A Survey of Data Stream Ensemble Classification Algorithms: Postprint

Authors: Xu Guanying, Han Meng, Wang Shaofeng, Jia Tao

Date: 2018-11-29T00:00:00+00:00

Abstract

Currently, ensemble classification algorithms constitute the mainstream paradigm in data stream classification, as they deliver superior performance and enhanced capabilities compared to single classifiers. Furthermore, they are readily deployable in real-world applications, exhibit rapid adaptability and resilience to concept drift, and achieve optimal classification performance in handling class imbalance problems. This paper provides a comprehensive survey of ensemble classification algorithms from both domestic and international perspectives, offering a detailed review of the two fundamental components: base classifier combination and dynamic ensemble model updating. It distinctly characterizes the strengths and weaknesses of various ensemble algorithms and provides comparative analyses of algorithms and experimental datasets. Additionally, it proposes future research directions and potential solutions under consideration.

Full Text

Preamble

Vol. 37 No. 1 Application Research of Computers ChinaXiv Partner Journal

Summarization of Data Stream Ensemble Classification Algorithms

Xu Guanying, Han Meng, Wang Shaofeng, Jia Tao

(School of Computer Science & Engineering, North University for Nationalities, Yinchuan 750021, China)

Abstract: Currently, the trend in data stream classification algorithms is moving toward ensemble classification algorithms, as they provide better performance and more outstanding results than single classification algorithms. They are also easy to deploy in real-world applications, offer rapid adaptability and recovery from concept drift, and deliver optimal classification performance when

handling class imbalance problems. This paper provides a detailed introduction to domestic and international ensemble classification algorithms, comprehensively reviewing the two core components of ensemble classification algorithms (base classifier combination and dynamic ensemble model updating). It clearly distinguishes the advantages and disadvantages of different ensemble algorithms, compares algorithms and experimental datasets, and proposes directions for further research and potential solutions.

Keywords: data stream classification; ensemble learning; concept drift

CLC Number: TP301.6 **doi:** 10.19734/j.issn.1001-3695.2018.09.0510

0 Introduction

In recent years, with the rapid development of big data, vast amounts of useful information are embedded in these data. To extract this information, researchers have undertaken numerous data mining tasks. Recently, the field of data stream mining has made significant progress in obtaining useful models from large volumes of rapidly generated data. Data streams present several challenges for learning algorithms. Ensemble learners have been extensively studied and deployed for real-world problems, with scholars offering three justifications for using ensembles rather than single learners: statistical, computational, and representational reasons. Another explanation for this preference is the difficulty in obtaining strong learners, whereas a group of weak learners can be relatively easily developed and effectively boosted into strong learners through strategic training and combination. Ensemble learners are particularly welcome in data stream settings because, beyond leveraging weak learners, they can also address general machine learning problems and specific data stream challenges. For instance, ensemble learners have been widely applied to solve problems such as concept drift in data streams, recurring concepts, and novel class detection, demonstrating superior performance compared to single classification models in these scenarios.

Compared with traditional static data, data streams are characterized by real-time, high-efficiency, rapid arrival, and the constraint that arriving instances can only be processed once. Consequently, mining tasks on data stream data face the following challenges: (a) data in streams can only be processed once, and the flowing data cannot be stored in data warehouses; (b) processing results can only be approximated to the greatest extent possible; and (c) the distribution of data in the stream changes over time, a phenomenon known as concept drift. Therefore, algorithms designed for stream processing must possess rapid recovery, adaptability, accuracy, and robustness, enabling real-time updates to handle subsequent changes in the stream' s distribution. Among algorithms for processing stream data, classification is the most important and critical component of data stream mining. While static data processing methods are relatively mature, traditional classification methods can no longer satisfy stream mining

tasks. For traditional mining algorithms, mining tasks become impossible on data streams experiencing concept drift, making algorithms specifically designed for stream data processing particularly crucial.

1.3.1 Single Classification Model

Single classifier models continuously update their own structure recursively using newly arriving data, enabling the structure to adapt to changes in the stream data and accurately classify instances in the stream. The primary fundamental techniques for single classification models include KNN, decision trees, SVM, Bayesian methods, logistic regression, and neural networks.

KNN finds the K points in the training set sample space that are closest to the prediction sample x, counts the categories of these K nearest points, and assigns x to the category with the highest count. Advantages include simple principles, mature theory, applicability to both classification and regression, and suitability for nonlinear classification with a training time complexity of O(n). Disadvantages include high computational cost and difficulty in handling class imbalance problems (where some categories have many samples while others have very few).

Decision Tree methods employ a top-down recursive approach, comparing attribute values at internal nodes and inferring downward branches based on different attribute values, with conclusions (predictions) obtained at leaf nodes. A decision tree is a flowchart-like tree structure where each internal node represents a test on an attribute, each branch represents a test output, and each leaf node represents a class or class distribution. Advantages include simple computation, strong interpretability, suitability for handling samples with missing attribute values, and ability to handle irrelevant features. The primary disadvantage is a tendency toward overfitting.

SVM (Support Vector Machine) is a binary classification model whose main ideas can be summarized in two points: (a) it analyzes linearly separable cases, and for linearly inseparable cases, uses nonlinear mapping algorithms to transform low-dimensional input space samples that are linearly inseparable into high-dimensional feature spaces where they become linearly separable, thereby enabling linear algorithms in high-dimensional feature spaces to perform linear analysis of nonlinear features; and (b) it is based on structural risk minimization theory, constructing an optimal hyperplane in feature space that yields a global optimal solution for the learner, with the expectation over the entire sample space satisfying a certain upper bound with some probability. Advantages include applicability to both linear/nonlinear classification and regression, low generalization error, and easy interpretability with low computational complexity. Disadvantages include sensitivity to parameter and kernel function selection.

Bayesian Classifiers perform classification through an object' s prior probability, using Bayes' formula to calculate its posterior probability—the probability

that the object belongs to a certain class—and then selecting the class with the maximum posterior probability as the object' s class. In other words, the class with the larger posterior probability is chosen. Advantages include good performance on small-scale data, suitability for multi-classification tasks, and suitability for incremental training. Disadvantages include sensitivity to the representation form of input data (particularly the handling of continuous data).

Logistic Regression is used for regression problems where the dependent variable is categorical, commonly for binary classification or binomial distribution problems. The relationship graph between probability and independent variables for binary classification problems is often an S-shaped curve implemented using the Sigmoid function. Advantages include simple implementation, minimal computational cost during classification, fast speed, and low storage resource requirements. Disadvantages include susceptibility to underfitting, generally low accuracy, and limitation to binary classification problems.

Neural Networks consist of a set of interconnected input-output neural units, where each connection between units is associated with a weight. During the network learning phase, the network adjusts weights to achieve correspondence between input samples and their correct categories. Since neural network learning primarily focuses on connection weights, it is sometimes called connectionist learning. Advantages include strong nonlinear fitting capabilities, ability to map any complex nonlinear relationship, and simple learning rules that are easy to implement computationally. Disadvantages include inability to pose necessary queries to users, inability to function when data is insufficient, and the fact that converting all problem features into numbers and all reasoning into numerical calculations inevitably results in information loss.

Single model structures are complex with limited expressive capability, but they offer good stability and high plasticity, performing well even on stream data experiencing concept drift. For example, the earliest proposed VFDT trains decision trees using small amounts of data that satisfy the Hoeffding bound, achieving classification results similar to those trained on large amounts of data that do not satisfy the Hoeffding bound. Due to limitations at the time, concept drift phenomena were not considered, though many current ensemble algorithms still retain VFDT as a training algorithm, demonstrating VFDT' s relatively outstanding single-classification performance. Building upon VFDT, CVFDT was proposed to address concept drift problems in data streams.

1.3.2 Ensemble Classification Model

In stationary data stream scenarios, training data is first divided into different subsets. On each subset, a learning algorithm is applied to learn from the data, generating a base learner (base classifier) for each subset. These multiple base learners are then combined into an ensemble learner (ensemble classifier) through some combination method. When predicting the class label of an instance, the ensemble classifier synthesizes the results from each base classifier

through some mechanism and outputs the final result to obtain the class label of the unknown instance (prediction). Since ensemble learning combines multiple learners, we must consider how to make the ensemble learner demonstrate better learning performance than a single learner. Two approaches have emerged to solve this problem: to obtain a good ensemble, individual learners should be "good but different" —meaning they must have a certain level of accuracy (not too poor) and diversity (differences among learners). To achieve accuracy improvement, base classifiers in the ensemble must have a certain degree of dissimilarity. This can be achieved by training base classifiers on different data or even using different base classifier algorithms.

Consider a binary classification problem with y {-1, +1} and true function f. Assuming the base classifier error rate is , for each base classifier h_i we have:

$$P(\varepsilon_i) = P(h_i(x) \neq f(x)) = \varepsilon$$

Assuming the combination of T classifiers through simple majority voting, if more than half of the base classifiers are correct, the ensemble classification is correct:

$$H(x) = \text{sign}\left(\sum_{i=1}^{T} h_i(x)\right)$$

Assuming base classifier errors are independent, by the Hoeffding inequality, the ensemble error rate is:

$$P(H(x) \neq f(x)) = \sum_{k=0}^{\lfloor T/2 \rfloor} \binom{T}{k} (1-\varepsilon)^k \varepsilon^{T-k} \leq \exp\left(-\frac{1}{2} T(1-2\varepsilon)^2\right)$$

This inequality clearly shows that as the number of base classifiers T in the ensemble model increases, the ensemble model' s error rate decreases exponentially, eventually approaching zero. Ensemble learning simply performs majority voting on the results of all individual learners, thereby achieving better generalization performance than individual learners. However, this requires a key assumption: base learner errors must be independent. In real tasks, individual learners are trained to solve the same problem and obviously cannot be independent. In fact, there is an inherent conflict between the accuracy and diversity of individual learners. Generally, after achieving high accuracy, increasing diversity requires sacrificing accuracy.

2 Concept Drift Issues in Ensemble Classifiers

2.1 Definition of Concept Drift

When performing stream data mining tasks, target concepts undergo dramatic changes over time and with the surrounding environment. Even stable concepts can change dramatically. For example, users' browsing tendencies on websites are influenced by real-time hot news and changes in personal preferences. This phenomenon where target concepts change due to nondeterministic factors is called concept drift. One classic definition of concept defines it as a set of objects, but this definition cannot be applied to data streams. Most literature on concept drift currently defines it using prior probability, conditional probability, and posterior probability. Reference [20] analyzes three types of concept drift:

a) The prior probability P(c) of a class changes over time.
b) The conditional probability $P(X|c)$ of one or several classes may change over time.
c) Changes in posterior probability $P(c|X)$ are considered true concept drift, meaning the same instance has different class labels in different time domains.

2.2 Types of Concept Drift

Based on prior probability, conditional probability, and posterior probability of class labels, concept drift is mainly divided into virtual concept drift and real concept drift [21]. The former does not affect decision boundaries (posterior probability) but affects conditional probability density functions, and therefore should not directly affect the classifier being used. The latter affects decision boundaries (or posterior probability) and may affect conditional probability density functions, potentially significantly impacting classifier performance. [Figure 1: see original paper] illustrates the different boundaries of these two drift types [22].

2.3 Concept Drift Handling Techniques

Current approaches to handling concept drift include sliding window models, concept drift detectors, ensemble learning models, online learners, etc. [23~25].

a) **Sliding Window**: Sliding window technology primarily maintains a buffer where the most recent instances are considered to best reflect the current data distribution in the stream. These instances are used to train and update models, and once new instances arrive, previous instances are discarded. This technique provides a way to analyze only the most recent data tuples in the data stream without requiring random sampling or retaining statistical information about outdated data [26,27]. Representative algorithms include ADWIN Bagging [28], Leveraging Bagging [29], ADWIN2, SERDRIFT [30,31], and ECISD [32].

b) **Concept Drift Detectors**: These can be viewed as external algorithms combined with a given classifier. Their purpose is to monitor specific properties of the data stream, such as standard deviation [33], prediction error [34], or instance distribution [35]. Any change in these features is assumed to be caused by the presence of drift. Therefore, by measuring the level of change, detectors can report detected changes. Representative algorithms include DDM [36] and EDDM [37].

c) **Online Learners**: These update models by processing instances in an online manner, thereby adjusting to the stream as quickly as possible when it occurs. Such learners must satisfy a series of requirements [38]: each object must be processed only once during training; the computational complexity of processing each instance must be minimized; and their accuracy should not be lower than that of classifiers trained on batch data. Representative algorithms include CUSUM [39] and FIMT-DD [40].

d) **Ensemble Learners**: Ensemble learners using combination methods can easily adapt to stream changes due to their diversity and complex structure, where each single classifier has good performance. They provide flexibility and gains in predictive capability [41]. Two main approaches assume either a changing ensemble [42] or updating base classifiers [43]. New classifiers are trained on recently arrived data (usually collected in blocks) and added to the ensemble model. Pruning is used to control the number of base classifiers and remove the worst-performing or oldest models. Representative algorithms include DWM [44], AWE [45], SEA [46], EB [47], OCBOOST [48], and OAUE [49].

e) **Other Techniques**: In reference [50], the authors utilize the range of Kappa coefficients for detection, considering a Kappa coefficient of 65% as acceptable harmony. When the Kappa coefficient for the last 100 samples of each input package falls below 65%, the proposed method calls the classification process "random." When this occurs, the weighting function is replaced and poor classifiers are discarded. Representative algorithms include reference [50] and ASHT [51].

3 Ensemble Classification Algorithms

Ensemble classification learning is a machine learning technique that integrates multiple base classifiers for joint decision-making. By invoking simple or complex incremental learning algorithms, multiple high-performance and diverse base classifiers are obtained and then combined into an ensemble classifier through some integration method. Section 1.3.2 introduced the theory and development direction of ensembles, so how to generate and combine good but different individual learners is the core of ensemble learning research. The key questions are: how to integrate? What kind of individual learners to integrate? Based on the generation method of individual learners, current ensemble learning methods can be roughly divided into two categories [52]: (a) sequential

methods where individual learners have strong dependencies and must be generated serially, represented by Boosting [53]; (b) parallel methods where individual learners have no strong dependencies and can be generated simultaneously, represented by Bagging [53] and Random Forest [54].

For ensemble learners, different benchmark algorithms are needed for different problems. Although the essential goal is to pursue good classification performance, selecting appropriate base learners based on specific classification problems is a necessary prerequisite for obtaining accurate ensemble classifiers. Classifiers can naturally handle only one type of feature domain without resorting to input preprocessing. Therefore, assuming all features have the same domain, base learners can be selected based on the input feature domain; for example, using base learners that handle discrete and continuous features separately. A widely used example is the Hoeffding tree, as the Hoeffding bound determination requires only a small amount of data to train a tree that approximates one trained on all data, with good classification performance in experiments. Algorithms using Hoeffding tree as the base algorithm include ASHT, HWT [55], and AWT-ADWIN [56]. In addressing data stream problems, CVFDT is a fast decision tree benchmark algorithm that can handle concept drift, such as the CVFDT Update Ensemble (CUE) [57] algorithm. Other base learners commonly used for ensemble stream learning include Naive Bayes, Support Vector Machines, and Multilayer Perceptrons.

Ensemble classification models are divided into two parts: base classifier combination and dynamic ensemble model updating.

3.1 Combination of Base Classifiers

The overall prediction of the ensemble should outperform that of a single classifier. This section seeks an appropriate method to combine these base classifiers to better distinguish difficult-to-classify classes. Previously, scholars focused on developing more accurate single classification models without studying classifier combination predictions [58]. Among combination methods, voting and fixed base classifier ensembles are most commonly used.

Voting methods are approaches for selecting the output results of individual classifiers during ensemble model prediction. Current main voting methods are divided into majority voting, weighted voting, and other voting methods.

a) **Majority Voting**: Majority voting initializes all base classifiers with the same weight. During final prediction, if the votes for a particular label from base classifiers exceed half, or if the classifier with the most votes is custom-defined, it is identified as the final prediction. If multiple base classifiers tie for the highest number of votes, one is randomly selected. Data stream ensemble classification algorithms using majority voting include online Bagging and Boosting, MOSOB [59][60], OOB and UOB [60], etc. The main idea of online algorithms is that data does not arrive in blocks but as individual instances in the data stream, with the learning algorithm

processing each instance online. As the data volume N approaches infinity, it satisfies a Poisson distribution with =1.

b) **Weighted Voting**: Due to performance differences among base classifiers, weights are assigned to base classifiers based on their performance in the ensemble, thereby emphasizing those with good classification performance to ensure correct overall predictions. Simple weighting is based on classifier accuracy. For example, AWE (Accuracy Weighted Ensemble) is a benchmark algorithm for data streams that assigns weights inversely proportional to the mean squared error of base classifiers, using multi-fold cross-validation to calculate weights, replacing weaker base classifiers with stronger ones during model updates. Weighted Majority (WM) and Matrix Multiplicative Weights (MMW) [61] algorithms weight classifier predictions based on past performance, where each classifier has a weight that decreases whenever it makes an incorrect prediction. Accuracy Update Ensemble (AUE) calculates errors of classifiers on the latest data block and all base classifiers in the ensemble on the latest data block, comparing their performance. If any base classifier in the ensemble has larger error than the classifier on the latest data block, the worst base classifier is replaced.

More complex weighting includes Online Accuracy Updated Ensemble (OAUE), which differs from traditional block-based approaches by using both block-based and incremental learning algorithms. In the AUE weighting formula, the concept of time is introduced, transforming the weighting formula into an incremental expression that flows with time. CVFDT Update Ensemble (CUE) uses VFDT as the algorithm for training base classifiers. During model updating, the idea is that the mean squared error of random guessing for any base classifier on the latest data block is $rMSE = \sum_y (p(y) - p(y))^2$. If a base classifier' s mean squared error $MSE_i > rMSE$, it indicates the base classifier' s accuracy is lower than random guessing, making it non-contributive to the model. Such base classifiers need to be updated with data from the latest data block. To increase diversity among base classifiers, bagging operations are performed on data in the block during the update process. Since the training data for base classifiers differs significantly, diversity among them is increased.

ECISD (Ensemble Classifiers for Imbalanced Data Stream) is a weighted ensemble algorithm for handling class imbalance that uses AUE2 [62] as its benchmark algorithm. As a class imbalance ensemble classification algorithm, it first combines SMOTE oversampling and Tomek-links undersampling to sample the original data. In weighting base classifiers, the concept of cost is introduced. During model updating, the worst-performing base classifiers are eliminated based on their contribution to model accuracy. Other weighted algorithms mainly include Adaptive Classifiers-Ensemble (ACE) [63] and Weighted Ensemble Online Bagging (WEOB) [64].

c) **Other Voting Methods**: Considering the defects of majority voting,

authors [65] proposed a new voting method where not all base classifiers participate in voting. Instead, an abstention threshold is set at 0.65. If a base classifier' s accuracy is below 0.65, it does not participate in the decision phase; only base classifiers with accuracy above 0.65 vote. Dynamic Weighted Majority (DWM) maintains a variable number of base classifiers, with ensemble predictions determined by weighted majority votes. Each classifier' s weight increases with correct predictions and decreases otherwise. If a classifier's weight falls below a given threshold, it is removed from the ensemble. If the ensemble decision is incorrect, a new expert is added. Since this ensemble update strategy can generate excessive additions and deletions for noisy data streams, the authors introduced a parameter to determine how many instances will undergo overall updating.

Modal Mixture Model (M3) [66] is a weighted majority ensemble algorithm based on heterogeneous model types, where model weights are updated online using reinforcement learning techniques. Because it uses a mixture of base classifiers and adjusts ensemble member weights through reinforcement learning that borrows from online concept-based approaches, M3 demonstrates strong performance. When data points from the data stream enter the application, they are initially used as test data to evaluate the overall algorithm from experimental reports. Subsequently, each data point (selected through uniform random selection) can be chosen as training data. The training data points are used to train each base classifier individually (mixing VFDT and Naive Bayes models), and simplified training accuracy is used to update each base classifier.

Droplets Ensemble Algorithm (DEA) [67] is a novel ensemble learning algorithm that won the Best Paper Award at ICDM in 2016. It dynamically maintains an ensemble of n base learners (BL) and a set of p Droplets related to the current concept. A BL can be any base algorithm as long as it can classify on data streams with concept drift. A Droplet is essentially a multidimensional feature space, with each Droplet associated with an observation value and maintaining a pointer to a BL. First, the algorithm needs to find the most recent feature subspace associated with a BL by summing the prediction errors of each BL on the most recent N Droplets. If a unique BL alone minimizes this sum, it is associated with the most recent Droplets; otherwise (if at least two BLs minimize the prediction error sum), the search space is sequentially expanded to N+1, N+2, N+3, ···until the best BL is found farthest from the Droplets. Then a new Droplet is added to the feature space at coordinate . The vector storing prediction errors is created and a pointer to the best BL found in the previous step is established. The algorithm then proceeds through the overlapping set of Droplets, and if it is not empty, it reduces the influence of Droplets that output incorrect predictions in . This is done by shrinking their radius, making them less likely to predict future observations received in that region of the feature space. If memory is full, the algorithm uses three different criteria to select which Droplet will be deleted: (1) remove the Droplet with the smallest radius; (2) if all Droplets have the same radius, remove the one that has output the largest number of incorrect predictions; (3) if criteria 1 and 2 fail, delete the

oldest Droplet.

To facilitate easier analysis of algorithm performance and advantages/disadvantages, Table 1 details the algorithms' datasets, comparison algorithms, and pros and cons. Overall, AWE is the most classic algorithm for data stream ensemble classifiers, pioneering an era where many subsequent algorithms are based on AWE' s weighting ideas. However, early algorithms did not have outstanding performance, with 不明显 effects on handling concept drift. Currently, algorithms with better performance that can be applied to multiple scenarios in practice mainly include AUE2, DEA, SEA2, WEOB1, and WEOB2.

3.1.2 Fixed Ensemble Architecture

Fixed ensemble architecture defines how base classifiers coordinate and work with each other. Broadly, there are three different architectures: linear and nonlinear combinations (e.g., weighted voting), cascading, and networks. Cascading is a framework where the output of one classifier includes inputs from multiple classifiers (e.g., stacking). Networks are hierarchical frameworks that arrange members into tree-like structures or network ensembles. The given ensemble structure can be classified into: simple linear combinations, meta-learners, and hierarchical tree-like structures.

a) **Linear and Nonlinear Combinations**: Base classifiers are trained on input data, and decision fusion is performed by a combination function for voting. Main algorithms include Online Accuracy Updated Ensemble (OAUE), Online Bagging and Boosting, and Leveraging Bagging. Reference [68] proposes linear and nonlinear weighted combinations, first dividing the data stream into blocks and training base classifiers on each block. The weighting process in the proposed method is performed on base classifiers; when input data is added under different conditions, a linear function and a nonlinear function are used. When the concept is stationary, the nonlinear function is more effective, while when fluctuations in input data are noticeable, the linear function is preferred. On the other hand, the nonlinear function without drift protects classifiers from noise and irrelevant data. Weight allocation uses Mean Absolute Error (MAE).

Linear weighting function: $W_{r_i}^{\text{Linear}} = \max(0, \text{MAE} - \text{MAE}_i - \varepsilon)$

Nonlinear weighting function: $W_{r_i}^{\text{Nonlinear}} = \frac{1}{\text{MAE}_i + \text{MAE} + \varepsilon}$

b) **Meta-Learners**: When training data is very large, meta-learning is considered a more powerful combination strategy. Meta-learning combines through another learner. Individual learners are called primary learners, while the learner used for combination is called a secondary or meta-level learner. A classic representative is Combining Restricted Hoeffding Trees using Stacking [69], which uses column attribute subsets to build a set of Hoeffding trees and then uses ADWIN monitoring mechanisms from data

streams to set the learning rate of sigmoid perceptrons. When perceptron classification performance is poor, ADWIN is used to reset ensemble members.

c) **Hierarchical Structure**: In this structure, ensemble members are represented as vertices of a network, with connections determined according to specific criteria. Connections between classifiers are generated according to a scale-free network model, making classifiers with higher estimated accuracy more likely to connect to recently added classifiers. During voting, classifier weights are proportional to a given centrality measure (e.g., feature vector, betweenness). Since highly accurate base classifiers usually need to receive most connections, these connections are expected to have higher influence on overall decisions. In Social Adaptive Ensemble (SEA) [70] and Advances on the Social Adaptive Ensemble (SAE2) [71], each pair of learners is connected and weighted according to a similarity function. The weighted network formed by all these connections is updated each cycle to better approximate the learners' current state. This network arrangement is used during prediction, where individual decisions are first grouped within similar classifier subsets, and these subset decisions are then combined to obtain the final prediction.

3.2 Dynamic Model Updating

Another important component of ensemble classification algorithms is how to dynamically update the ensemble model. The goal is to retain base classifiers that can adapt to the current data distribution while deleting poorly performing or older classifiers. Therefore, when performing data stream classification tasks, algorithms that learn from data streams require not only accuracy but also rapid environmental adaptation and recovery capabilities. Adaptability and recovery from concept drift are important evaluation criteria for classifier performance, making dynamic updating of ensemble classifiers paramount.

3.2.1 Incremental Model Section 2.2 introduced two approaches to incremental learning in data streams. This section focuses on typical incremental ensemble algorithms and compares incremental models with batch processing (block-based) algorithms. Batch learners must store a batch of instances before training using data stream instances, dividing the data stream into different blocks and training on each block. Whenever the latest data block arrives, the ensemble model is updated using this block, typically by comparing the performance of candidate classifiers with all base classifiers in the ensemble, eliminating or deleting the worst-performing base classifiers based on some performance evaluation. Incremental learners train instances individually as they arrive, and are generally more effective when applied to streams with gradual or progressive drift or when combined with drift detectors. In cases of sudden drift, incremental learners (without drift detector assistance) may require more time to recover because the model is influenced by previously presented concepts, whereas batch

learners completely discard their previous models. Due to the characteristics of data streams, incremental learning algorithms are essential components of data stream classification.

The Learn++ algorithm is a very typical incremental ensemble algorithm that uses weighted voting for final predictions. Based on Learn++, several algorithms have emerged to address different practical problem requirements: Learn++.MT, Learn++.MT2, Learn++.NSE, Learn++.SMOTE, etc. Examples of incremental learners also include Bayesian classifiers, decision trees, and regression trees [72~75].

3.2.2 Sliding Window Sliding windows are similar to landmark windows in that they both define a window size n, though sliding windows discard only one instance at a time. Instance-based classifiers [76~78] operate on this principle. In Multi-Window Based Ensemble Learning (MWEL) [79], three types of windows are defined to store the most recent instances in the data stream: a window for the most recent instances and an ensemble classifier (containing two windows). The ensemble classifier consists of the most recent base classifier and each base classifier used for training (i.e., the ensemble classifier is first composed of a sub-classifier trained on the most recent instances from the data stream and sub-classifiers trained on instance subsets from the data stream). These are the three defined window types. Before predicting the label of a newly arrived instance, all sub-classifiers undergo weighting operations, and sub-classifiers continue training only when their accuracy falls below a defined threshold. If accuracy is below or equal to the defined threshold, it indicates that the current sub-classifier' s classification performance runs counter to the data distribution in the current data stream.

3.2.3 Adaptive Window Adaptive window models can be viewed as landmark windows with different n values. Assuming the stream contains drifts with varying degrees and rates, using windows of different sizes is an appropriate strategy. The challenge is how to dynamically adjust n based on observed stream characteristics. The FLORA2 [80] algorithm uses heuristics (window adjustment algorithms) to increase or decrease window size based on another heuristic guessing whether drift has occurred. This window size adjustment method may be useful in practice but depends on fixed thresholds to define "should decrease" or "increase size." Most importantly, it relies on heuristics to determine whether the current concept is stable or experiencing drift.

ADWIN Bagging and Leveraging Bagging both use ADWIN (Adaptive Window) drift detectors to selectively reset classifiers. Specifically, in these algorithms, classifiers are reset whenever their associated ADWIN detector signals that drift has occurred. Therefore, the ensemble may end up with classifiers at different levels of relevance to the current concept. The main idea of the ADWIN algorithm is: if the two sub-windows w_1 and w_2 of the most recent window W show significantly different averages and the corresponding predicted values are

inferred to be different, the old window is deleted. According to the Hoeffding bound, the difference between the averages of the two windows is greater than threshold ε_{cut} as shown in the formula, where |w| is the size of the most recent window, $|w_1|$ and $|w_2|$ are the sizes of the two sub-windows, and $|w| = |w_1|+|w_2|$. m_1 and m_2 are the harmonic means of the two sub-windows.

$$\varepsilon_{cut} = \sqrt{\frac{1}{2\mu} \ln \frac{4}{\delta}}$$

where $\mu = \frac{|w_1|\cdot|w_2|}{|w_1|+|w_2|}$.

3.2.4 Landmark Window Landmark windows use a marking approach to separate the data stream into disjoint data blocks. Whenever a new instance reaches the landmark, all instances from the previous data block are completely discarded. Ensemble classifiers typically use fixed-size landmark windows of size n to control the periodicity of ensemble model updates, such as classifier deletion, resetting, addition, or statistical resetting. This approach was first introduced in the streaming ensemble algorithm SEA and later used in other algorithms such as Dynamic Weighted Majority (DWM), Accuracy Updated Ensemble (AUE), Accuracy Updated Ensemble2 (AUE2), Social Adaptive Ensemble (SEA), Advances on the Social Adaptive Ensemble (SAE2), and Online Accuracy Updated Ensemble (OAUE). Many ensemble classification algorithms for data streams combine landmark windows with incremental base learners (such as Hoeffding trees). This design choice allows reasonably fast adaptation to sudden drift (given small n values) while enabling incremental updates of ensemble members. Fixed landmark window approaches allow the use of traditional batch learning algorithms for stream learning. In this case, batch learners are trained on instances in window w, and their models are used to classify instances in the next window w+1. After window w+1 ends, the model learned on w is replaced by the model trained on w+1. If this approach is used to adapt batch learners to stream learning, several issues may arise, most notably: the training focus is on the transition period between windows, so if new instances arrive rapidly, delays in prediction must be considered when training new models; batch learners typically require large amounts of data to train accurate models, so windows must be very large, otherwise the learned model will be weak; finally, if concept drift occurs, it will not be considered until the window ends and a new model is generated, resulting in slow adaptation to sudden drift. Despite the simplicity of using fixed landmark windows, it is difficult to define the landmark size parameter n.

4 Future Research Directions

Although researchers have proposed many data stream ensemble classification algorithms that can solve most classification problems, there remain many currently unsolvable issues, such as novel class detection and multi-label detection.

Moreover, in cases with sudden and recurring concept drift, how to improve classifier performance and enable classifiers to have rapid adaptation and recovery capabilities are the main directions for future research.

a) **Novel Class Detection**: For data stream ensemble classification algorithms that can detect novel classes, current issues include inability to handle mixed attributes and low accuracy in novel class detection. The proposed solution uses AUE2 as the benchmark algorithm and improves novel class detection methods to handle mixed-attribute data and improve detection accuracy. Based on the assumption that instances of the same class label are farther apart than instances of other class labels, the proportion of space occupied by attributes is used to determine the existence range of novel class labels, as novel class labels will necessarily fall into different regions while satisfying high intra-cluster cohesion characteristics.

b) **Multi-Label Detection**: Ensemble methods clearly demonstrate better performance than single classification models for multi-label problems, so the proposed approach uses ensemble methods to solve this issue. The consideration is whether to place multiple class labels in a set and weight the class labels in the set using ensemble model weights. When predicting unknown sample labels, the ensemble model provides the most likely class label, ultimately transforming the multi-label problem into a single-label problem. However, this approach requires considering how to update all class labels in the set, with the current plan still using intra-cluster cohesion characteristics to solve this problem, requiring further research and experimentation.

c) **Recurring Concept Drift**: The most difficult problem in recurring concept drift is how to determine whether a newly arrived concept is one that the learner has previously learned. Research will focus on solving this issue, considering whether a multi-ensemble window model can be used. Based on this experimental research idea: in sub-ensemble windows, when concepts from the latest data stream flow into the window, it is determined whether they have novel class labels. If they have novel class labels, they must be concepts not previously learned. When concepts that have not occurred for a long time suddenly reappear, drift detection, recurring drift detection, novel class label detection, and forgetting mechanisms are added to the multi-ensemble window to solve this problem.

5 Conclusion

This paper reviews more than 40 existing data stream ensemble classification algorithms, detailing various algorithms and their applicable environments. It analyzes the advantages and disadvantages of algorithms, experimental datasets, and comparison algorithms. Finally, it introduces future research directions and problems to be solved, proposing research ideas and solutions.

References

[1] Aloraini A. Penalized ensemble feature selection methods for hidden associations in time series environments case study: equities companies in Saudi Stock Exchange Market [J]. Evolving Systems, 2014, 6 (2): 1-8.

[2] Alzoubi O, Fossati D, D' Mello S, et al. Affect detection from non-stationary physiological data using ensemble classifiers [J]. Evolving Systems, 2015, 6 (2): 89-101.

[3] Ditzler G, Roveri M, Alippi C, et al. Learning in Nonstationary Environments: A Survey [J]. IEEE Computational Intelligence Magazine, 2015, 10 (4): 12-25.

[4] Amiribesheli M, Benmansour A, Bouchachia A. A review of smart homes in healthcare [J]. Journal of Ambient Intelligence & Humanized Computing, 2015, 6 (4): 495-517.

[5] Bifet A, Read J, Pfahringer B, et al. Pitfalls in benchmarking data stream classification and how to avoid them [C]// Proc of European Conference on Machine Learning and Knowledge Discovery in Databases. New York: Springer-Verlag Inc., 2013: 465-479.

[6] Lu Zhenyu, Wu Xindong, Bongard J C. Active learning through adaptive heterogeneous ensembling [J]. IEEE Trans on Knowledge & Data Engineering, 2014, 27 (2): 368-381.

[7] Haque A, Parker B, Khan L. Labeling Instances in evolving data streams with MapReduce [C]// Proc of IEEE International Congress on Big Data. 2013: 387-394.

[8] Khamassi I, Sayed-Mouchaweh M, Hammami M, et al. Discussion and review on evolving data streams and concept drift adapting [J]. Evolving Systems, 2016, 9 (1): 1-23.

[9] Vivekanandan P, Nedunchezhian R. Mining data streams with concept drifts using genetic algorithm [J]. Artificial Intelligence Review, 2011, 36 (3): 163-178.

[10] Mao Guojun, Hu Dianjun, Xie Songyan. Big data classification model and algorithm based on distributed data stream [J]. Chinese Journal of Computers, 2017 (1): 161-175.

[11] Chapelle O, Chapelle O, Langford J. A reliable effective terascale linear learning system [J]. Journal of Machine Learning Research, 2014, 15 (1): 1111-1133.

[12] Yousefi M, Yousefi M, Ferreira R, et al. Chaotic genetic algorithm and Adaboost ensemble metamodeling approach for optimum resource planning in emergency departments [J]. Artificial Intelligence in Medicine, 2017, 84: 23-33.

[13] Barddal J P, Gomes H M, Enembreck F. A survey on feature drift adaptation [C]// Proc of IEEE International Conference on Tools with Artificial Intelligence. Washington DC: IEEE Computer Society, 2015: 1053-1060.

[14] Biggio B, Corona I, Nelson B, et al. Security evaluation of support vector machines in adversarial environments [M]// Support Vector Machines Applications. Springer, 2014: 105-153.

[15] Lu Yanyun, Boukharouba K, Boonært J, et al. Application of an incremental SVM algorithm for on-line human recognition from video surveillance using texture and color features [J]. Neurocomputing, 2014, 126 (3): 132-140.

[16] Saffari A, Leistner C, Santner J, et al. On-line Random Forests [C]// Proc of IEEE International Conference on Computer Vision Workshops. 2009: 79-92.

[17] Losing V, Hammer B, Wersing H. Interactive online learning for obstacle classification on a mobile robot [C]// Proc of International Joint Conference on Neural Networks. 2015: 1-8.

[18] Barddal J P, Gomes H M. SFNClassifier: a scale-free social network method to handle concept drift [S]. 2014: 786-791.

[19] Zhou Zhihua. Machine Learning [M]. Beijing: Tsinghua University Press, 2016.

[20] Abbaszadeh O, Amiri A, Khanteymoori A R. An ensemble method for data stream classification in the presence of concept drift [J]. Frontiers of Information Technology & Electronic Engineering, 2015, 16 (12): 1059-1069.

[21] Ding Jian, Han Meng, Li Juan. Overview of conceptual drift data stream mining algorithms [J]. Computer Science, 2016, 43 (12): 24-29.

[22] Krawczyk B, Cano A. Online ensemble learning with abstaining classifiers for drifting and noisy data streams [J]. Applied Soft Computing, 2018, 68: 677-690.

[23] Wen Yimin, Qiang Baohua, Fan Zhigang. A review of research on conceptual drift data stream classification [J]. Journal of Intelligent Systems, 2013, 8 (2): 95-104.

[24] Czarnowski I, Jędrzejowicz P. Ensemble classifier for mining data streams [J]. Procedia Computer Science, 2014, 35 (9): 397-406.

[25] Bosnić Z, Demšar J, Kešpret G, et al. Enhancing data stream predictions with reliability estimators and explanation [J]. Engineering Applications of Artificial Intelligence, 2014, 34 (3): 178-192.

[26] Wozniak M. A hybrid decision tree training method using data streams [M]. New York: Springer-Verlag Inc., 2011.

[27] Shan Jingsong, Luo Jianxin, Ni Guiqiang, et al. CVS: Fast cardinality estimation for large-scale data streams over sliding windows [J]. Neurocomputing, 2016, 194 (1): 107-116.

[28] Bifet A, Holmes G, Pfahringer B, et al. New ensemble methods for evolving data streams [C]// Proc of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2009: 139-148.

[29] Bifet A, Holmes G, Pfahringer B. Leveraging Bagging for Evolving Data Streams [C]// Proc of European Conference on Machine Learning and Knowledge Discovery in Databases. Springer-Verlag, 2010: 135-150.

[30] Sakthithasan S, Pears R, Koh Y S. One pass concept change detection for data streams [C]// Proc of Pacific-Asia Conference on Knowledge Discovery and Data Mining. Berlin: Springer, 2013: 461-472.

[31] Pears R, Sakthithasan S, Koh Y S. Detecting concept change in dynamic data streams [J]. Machine Learning, 2014, 97 (3): 259-293.

[32] Bai Yang. Research on data stream concept drift detection and unbalanced data stream classification algorithm [D]. Beijing: Beijing Jiaotong University, 2017.

[33] Li Nan, Guo Yude, Chen Lifei. Concept drift detection algorithm based on a small number of class labels [J]. Journal of Computer Applications, 2012, 32 (8): 2176-2181.

[34] Mejri D, Khanchel R, Limam M. An ensemble method for concept drift in nonstationary environment [J]. Journal of Statistical Computation & Simulation, 2013, 83 (6): 1115-1128.

[35] Trawiński B, Smętek M, Lasota T, et al. Evaluation of Fuzzy System Ensemble Approach to Predict from a Data Stream [M]// Intelligent Information and Database Systems. Springer International Publishing, 2014: 445-454.

[36] Khamassi I, Sayed-Mouchaweh M. Drift detection and monitoring in non-stationary environments [C]// Evolving and Adaptive Intelligent Systems. IEEE, 2014: 1-6.

[37] Barddal J P, Gomes H M, Enembreck F. A survey on feature drift adaptation [C]// Proc of IEEE International Conference on TOOLS with Artificial Intelligence. Washington DC: IEEE Computer Society, 2015: 1053-1060.

[38] Han Donghong, et al. Two birds with one stone: classifying positive and unlabeled examples on uncertain data streams [J]. Neurocomputing 277 (2018): 149-160.

[39] Sun Yange, Wang Zhihai, Yuan Jidong, et al. Adaptive integrated classification algorithm for data stream sliding window mode [J]. Journal of Beijing Jiaotong University, 2016, 40 (5): 9-15.

[40] Ikonomovska E, Gama J, Džeroski S. Learning model trees from evolving data streams [J]. Data Mining & Knowledge Discovery, 2011, 23 (1): 128-168.

[41] Bifet A. SAMOA: scalable advanced massive online analysis [M]. JMLR.org, 2015.

[42] Ditzler G, Roveri M, Alippi C, et al. Learning in Nonstationary Environments: A Survey [J]. IEEE Computational Intelligence Magazine, 2015, 10 (4): 12-25.

[43] Jiang Aike, Zhao Feng, Zhang Jie. Data flow concept drift algorithm based on adaptive integrated classifier [J]. Statistics and Decision, 2016 (7): 13-17.

[44] Sidhu P, Bhatia M P S. A novel online ensemble approach to handle concept drifting data streams: diversified dynamic weighted majority [J]. International Journal of Machine Learning & Cybernetics, 2015 (1): 1-25.

[45] Wang Haixun, Han Jiawei, et al. Mining Concept-Drifting Data Streams [M]// Data Mining and Knowledge Discovery Handbook. 2009: 789.

[46] Street W N, Kim Y S. A streaming ensemble algorithm (SEA) for large-scale classification [C]// Proc. of ACM SIGKDD Int. Conference on Knowledge Discovery and Data Mining. 2001: 377-382.

[47] Ramamurthy S, Bhatnagar R. Tracking recurrent concept drift in streaming data using ensemble classifiers [C]// Proc of International Conference on Machine Learning and Applications. Washington DC: IEEE Computer Society, 2007: 404-409.

[48] Pelossof R, Jones M, Vovsha I, et al. Online coordinate boosting [C]// Proc of IEEE International Conference on Computer Vision Workshops. 2008: 1-8.

[49] Brzezinski D, Stefanowski J. Combining block-based and online methods in learning ensembles from concept drifting data streams [J]. Information Sciences, 2014, 265 (5): 50-67.

[50] Ludmila I. Kuncheva. A Bound on Kappa-Error Diagrams for Analysis of Classifier Ensembles [J]. IEEE Trans on Knowledge and Data Engineering, 2013, 25 (3): 494-501.

[51] Ikonomovska E, Gama J, Džeroski S. Online tree-based ensembles and option trees for regression on evolving data streams [J]. Neurocomputing, 2015, 150: 458-470.

[52] Wang Boyu, Joelle Pineau. Online bagging and boosting for imbalanced data streams [J]. IEEE Trans on Knowledge & Data Engineering 1 (2016): 1.

[53] Bühlmann P. Bagging, boosting and ensemble methods [M]// Handbook of Computational Statistics. Berlin: Springer, 2012: 985-1022.

[54] Bifet A. Adaptive learning from evolving data streams [C]// Proc of International Symposium on Intelligent Data Analysis: Advances in Intelligent Data Analysis Viii. Springer-Verlag, 2009: 249-260.

[55] Žliobaitė I, Bifet A, Read J, et al. Evaluation methods and decision theory for classification of streaming data with temporal dependence [J]. Machine Learning, 2015, 98 (3): 455-482.

[56] Ma Xianzhe. Research on data stream classification algorithm based on ensemble classifier [D]. Changchun: Northeastern University, 2012.

[57] Kourtellis N, DeMorales G F, Bifet A, et al. VHT: vertical hoeffding tree [C]// Proc of IEEE International Conference on Big Data. Piscataway, NJ:

IEEE Press, 2016: 915-922.

[58] Wang Shuo, Minku, Leandro L, Yao Xin. A multi-objective ensemble method for online class imbalance learning [C]// Proc of International Joint Conference on Neural Networks. 2014: 3311-3318.

[59] Wang Shuo, Minku, Leandro L, Yao Xin. A learning framework for online class imbalance learning [C]// Proc of IEEE Symposium on Computational Intelligence and Ensemble Learning. 2013: 36-45.

[60] Littlestone N, Warmuth M K. The weighted majority algorithm (supersedes 89-16) [J]. Revista Española De Física, 2011.

[61] Brzezinski D, Stefanowski J. Reacting to different types of concept drift: the Accuracy Updated Ensemble algorithm [J]. IEEE Trans on Neural Networks & Learning Systems, 2014, 25 (1): 81-94.

[62] Md Farid D, Zhang Li, Hassain A, et al. An adaptive ensemble classifier for mining concept drifting data streams [J]. Expert Systems with Applications, 2013, 40 (15): 5895-5906.

[63] Wang Shuo, Minku L L, Yao Xin. Resampling-based ensemble methods for online class imbalance learning [J]. IEEE Trans on Knowledge and Data Engineering, 2015, 27 (5): 1356-1368.

[64] Krawczyk B, Cano A. Online ensemble learning with abstaining classifiers for drifting and noisy data streams [J]. Applied Soft Computing, 2018, 68: 677-690.

[65] Parker B S, Khan L, Bifet A. Incremental ensemble classifier addressing non-stationary fast data streams [C]// Proc of IEEE International Conference on Data Mining. 2014: 716-723.

[66] Loeffel P X, Bifet A, Marsala C, et al. Droplet Ensemble Learning on Drifting Data Streams [C]// Proc of International Symposium on Intelligent Data Analysis. Cham: Springer, 2017: 210-222.

[67] Abbaszadeh O, Amiri A, Khanteymoori A R. An ensemble method for data stream classification in the presence of concept drift [J]. Frontiers of Information Technology & Electronic Engineering, 2015, 16 (12): 1059-1070.

[68] Bifet A, Frank E, Holmes G, et al. Accurate ensembles for data streams: combining restricted Hoeffding trees using stacking [J]. Journal of Machine Learning Research, 2010, 13 (13): 225-240.

[69] Gomes H M, Enembreck F. SAE: social adaptive ensemble classifier for data streams [C]// Computational Intelligence and Data Mining. 2013: 199-206.

[70] Gomes H M, Enembreck F. SAE2: advances on the social adaptive ensemble classifier for data streams [C]// Proc of ACM Symposium on Applied Computing. 2014.

[71] Wozniak M, Ksieniewicz P, Cyganek B, et al. Active learning classification of drifted streaming data [J]. Procedia Computer Science, 2016, 80 (C): 186-193.

[72] Yin Chunyong, Feng Lu, Ma Luyu. An improved Hoeffding-ID data-stream classification algorithm [J]. Journal of Supercomputing, 2016, 72 (7): 2670-2687.

[73] Sarafis I, Diou C, Delopoulos A. Online training of concept detectors for image retrieval using streaming clickthrough data [J]. Engineering Applications of Artificial Intelligence, 2016, 51: 150-162.

[74] Sancho-Asensio A, Orriols-Puig A, Casillas J. Evolving association streams [J]. Information Sciences, 2016, 334–335 (C): 250-272.

[75] Ryang H, Yun U. High utility pattern mining over data streams with sliding window technique [J]. Expert Systems with Applications, 2016, 57: 214-231.

[76] Naik S B, Pawar J D. A quick algorithm for incremental mining closed frequent itemsets over data streams [C]// Proc of ACM IKDD Conference on Data Sciences. New York: ACM Press, 2015: 126-127.

[77] Lifna C S, M. Vijayalakshmi D. Identifying concept-drift in Twitter streams [J]. Procedia Computer Science, 2015, 45: 86-94.

[78] Wang Ye, Li Hu, Wang Hua, et al. Multi-window based ensemble learning for classification of imbalanced streaming data [C]// Proc of International Conference on Web Information Systems Engineering. 2015: 78-92.

[79] Widmer G, Kubat M. Learning flexible concepts from streams of examples: FLORA2 [C]// Proc of European Conference on Artificial Intelligence. Hoboken: Wiley, 1992: 463-467.

[80] Krawczyk B, Cano A. Online ensemble learning with abstaining classifiers for drifting and noisy data streams [J]. Applied Soft Computing, 2018, 68: 677-690.

Note: Figure translations are in progress. See original paper for figures.