* 2017; 136: 187–199. doi: 10.1016/j.knosys.2017.09.006
38. You D, Wu X, Shen L, et al. Online feature selection for streaming features using self-adaption sliding-window sampling. *IEEE Access* 2019; 7: 16088–16100. doi: 10.1109/ACCESS.2019.2894121
39. You D, Wu X, Shen L, et al. Online streaming feature selection via conditional independence. *Applied Sciences* 2018; 8(12): 2548. doi: 10.3390/app8122548
40. Ni P, Zhao S, Wang X, et al. Incremental feature selection based on fuzzy rough sets. *Information Sciences* 2020; 536: 185–204. doi: 10.1016/j.ins.2020.04.038
41. Liyanage YW, Zois DS, Chelmis C. On-the-fly joint feature selection and classification. Available online: https://arxiv.org/abs/2004.10245 (accessed on 14 July 2023).

42. Wei G, Zhao J, Feng Y, et al. A novel hybrid feature selection method based on dynamic feature importance. *Applied Soft Computing* 2020; 93: 106337. doi: 10.1016/j.asoc.2020.106337
43. Sahmoud S, Topcuoglu HR. A general framework based on dynamic multi-objective evolutionary algorithms for handling feature drifts on data streams. *Future Generation Computer Systems* 2020; 102: 42–52. doi: 10.1016/j.future.2019.07.069
44. Wang Z, Wang T, Wan B, Han M. Partial classifier chains with feature selection by exploiting label correlation in Multi-label classification. *Entropy* 2020; 22(10): 1143. doi: 10.3390/e22101143
45. Lei C, Zhu X. Unsupervised feature selection via local structure learning and sparse learning. *Multimedia Tools Applications* 2018; 77(22): 29605–29622. doi: 10.1007/s11042-017-5381-7
46. Fan W, Liu K, Liu H, et al. Interactive reinforcement learning for feature selection with decision tree in the loop. *IEEE Transactions on Knowledge Data Engineering* 2023; 35(2): 1624–1636. doi: 10.1109/TKDE.2021.3102120
47. Fan W, Liu K, Liu H, et al. AutoGFS: Automated group-based feature selection via interactive reinforcement learning. In: Proceedings of the 2021 SIAM International Conference on Data Mining (SDM); 2021. pp. 342–350.
48. Liu K, Huang H, Zhang W, et al. Multi-armed bandit based feature selection. In: Proceedings of the 2021 SIAM International Conference on Data Mining (SDM); 2021. pp. 316–323.