Learning under Concept Drift: A Review

Jie Lu, *Fellow, IEEE,* Anjin Liu, *Member, IEEE,* Fan Dong, Feng Gu, João Gama, and Guangquan Zhang

Abstract—Concept drift describes unforeseeable changes in the underlying distribution of streaming data over time. Concept drift research involves the development of methodologies and techniques for drift detection, understanding and adaptation. Data analysis has revealed that machine learning in a concept drift environment will result in poor learning results if the drift is not addressed. To help researchers identify which research topics are significant and how to apply related techniques in data analysis tasks, it is necessary that a high quality, instructive review of current research developments and trends in the concept drift field is conducted. In addition, due to the rapid development of concept drift in recent years, the methodologies of learning under concept drift have become noticeably systematic, unveiling a framework which has not been mentioned in literature. This paper reviews over 130 high quality publications in concept drift related research areas, analyzes up-to-date developments in methodologies and techniques, and establishes a framework of learning under concept drift including three main components: concept drift detection, concept drift understanding, and concept drift adaptation. This paper lists and discusses 10 popular synthetic datasets and 14 publicly available benchmark datasets used for evaluating the performance of learning algorithms aiming at handling concept drift. Also, concept drift related research directions are covered and discussed. By providing state-of-the-art knowledge, this survey will directly support researchers in their understanding of research developments in the field of learning under concept drift.

Index Terms—concept drift, change detection, adaptive learning, data streams

1 Introduction

GOVERNMENTS and companies are generating huge amounts of streaming data and urgently need efficient data analytics and machine learning techniques to support them making predictions and decisions. However, the rapidly changing environment of new products, new markets and new customer behaviors inevitably results in the appearance of concept drift problem. Concept drift means that the statistical properties of the target variable, which the model is trying to predict, change over time in unforeseen ways [1]. If the concept drift occurs, the induced pattern of past data may not be relevant to the new data, leading to poor predictions and decision outcomes. The phenomenon of concept drift has been recognized as the root cause of decreased effectiveness in many data-driven information systems such as data-driven early warning systems and data-driven decision support systems. In an ever-changing and big data environment, how to provide more reliable data-driven predictions and decision facilities has become a crucial issue.

Concept drift problem exists in many real-world situations. An example can be seen in the changes of behavior in mobile phone usage, as shown in Fig. 1. From the bars in this figure, the time percentage distribution of the mobile phone usage pattern has changed from "Audio Call" to "Camera" and then to "Mobile Internet" over the past two decades.

Recent attractive research in the field of concept drift targets more challenging problems, i.e., how to accurately detect concept drift in unstructured and noisy datasets [2], [3], how to quantitatively understand concept drift in a explainable way [4], [5], and how to effectively react to drift by adapting related knowledge [6], [7].

Solving these challenges endows prediction and decision-making with the adaptability in an uncertain environment. Conventional research related to machine learning has been significantly improved by introducing concept drift techniques in data science and artificial intelligence in general, and in pattern recognition and data stream mining in particular. These new studies enhance the effectiveness of analogical and knowledge reasoning in an ever-changing environment. A new topic is formed during this development: adaptive data-driven prediction/decision systems. In particular, concept drift is a highly prominent and significant issue in the context of the big data era because the uncertainty of data types and data distribution is an inherent nature of big data.

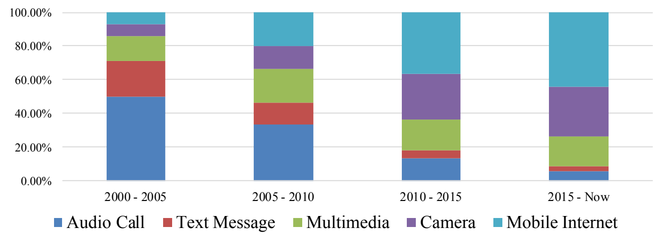


Fig. 1. Concept drift in mobile phone usage (data used in figure are for demonstration only)

Conventional machine learning has two main components: training/learning and prediction. Research on learning under concept drift presents three new components: drift detection (whether or not drift occurs), drift understanding (when, how, where it occurs) and drift adaptation (reaction to the existence of drift) as shown in Fig. 2. These will be discussed in Section 3-5.

In literature, a detailed concept drift survey paper [8] was published in 2014 but intentionally left certain subproblems of concept drift to other publications, such as the details of the data distribution change ($P(X)$) as mentioned in their Section 2.1. In 2015, another comprehensive survey paper [9] was published, which surveys and gives tutorial of both the established and the state-of-the-art approaches. It provides a hybrid-view about concept drift from two

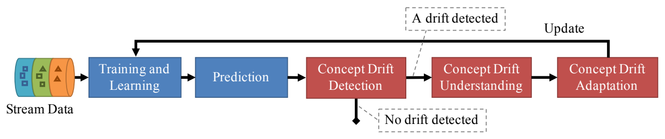


Fig. 2. Framework for handling concept drift in machine learning. Please note that some methods can do concept drift detection and concept drift understanding simultaneously.

primary perspectives, active and passive. Both survey papers are comprehensive and can be a good introduction to concept drift researching. However, many new publications have become available in the last three years, even a new category of drift detection methods has arisen, named multiple hypothesis tests drift detection. It is necessary to review the past research focuses and give the most recent research trends about concept drift, which is one of the main contribution of this survey paper.

Besides these two publications, four related survey papers [6], [7], [10], [11] have also provided valuable insights into how to address concept drift, but their specific research focus is only on data stream learning, rather than analyzing concept drift adaptation algorithms and understanding concept drift. Specifically, paper [7] focuses on data reduction for stream learning incorporating concept drift, while [6] only focuses on investigating the development in learning ensembles for data stream learning in a dynamic environment. [11] concerns the evolution of data stream clustering, and [10] focuses on investigating the current and future trends of data stream learning. There is therefore a gap in the current literature that requires a fuller picture of established and the new emerged research on concept drift; a comprehensive review of the three major aspects of concept drift: concept drift detection, understanding and adaptation, as shown in Fig. 2; and a discussion about the new trend of concept drift research.

The selection of references in this survey paper was performed according to the following steps:

Step 1. Publication database: Science Direct, ACM Digital Library, IEEE Xplore and SpringerLink.

Step 2. Preliminary screening of articles: The first search was based on keywords. The articles were then selected as references if they 1) present new theory, algorithm or methodology in the area of concept drift, or 2) report a concept drift application.

Step 3. Result filtering for articles: The articles selected in Step 2 were divided into three groups: concept drift detection, understanding, and adaptation. The references in each group were filtered again, based on 1) Time: published mainly within the last 10 years, or 2) Impact: published in high quality journals/conferences or having high citations.

Step 4. Dataset selection: To help readers test their research results, this paper lists popular datasets and their characteristics, the dataset providers, and how each dataset can be used.

On completion of this process, 137 research articles, 10 widely used synthetic datasets for evaluating the performance of learning algorithms dealing with concept drift, and 14 publicly available and widely used real-world datasets were listed for discussion.

The main contributions of this paper are:

1) It perceptively summarizes concept drift research achievements and clusters the research into three categories: concept drift detection, understanding and adaptation, providing a clear framework for concept drift research development (Fig. 2);
2) It proposes a new component, concept drift understanding, for retrieving information about the status of concept drift in aspects of when, how, and where. This also creates a connection between drift detection and drift adaptation;
3) It uncovers several very new concept drift techniques, such as active learning under concept drift and fuzzy competence model-based drift detection, and identifies related research involving concept drift;
4) It systematically examines two sets of concept drift datasets, Synthetic datasets and Real-world datasets, through multiple dimensions: dataset description, availability, suitability for type of drift, and existing applications;
5) It suggests several emerging research topics and potential research directions in this area.

The remainder of this paper is structured as follows. In Section 2, the definitions of concept drift are given and discussed. Section 3 presents research methods and algorithms in concept drift detection. Section 4 discusses research developments in concept drift understanding. Research results on drift adaptation (concept drift reaction) are reported in Section 5. Section 6 presents evaluation systems and related datasets used to test concept drift algorithms. Section 7 summaries related research concerning the concept drift problem. Section 8 presents a comprehensive analysis of main findings and future research directions.

2 Problem Description

This section first gives the formal definition and the sources of concept drift in Section 2.1. Then, in Section 2.2, the commonly defined types of concept drift are introduced.

2.1 Concept drift definition and the sources

Concept drift is a phenomenon in which the statistical properties of a target domain change over time in an arbitrary way [3]. It was first proposed by [12] who aimed to point out that noise data may turn to non-noise information at different time. These changes might be caused by changes in hidden variables which cannot be measured directly [4]. Formally, concept drift is defined as follows:

Given a time period $[0, t]$, a set of samples, denoted as $S_{0,t} = \{d_0, \ldots, d_t\}$, where $d_i = (X_i, y_i)$ is one observation (or a data instance), X_i is the feature vector, y_i is the label, and $S_{0,t}$ follows a certain distribution $F_{0,t}(X, y)$. Concept drift occurs at timestamp $t+1$, if $F_{0,t}(X, y) \neq F_{t+1,\infty}(X, y)$, denoted as $\exists t\colon P_t(X, y) \neq P_{t+1}(X, y)$ [2], [8], [13], [14].

Concept drift has also been defined by various authors using alternative names, such as dataset shift [15] or concept shift [1]. Other related terminologies were introduced in [16]'s work, the authors proposed that concept drift or shift is only one subcategory of dataset shift and the dataset shift is consists of covariate shift, prior probability shift and

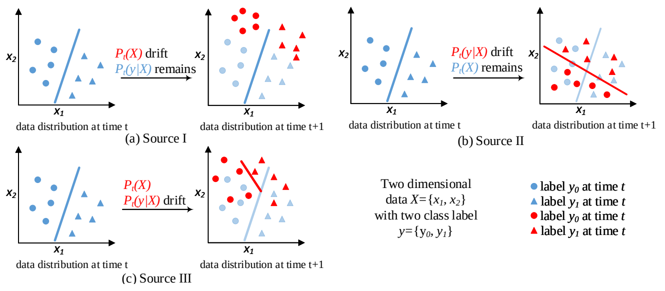

x_2
x_1
$P_t(X)$ drift
$P_t(y|X)$ remains
data distribution at time t
data distribution at time t+1
(a) Source I
$P_t(y|X)$ drift
$P_t(X)$ remains
(b) Source II
$P_t(X)$
$P_t(y|X)$ drift
(c) Source III
Two dimensional data $X=\{x_1, x_2\}$ with two class label $y=\{y_0, y_1\}$
● label y_0 at time t
▲ label y_1 at time t
● label y_0 at time t
▲ label y_1 at time t

Fig. 3. Three sources of concept drift

concept shift. These definitions clearly stated the research scope of each research topics. However, since concept drift is usually associated with covariate shift and prior probability shift, and an increasing number of publications [2], [8], [13], [14] refer to the term "concept drift" as the problem in which $\exists t$: $P_t(X, y) \neq P_{t+1}(X, y)$. Therefore, we apply the same definition of concept drift in this survey. Accordingly, concept drift at time t can be defined as the change of joint probability of X and y at time t. Since the joint probability $P_t(X, y)$ can be decomposed into two parts as $P_t(X, y) = P_t(X) \times P_t(y|X)$, concept drift can be triggered by three sources:

- **Source I**: $P_t(X) \neq P_{t+1}(X)$ while $P_t(y|X) = P_{t+1}(y|X)$, that is, the research focus is the drift in $P_t(X)$ while $P_t(y|X)$ remains unchanged. Since $P_t(X)$ drift does not affect the decision boundary, it has also been considered as virtual drift [7], Fig. 3(a).
- **Source II**: $P_t(y|X) \neq P_{t+1}(y|X)$ while $P_t(X) = P_{t+1}(X)$ while $P_t(X)$ remains unchanged. This drift will cause decision boundary change and lead to learning accuracy decreasing, which is also called actual drift, Fig. 3(b).
- **Source III**: mixture of Source I and Source II, namely $P_t(X) \neq P_{t+1}(X)$ and $P_t(y|X) \neq P_{t+1}(y|X)$. Concept drift focus on the drift of both $P_t(y|X)$ and $P_t(X)$, since both changes convey important information about learning environment Fig. 3(c).

Fig. 3 demonstrates how these sources differ from each other in a two-dimensional feature space. Source I is feature space drift, and Source II is decision boundary drift. In many real-world applications, Source I and Source II occur together, which creates Source III.

2.2 The types of concept drift

Commonly, concept drift can be distinguished as four types [8] as shown in Fig. 4:

Research into concept drift adaptation in Types 1-3 focuses on how to minimize the drop in accuracy and achieve the fastest recovery rate during the concept transformation process. In contrast, the study of Type 4 drift emphasizes the use of historical concepts, that is, how to find the best matched historical concepts with the shortest time. The new concept may suddenly, incrementally, or gradually reoccur.

To better demonstrate the differences between these types, the term "intermediate concept" was introduced by [8] to describe the transformation between concepts. As mentioned by [4], a concept drift may not only take place

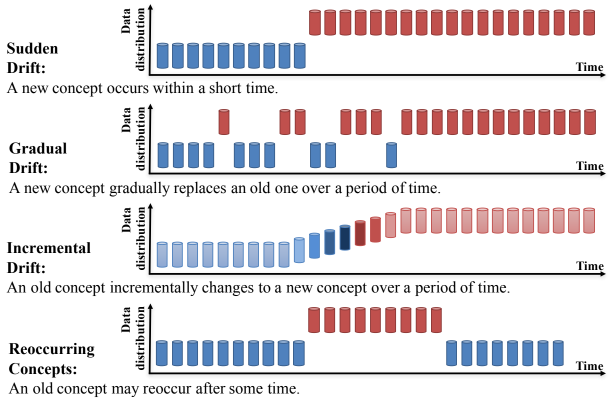

Sudden Drift:
Data distribution
Time
A new concept occurs within a short time.
Gradual Drift:
A new concept gradually replaces an old one over a period of time.
Incremental Drift:
An old concept incrementally changes to a new concept over a period of time.
Reoccurring Concepts:
An old concept may reoccur after some time.

Fig. 4. An example of concept drift types.

at an exact timestamp, but may also last for a long period. As a result, intermediate concepts may appear during the transformation as one concept (starting concept) changes to another (ending concept). An intermediate concept can be a mixture of the starting concept and the ending concept, like the incremental drift, or one of the starting or ending concept, such as the gradual drift.

3 Concept Drift Detection

This section focuses on summarizing concept drift detection algorithms. Section 3.1 introduces a typical drift detection framework. Then, Section 3.2 systematically reviews and categorizes drift detection algorithms according to their implementation details for each component in the framework. At last, Section 3.3 lists the state-of-the-art drift detection algorithms with comparisons of their implementation details.

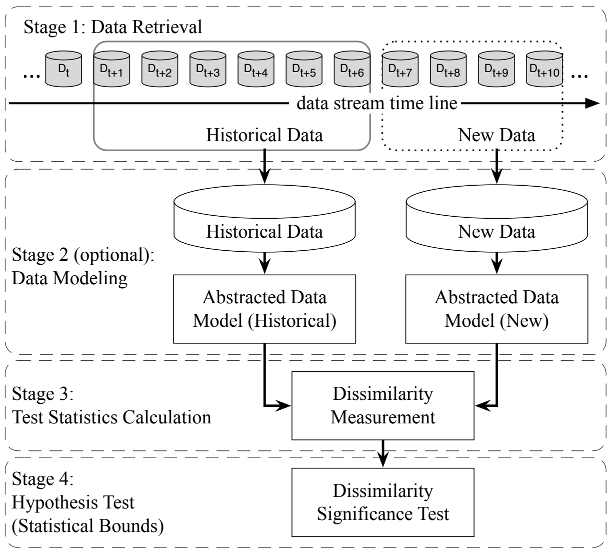
3.1 A general framework for drift detection

Drift detection refers to the techniques and mechanisms that characterize and quantify concept drift via identifying change points or change time intervals [17]. A general framework for drift detection contains four stages, as shown in Fig. 5.

Stage 1 (Data Retrieval) aims to retrieve data chunks from data streams. Since a single data instance cannot carry enough information to infer the overall distribution [2], knowing how to organize data chunks to form a meaningful pattern or knowledge is important in data stream analysis tasks [7].

Stage 2 (Data Modeling) aims to abstract the retrieved data and extract the key features containing sensitive information, that is, the features of the data that most impact a system if they drift. This stage is optional, because it mainly concerns dimensionality reduction, or sample size reduction, to meet storage and online speed requirements [4].

Stage 3 (Test Statistics Calculation) is the measurement of dissimilarity, or distance estimation. It quantifies the severity of the drift and forms test statistics for the hypothesis test. It is considered to be the most challenging aspect of concept drift detection. The problem of how to define an accurate and robust dissimilarity measurement is still an open question. A dissimilarity measurement can also be



Stage 1: Data Retrieval
... D_t D_t+1 D_t+2 D_t+3 D_t+4 D_t+5 D_t+6 D_t+7 D_t+8 D_t+9 D_t+10 ...
data stream time line
Historical Data
New Data
Historical Data
New Data
Stage 2 (optional): Data Modeling
Abstracted Data Model (Historical)
Abstracted Data Model (New)
Stage 3: Test Statistics Calculation
Dissimilarity Measurement
Stage 4: Hypothesis Test (Statistical Bounds)
Dissimilarity Significance Test

Fig. 5. An overall framework for concept drift detection

used in clustering evaluation [11], and to determine the dissimilarity between sample sets [18].

Stage 4 (Hypothesis Test) uses a specific hypothesis test to evaluate the statistical significance of the change observed in Stage 3, or the p-value. They are used to determine drift detection accuracy by proving the statistical bounds of the test statistics proposed in Stage 3. Without Stage 4, the test statistics acquired in Stage 3 are meaningless for drift detection, because they cannot determine the drift confidence interval, that is, how likely it is that the change is caused by concept drift and not noise or random sample selection bias [3]. The most commonly used hypothesis tests are: estimating the distribution of the test statistics [19], [20], bootstrapping [21], [22], the permutation test [3], and Hoeffding's inequality-based bound identification [23].

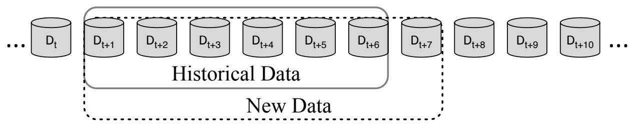
It is also worth to mention that, without Stage 1, the concept drift detection problem can be considered as a two-sample test problem which examines whether the population of two given sample sets are from the same distribution [18]. In other words, any multivariate two-sample test is an option that can be adopted in Stages 2-4 to detect concept drift [18]. However, in some cases, the distribution drift may not be included in the target features, therefore the selection of the target feature will affect the overall performance of a learning system and is a critical problem in concept drift detection [24].

3.2 Concept drift detection algorithms

This section surveys drift detection methods and algorithms, which are classified into three categories in terms of the test statistics they apply.

3.2.1 Error rate-based drift detection

PLearner error rate-based drift detection algorithms form the largest category of algorithms. These algorithms focus on tracking changes in the online error rate of base classifiers. If an increase or decrease of the error rate is proven to be statistically significant, an upgrade process (drift alarm) will be triggered.



... D_t D_t+1 D_t+2 D_t+3 D_t+4 D_t+5 D_t+6 D_t+7 D_t+8 D_t+9 D_t+10 ...
Historical Data
New Data

Fig. 6. Landmark time window for drift detection. The starting point of the window is fixed, while the end point of the window will be extended after a new data instance has been received.

One of the most-referenced concept drift detection algorithms is the Drift Detection Method (DDM) [20]. It was the first algorithm to define the warning level and drift level for concept drift detection. In this algorithm, Stage 1 is implemented by a landmark time window, as shown in Fig. 6. When a new data instance become available for evaluation, DDM detects whether the overall online error rate within the time window has increased significantly. If the confidence level of the observed error rate change reaches the warning level, DDM starts to build a new learner while using the old learner for predictions. If the change reached the drift level, the old learner will be replaced by the new learner for further prediction tasks. To acquire the online error rate, DDM needs a classifier to make the predictions. This process converts training data to a learning model, which is considered as the Stage 2 (Data Modeling). The test statistics in Stage 3 constitute the online error rate. The hypothesis test, Stage 4, is conducted by estimating the distribution of the online error rate and calculating the warning level and drift threshold.

Similar implementations have been adopted and applied in the Learning with Local Drift Detection (LLDD) [25], Early Drift Detection Method (EDDM) [26], Heoffding's inequality based Drift Detection Method (HDDM) [23], Fuzzy Windowing Drift Detection Method (FW-DDM) [5], Dynamic Extreme Learning Machine (DELM) [27]. LLDD modifies Stages 3 and 4, dividing the overall drift detection problem into a set of decision tree node-based drift detection problems; EDDM improves Stage 3 of DDM using the distance between two correct classifications to improve the sensitivity of drift detection; HDDM modifies Stage 4 using Hoeffding's inequality to identify the critical region of a drift; FW-DDM improves Stage 1 of DDM using a fuzzy time window instead of a conventional time window to address the gradual drift problem; DEML does not change the DDM detection algorithm but uses a novel base learner, which is a single hidden layer feedback neural network called Extreme Learning Machine (ELM) [28] to improve the adaptation process after a drift has been confirmed. EWMA for Concept Drift Detection (ECDD) [29] takes advantage of the error rate to detect concept drift. ECDD employs the EWMA chart to track changes in the error rate. The implementation of Stages 1-3 of ECDD is the same as for DDM, while Stage 4 is different. ECDD modifies the conventional EWMA chart using a dynamic mean $\hat{p}_{0,t}$ instead of the conventional static mean p_0, where $\hat{p}_{0,t}$ is the estimated online error rate within time $[0,t]$, and p_0 implies the theoretical error rate when the learner was initially built. Accordingly, the dynamic variance can be calculated by $\sigma^2_{Z_t} = \hat{p}_{0,t}(1-\hat{p}_{0,t})\sqrt{\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2t})}$ where λ controls how much weight is given to more recent data as opposed to older data, and $\lambda = 0.2$ is recommended by the authors.

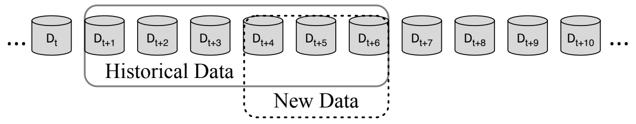


Fig. 7. Two time windows for concept drift detection. The New Data window has to be defined by the user.

Also, when the test statistic of the conventional EWMA chart is $Z_t > \hat{p}_{0,t} + 0.5L\sigma_{Z_t}$, ECDD will report a concept drift warning; when $Z_t > \hat{p}_{0,t} + L\sigma_{Z_t}$, ECDD will report a concept drift. The control limits L is given by the authors through experimental evaluation.

In contrast to DDM and other similar algorithms, Statistical Test of Equal Proportions Detection (STEPD) [30] detects error rate change by comparing the most recent time window with the overall time window, and for each timestamp, there are two time windows in the system, as shown in Fig. 7. The size of the new window must be defined by the user. According to [30], the test statistic θ_{STEPD} conforms to standard normal distribution, denoted as $\theta_{\text{STEPD}} \sim \mathcal{N}(0, 1)$. The significance level of the warning level and the drift level were suggested as $\alpha_w = 0.05$ and $\alpha_d = 0.003$ respectively. As a result, the warning threshold and drift threshold can be easily calculated.

Another popular two-time window-based drift detection algorithm is **AD**aptive **WIN**dowing (ADWIN) [31]. Unlike STEPD, ADWIN does not require users to define the size of the compared windows in advance; it only needs to specify the total size n of a "sufficiently large" window W. It then examines all possible cuts of W and computes optimal sub-window sizes n_{hist} and n_{new} according to the rate of change between the two sub-windows w_{hist} and w_{new}. The test statistic is the difference of the two sample means $\theta_{\text{ADWIN}} = |\hat{\mu}_{\text{hist}} - \hat{\mu}_{\text{new}}|$. An optimal cut is found when the difference exceeds a threshold with a predefined confidence interval δ. The author proved that both the false positive rate and false negative rate are bounded by δ. It is worth noting that many concept drift adaptation methods/algorithms in the literature are derived from or combined with ADWIN, such as [32]–[35]. Since their drift detection methods are implemented with almost the same strategy, we will not discuss them in detail.

3.2.2 Data Distribution-based Drift Detection

The second largest category of drift detection algorithms is data distribution-based drift detection. Algorithms of this category use a distance function/metric to quantify the dissimilarity between the distribution of historical data and the new data. If the dissimilarity is proven to be statistically significantly different, the system will trigger a learning model upgradation process. These algorithms address concept drift from the root sources, which is the distribution drift. Not only can they accurately identify the time of drift, they can also provide location information about the drift. However, these algorithms are usually reported as incurring higher computational cost than the algorithms mentioned in Section 3.2.1 [2]. In addition, these algorithms usually require users to predefine the historical time window and new data window. The commonly used strategy is two sliding windows with the historical time window fixed

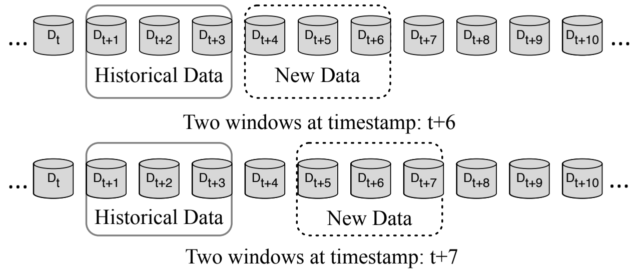


Fig. 8. Two sliding time windows, of fixed size. The Historical Data window will be fixed while the New Data window will keep moving.

while sliding the new data window [3], [22], [36], as shown in Fig. 8.

According to the literature, the first formal treatment of change detection in data streams was proposed by [37]. In their study, the authors point out that the most natural notion of distance between distributions is total variation, as defined by: $TV(P_1, P_2) = 2sup_{E\in\varepsilon}|P_1(E) - P_2(E)|$ or equivalently, when the distribution has the density functions f_1 and f_2, $dist_{L^1} = \int |f_1(x) - f_2(x)|\mathrm{d}x$. This provides practical guidance on the design of a distance function for distribution discrepancy analysis. Accordingly, [37] proposed a family of distances, called Relativized Discrepancy (RD). The authors also present the significance level of the distance according to the number of data instances. The bounds on the probabilities of missed detections and false alarms are theoretically proven, using Chernoff bounds and the Vapnik-Chervonenkis dimension. The authors of [37] do not propose novel high-dimensional friendly data models for Stage 2 (data modeling); instead, they stress that a suitable model choice is an open question.

Another typical density-based drift detection algorithm is the Information-Theoretic Approach (ITA) [22]. The intuitive idea underlying this algorithm is to use kdqTree to partition the historical and new data (multi-dimensional) into a set of bins, denoted as $\mathcal{A}$,and then use Kullback-Leibler divergence to quantify the difference of the density θ_{ITA} in each bin. The hypothesis test applied by ITA is bootstrapping by merging W_{hist}, W_{new} as W_{all} and resampling as W'_{hist}, W'_{new} to recompute the θ^*_{ITA}. Once the estimated probability $P(\theta^*_{\text{ITA}} \geq \theta_{\text{ITA}}) < 1 - \alpha$, concept drift is confirmed, where α is the significant level controlling the sensitivity of drift detection.

Similar distribution-based drift detection methods/algorithms are: Statistical Change Detection for multi-dimensional data (SCD) [38], Competence Model-based drift detection (CM) [2], a prototype-based classification model for evolving data streams called SyncStream [36], PCA-based change detection framework (PCA-CD) [39], Equal Density Estimation (EDE) [40], Least Squares Density Difference-based Change Detection Test (LSDD-CDT) [21], Incremental version of LSDD-CDT (LSDD-INC) [41] and Local Drift Degree-based Density Synchronized Drift Adaptation (LDD-DSDA) [4].

3.2.3 Multiple Hypothesis Test Drift Detection

Multiple hypothesis test drift detection algorithms apply similar techniques to those mentioned in the previous two categories. The novelty of these algorithms is that they use multiple hypothesis tests to detect concept drift in different

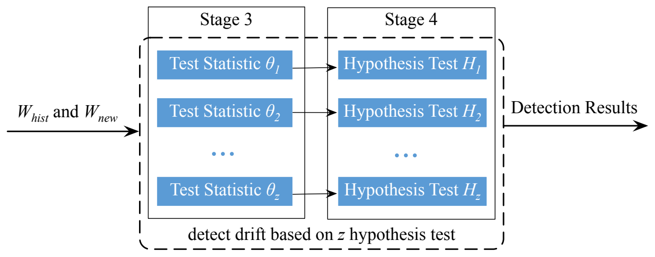


Fig. 9. Parallel multiple hypothesis test drift detection.

ways. These algorithms can be divided into two groups: 1) parallel multiple hypothesis tests; and 2) hierarchical multiple hypothesis tests.

The idea of parallel multiple hypothesis drift detection algorithm is demonstrated in Fig. 9. According to the literature, Just-In-Time adaptive classifiers (JIT) [19] is the first algorithm that set multiple drift detection hypothesis in this way. The core idea of JIT is to extend the CUSUM chart, known as the Computational Intelligence-based **CUSUM** test (CI-CUSUM), to detect the mean change in the features interested by learning systems. The authors of [19], gave the following four configurations for the drift detection target. Config1: the features extracted by Principal Component Analysis (PCA), which removes eigenvalues whose sum is below a threshold, e.g. 0.001. Config2: PCA extracted features plus one generic component of the original features x_i; Config3: detects the drift in each x_i individually. Config4: detects drift in all possible combinations of the feature space x_i. The authors stated that Config2 is the preferred setting for most situations, according to their experimentation, and also mentioned that Config1 may have a high missing rate, Config3 suffers from a high false alarm rate, and Config4 has exponential computational complexity. The same drift detection strategy has also been applied in [42]–[45] for concept drift adaptation.

Similar implementations have been applied in Linear Four Rate drift detection (LFR) [46], which maintains and tracks the changes in True Positive rate (TP), True Negative rate (TN), False Positive rate (FP) and False Negative rate (FN) in an online manner. The drift detection process also includes warning and drift levels.

Another parallel multiple hypothesis drift detection algorithm is three-layer drift detection, based on Information Value and Jaccard similarity (IV-Jac) [47]. IV-Jac aims to individually address the label drift $P_t(y)$ Layer I, feature space drift $P_t(X)$ Layer II, and decision boundary drift $P_t(y|X)$ Layer III. It extracts the Weight of Evidence (WoE) and Information Value (IV) from the available data and then detects whether a significant change exists between the WoE and IV extracted from W_{hist} and W_{new} by measuring the contribution to the label for a feature value. The hypothesis test thresholds are predefined parameters $\theta_{P_t(y)} = \theta_{P_t(X)} = \theta_{P_t(X|y)} = 0.5$ by default, which are chosen empirically.

Ensemble of Detectors (e-Detector) [48] proposed to detect concept drift via ensemble of heterogeneous drift detector. The authors consider two drift detectors are homogeneous as if they are equivalent in finding concept drifts, otherwise they are heterogeneous. e-Detector groups homogeneous drift detectors via a diversity measurement, named diversity vector. For each group, it select the one with the smallest coefficient of failure as the base detector to form the ensemble. e-Detector reports concept drift following the early-find-early-report rule, which means no matter which base detector detect a drift, the e-Detector reports a drift. Similar strategy has been applied in drift detection ensemble (DDE) [49].

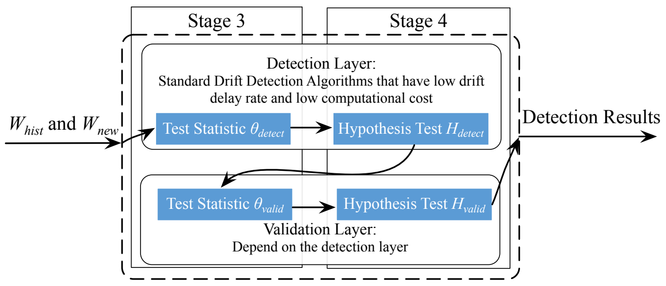

Stage 3
Stage 4
Detection Layer:
Standard Drift Detection Algorithms that have low drift delay rate and low computational cost
W_{hist} and W_{new}
Test Statistic θ_{detect}
Hypothesis Test H_{detect}
Detection Results
Test Statistic θ_{valid}
Hypothesis Test H_{valid}
Validation Layer:
Depend on the detection layer

Fig. 10. Hierarchical multiple hypothesis test drift detection.

Hierarchical drift detection is an emerging drift detection category that has a multiple verification schema. The algorithms in this category usually detect drift using an existing method, called the detection layer, and then apply an extra hypothesis test, called the validation layer, to obtain a second validation of the detected drift in a hierarchical way. The overall workflow is shown in Fig. 10.

According to the claim made by [50], Hierarchical Change-Detection Tests (HCDTs) is the first attempt to address concept drift using a hierarchical architecture. The detection layer can be any existing drift detection method that has a low drift delay rate and low computational burden. The validation layer will be activated and deactivated based on the results returned by the detection layer. The authors recommend two strategies for designing the validation layer: 1) estimating the distribution of the test statistics by maximizing the likelihood; 2) adapting an existing hypothesis test, such as the Kolmogorov-Smirnov test or the Cramer-Von Mises test.

Hierarchical Linear Four Rate (HLFR) [51] is another recently proposed hierarchical drift detection algorithm. It applies the drift detection algorithm LFR as the detection layer. Once a drift is confirmed by the detection layer, the validation layer will be triggered. The validation layer of HLFR is simply a zero-one loss, denoted as E, over the ordered train-test split. If the estimated zero-one loss exceeds a predefined threshold, $\eta = 0.01$, the validation layer will confirm the drift and report to the learning system to trigger a model upgradation process.

Two-Stage Multivariate Shift-Detection based on **EWMA** (TSMSD-EWMA) [52] has a very similar implementation, however, the authors do not claim that their method is a hierarchy-based algorithm.

Hierarchical Hypothesis Testing with Classification Uncertainty (HHT-CU) and Hierarchical Hypothesis Testing with Attribute-wise "Goodness-of-fit" (HHT-AG) are two drift detection algorithms based on request and reverify strategy [53]. For HHT-CU, the detection layer is a hypotheses test based on Heoffding's inequality that monitoring the change of the classification uncertainty measurement. The validation layer is a permutation test that evaluates the change of the zero-one loss of the learner. For HHT-AG, the detection layer is conducted based on Kolmogorov-Smirnov (KS) test for each feature distribution. Then HHT-AG validate the potential drift points by requiring true labels of data that come from w_{new}, and performing d independent two-dimensional (2D) KS test with each feature-label bivariate distribution. Compare to other drift detection algorithms, HHT-AG can handle concept drift with less true labels, which makes it more powerful when dealing with high verification latency.

3.3 Summary of concept drift detection methods/algorithms

TABLE 1 lists the most popular concept drift detection methods/algorithms against the general framework summarized in Section 3.1 (Fig. 5). A comparative study on eight popular drift detection methods can be found in [54].

4 Concept Drift understanding

Drift understanding refers to retrieving concept drift information about "When" (the time at which the concept drift occurs and how long the drift lasts), "How" (the severity /degree of concept drift), and "Where" (the drift regions of concept drift). This status information is the output of the drift detection algorithms, and is used as input for drift adaptation.

4.1 The time of concept drift occurs (When)

The most basic function of drift detection is to identify the timestamp when a drift occurs. Recalling the definition of concept drift $\exists t$: $P_t(X, y) \neq P_{t+1}(X, y)$, the variable t represents the time at which a concept drift occurs. In drift detection methods/algorithms, an alarm signal is used to indicate whether the concept drift has or has not occurred or not at the current timestamp. It is also a signal for a learning system to adapt to a new concept. Accurately identifying the time a drift occurs is critical to the adaptation process of a learning system; a delay or a false alarm will lead to failure of the learning system to track new concepts.

A drift alarm usually has a statistical guarantee with a predefined false alarm rate. Error rate-based drift detection algorithms monitor the performance of the learning system, based on statistical process control. For example, DDM [20] sends a drift signal when the learning accuracy of the learner drops below a predefined threshold, which is chosen by the three-sigma rule [55]. ECCD [29] reports a drift when the online error rate exceeds the control limit of EWMA. Most data distribution-based drift detection algorithms report a drift alarm when two data samples have a statistically significant difference. PCA-based drift detection [36] outputs a drift signal when the p-value of the generalized Wilcoxon test statistic W^l_{BF} is significantly large. The method in [3] confirms that a drift has occurred by verifying whether the empirical competence-based distance is significantly large through permuataion test.

Taking into account the various drift types, concept drift understanding needs to explore the start time point, the change period, and the end time point of concept drift. And these time information could be useful input for the adaptation process of the learning system. However the

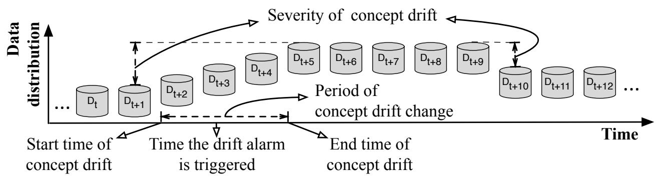


Fig. 11. An example of the occurrence time and the severity of concept drift. One incremental drift starts to change at $t+1$ and ends at $t+5$. The other sudden drift occurs between $t+9$ and $t+10$. The severity of these two concept drifts (D_{t+1}-D_{t+5} and D_{t+9}-D_{t+10}) is marked with brackets.

drift timestamp alert in existing drift detection algorithms is delayed compared to the actual drifting timestamp, since most drift detectors require a minimum number of new data to evaluate the status of the drift, as shown in Fig. 11. The emergence time of the new concept is therefore still vague. Some concept drift detection algorithms such as DDM [20], EDDM [26], STEPD [30], and HDDM [23], trigger a warning level to indicate a drift may have occurred. The threshold used to trigger warning level is a relaxed condition of the threshold used for the drift level; for example, the warning level is set p-value to 95% or 2σ, and the drift level is set p-value to 99% or 3σ. The data accumulated between the warning level and the drift level are used as the training set for updating a learning model.

4.2 The severity of concept drift (How)

The severity of concept drift refers to using a quantified value to measure the similarity between the new concept and the previous concept, as shown in Fig. 11. Formally, the severity of concept drift can be represented as $\Delta = \delta(P_t(X, y), P_{t+1}(X, y))$, where δ is a function to measure the discrepancy of two data distributions, and t is the timestamp when the concept drift occurred. Δ usually is a non-negative value indicating the severity of concept drift. The greater the value of Δ, the larger the severity of the concept drift is.

In general, error rate-based drift detection cannot directly measure the severity of concept drift, because it mainly focuses on monitoring the performance of the learning system, not the changes in the concept itself. However, the degree of decrease in learning accuracy can be used as an indirect measurement to indicate the severity of concept drift. If learning accuracy has dropped significantly when drift is observed, this indicates that the new concept is different from the previous one. For example, the severity of concept drift could be reflected by the difference between p_i and p_{min} in [20], [27], denoted as $\Delta \sim p_i - p_{\text{min}}$; the difference between overall accuracy $\hat{p}_{\text{hist}}$ and recent accuracy $\hat{p}_{\text{new}}$ in [30], expressed as $\Delta \sim \hat{p}_{\text{new}} - \hat{p}_{\text{hist}}$; and the difference between test statistics in the former window $E[\hat{X}_{\text{cut}}]$ and test statistics in the later window $E[\hat{Y}_{i-\text{cut}}]$ [23], denoted as $\Delta \sim E[\hat{Y}_{i-\text{cut}}] - E[\hat{X}_{\text{cut}}]$. However, the meaning of these differences is not discussed in existing publications. The ability of error rate-based drift detection to output the severity of concept drift is still vague.

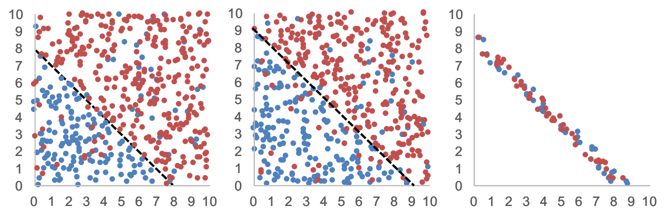
Data distribution-based drift detection methods can directly quantify the severity of concept drift since the measurement used to compare two data samples already reflects the difference. For example, [37] employed *a relaxation of the total variation distance* $d_{\mathcal{A}}(S_1, S_2)$ to measure the difference

TABLE 1
Summary of drift detection algorithms

Category	Algorithms	Stage 1	Stage 2	Stage 3	Stage 4
Error rate-based	DDM [20]	Landmark	Learner	Online error rate	Distribution estimation
	EDDM [26]	Landmark	Learner	Online error rate	Distribution estimation
	FW-DDM [5]	Landmark	Learner	Online error rate	Distribution estimation
	DEML [27]	Landmark	Learner	Online error rate	Distribution estimation
	STEPD [30]	Predefined w_{hist}, w_{new}	Learner	Error rate difference	Distribution estimation
	ADWIN [31]	Auto cut w_{hist}, w_{new}	Learner	Error rate difference	Hoeffding's Bound
	ECDD [29]	Landmark	Learner	Online error rate	EWMA Chart
	HDDM [23]	Landmark	Learner	Online error rate	Hoeffding's Bound
	LLDD [25]	Landmark, or sliding w_{hist}, w_{new}	Decision trees	Tree node error rate	Hoeffding's Bound
Data distribution-based	kdqTree [22]	Fixed w_{hist}, Sliding w_{new}	kdqTree	KL divergence	Bootstrapping
	CM [2], [3]	Fixed w_{hist}, Sliding w_{new}	Competence model	Competence distance	Permutation test
	RD [37]	Fixed w_{hist}, Sliding w_{new}	KS structure	Relativized Discrepancy	VC-Dimension
	SCD [38]	Fixed w_{hist}, Sliding w_{new}	kernel density estimator	log-likelihood	Distribution estimation
	EDE [40]	Fixed w_{hist}, Sliding w_{new}	Nearest neighbor	Density scale	Permutation test
	SyncStream [36]	Fixed w_{hist}, Sliding w_{new}	PCA	P-Tree	Wilcoxon test
	PCA-CD [39]	Fixed w_{hist}, Sliding w_{new}	PCA	Change-Score	Page-Hinkley test
	LSDD-CDT [21]	Fixed w_{hist}, Sliding w_{new}	Learner	Relative difference	Distribution estimation
	LSDD-INC [41]	Fixed w_{hist}, Sliding w_{new}	Learner	Relative difference	Distribution estimation
	LDD-DSDA [4]	Fixed w_{hist}, Sliding w_{new}	k-nearest neighbor	Local drift degree	Distribution estimation
Multiple Hypothesis Tests	JIT [19]	Landmark	Selected features	4 configurations	Distribution estimation
	LFR [46]	Landmark	Learner	TP, TN, FP, FN	Distribution estimation
	Three-layer [47]	Sliding both w_{hist}, w_{new}	Learner	$P(y)$, $P(X)$, $P(X\|y)$	Distribution estimation
	e-Detector [48]	depends on base detector	depends	depends	depends
	DDE [49]	depends on base detector	depends	depends	depends
	TSMSD-EWMA [52]	Landmark	Learner	Online error rate	EWMA Chart
	HCDTs [50]	Landmark	Depending on layers	Depending on layers	Depending on layer
	HLFR [51]	Landmark	Learner	TP, TN, FP, FN	Distribution estimation
	HHT-CU [53]	Landmark	Learner	Classification uncertainty	Layer-I Hoeffding's Bound, Layer-II Permutation Test
	HHT-AG [53]	Fixed w_{hist}, Sliding w_{new}	N/A	KS statistic on each attribute	Layer-I KS test, Layer -II 2D KS test

between two data distributions. [3] proposed *a competence-based empirical distance* $d^{CB}(S_1, S2)$ to show the difference between two data samples. Other drift detection methods have used information-theoretic distance; for example, *Kullback-Leibler divergence* $D(W_1||W_2)$, also called relative entropy, was used in the study reported in [22]. The range of these distances is $[0, 1]$. The greater the distance, the larger the severity of the concept drift is. The distance "1" means that a new concept is different from the pervious one, while the distance "0" means that two data concepts are identical. The *test statistic* δ used in [38] gives an extremely small negative value if the new concept is quite different from the previous concept. The degree of concept drift in [36] is measured by the resulting p-value of the test statistic W^l_{BF} and the defined $Angle(D_t, D_{t+1})$ of two datasets D_t and D_{t+1}.

The severity of concept drift can be used as a guideline for choosing drift adaptation strategies. For example, if the severity of drift in a classification task is low, the decision boundary may not move much in the new concept. Thus, adjusting the current learner by incremental learning will be adequate. In contrast, if the severity of the concept drift is high, the decision boundary could change significantly, therefore discarding the old learner and retraining a new one could be better than incrementally updating the old learner. We would like to mention that, even though some researches have promoted the ability to describe and quantify the severity of the detected drift, this information is not yet widely utilized in drift adaptation.



(a) Data at time t (b) Data at time $t+1$ (c) Data in drift regions

Fig. 12. An example of the drift regions of concept drift.

4.3 The drift regions of concept drift (Where)

The drift regions of concept drift are the regions of conflict between a new concept and the previous concept. Drift regions are located by finding regions in data feature space X where $P_t(X, y)$ and $P_{t+1}(X, y)$ have significant difference. To illustrate this, we give an example of a classification task in Fig. 12. The data used in this task are uniformly sampled in the range of $[0, 10]^2$. The left sub-figure is the data sample accumulated at time t, where the decision boundary is $x_1 + x_2 = 8$. The middle sub-figure is the data accumulated at time $t + 1$, where the decision boundary is $x_1 + x_2 = 9$. Intuitively, data located in regions $8 \leq x_1 + x_2 < 9$ have different classes in time t and time $t + 1$, since the decision boundary has changed. The right sub-figure shows the data located in the drift regions.

The techniques to identify drift regions are highly dependent on the data model used in the drift detection methods/algorithms. Paper [25] detected drift data in local regions of the instance space by using online error-rate inside the inner-nodes of a decision tree. The whole data feature space is partitioned by decision tree. Each leaf of this decision tree corresponds to a hyper-rectangle in the data feature space. All leaf nodes contain a drift detector. When the leaf nodes are alerted to a drift, the corresponding hyper-rectangles indicate the regions of drift in the data feature space. Similar to [25], [22] utilized the nodes of the *kdq-tree* with Kulldorff's spatial scan statistic to identify the regions in which data had changed the most. Once a drift has been reported, Kulldorff's statistic measures how the two datasets differ only with respect to the region associated with the leaf node of the *kdq-tree*. The leaf nodes with the greater statistical value of show the drift regions. [2] highlighted the most severe regions of concept drift through top-p-competence areas. Utilizing the *RelatedSets* of the competence model, the data feature space is partitioned by a set of overlapping hyperspheres. The *RelatedSet*-based empirical distance defines the distance between two datasets on a particular hypersphere. The drift regions are identified by the corresponding hyperspheres with large empirical distance at top p% level. [4] identified drift regions by monitoring the discrepancy in the regional density of data, which is called the local drift degree. A local region with a corresponding increase or decrease in density will be highlighted as a critical drift region.

Locating concept drift regions benefits drift adaptation. Paper [56] pointed out that even if the concept of the entire dataset drifts, there are regions of the feature space that will remain stable significantly longer than other regions. In an ensemble scenario, the old learning models of stable regions could still be useful for predicting data instances located within stable regions, or data instances located in drift regions could be used to build a more updated current model. The authors of [3] also pointed out that identifying drift regions can help in recognizing obsolete data that conflict with current concepts and distinguish noise data from novel data. In their later research [2], they utilized top-p-competence areas to edit cases in a case-based reasoning system for fast new concept switching. One step in their drift adaptation is to remove conflict instances. To preserve as many instances of a new concept as possible, they only remove obsolete conflict instances which are outside the drift regions. LDD-DSDA [4] utilized drift regions as an instance selection strategy to construct a training set that continually tracked a new concept.

4.4 Summary of drift understanding

We summarize concept drift detection algorithms according to their ability to identify when, how, and where concept drift occurs, as shown in TABLE 2. All drift detection algorithms can identify the occurrence time of concept drift (when), and most data distribution-based drift detection algorithms can also measure the severity of concept drift (how) through the test statistics, but only a few algorithms have the ability to locate drift regions (where).

TABLE 2
Summary of drift understanding for drift detection algorithms

Category	Algorithms	When	How	Where
Error rate-based	DDM [20]	√		
	EDDM [26]	√		
	FW-DDM [5]	√		
	DEML [27]	√		
	STEPD [30]	√		
	ADWIN [31]	√		
	ECDD [29]	√		
	HDDM [23]	√		
	LLDD [25]	√		√
Data distribution-based	kdqTree [22]	√	√	√
	CM [2], [3]	√	√	√
	RD [37]	√	√	
	SCD [38]	√	√	
	EDE [40]	√		
	SyncStream [36]	√	√	
	PCA-CD [39]	√	√	
	LSDD-CDT [21]	√		
	LSDD-INC [41]	√		
	LDD-DSDA [4]	√	√	√
Multiple hypothesis tests	JIT [19]	√		
	LFR [46]	√		
	Three-layer drift detection [47]	√		
	e-Detector [48]	√		
	DDE [49]	√		
	EWMA [52]	√		
	HCDTs [50]	√		
	HLFR [51]	√		
	HHT-CU [53]	√		
	HHT-AG [53]	√		

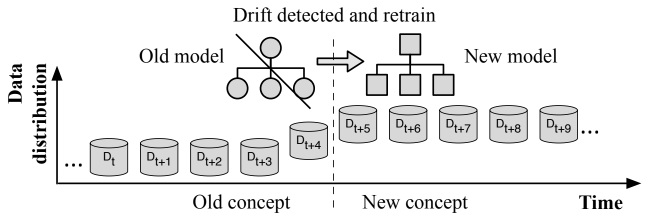


Fig. 13. A new model is trained with latest data to replace the old model when a concept drift is detected.

5 Drift Adaptation

This section focuses on strategies for updating existing learning models according to the drift, which is known as drift adaptation or reaction. There are three main groups of drift adaptation methods, namely simple retraining, ensemble retraining and model adjusting, that aim to handle different types of drift.

5.1 Training new models for global drift

Perhaps the most straightforward way of reacting to concept drift is to retrain a new model with the latest data to replace the obsolete model, as shown in Fig. 13. An explicit concept drift detector is required to decide when to retrain the model (see Section 3 on drift detection). A window strategy is often adopted in this method to preserve the most recent data for retraining and/or old data for distribution change test. *Paired Learners* [57] follows this strategy and uses two learners: the *stable learner* and the *reactive learner*. If the stable learner frequently misclassifies instances that the reactive learner correctly classifies, a new concept is detected and the stable learner will be replaced with the reactive learner. This method is simple to understand and easy to implement, and can be applied at any point in the data stream.

When adopting a window-based strategy, a trade-off must be made in order to decide an appropriate window size. A small window can better reflect the latest data distribution, but a large window provides more data for training a new model. A popular window scheme algorithm that aims to mitigate this problem is ADWIN [31]. Unlike most earlier works, it does not require the user to guess a fixed size of the windows being compared in advance; instead, it examines all possible cuts of the window and computes optimal sub-window sizes according to the rate of change between the two sub-windows. After the optimal window cut has been found, the window containing old data is dropped and a new model can be trained with the latest window data.

Instead of directly retraining the model, researchers have also attempted to integrate the drift detection process with the retraining process for specific machine learning algorithms. DELM [27] extends the traditional ELM algorithm with the ability to handle concept drift by adaptively adjusting the number of hidden layer nodes. When the classification error rate increases — which could indicate the emergence of a concept drift — more nodes are added to the network layers to improve its approximation capability. Similarly, FP-ELM [58] is an ELM-extended method that adapts to drift by introducing a forgetting parameter to the ELM model. A parallel version of ELM-based method [59] has also been developed for high-speed classification tasks under concept drift. OS-ELM [60] is another online learning ensemble of repressor models that integrates ELM using an ordered aggregation (OA) technique, which overcomes the problem of defining the optimal ensemble size.

Instance-based lazy learners for handling concept drift have also been studied intensively. The *Just-in-Time* adaptive classifier [19], [42] is such a method which follows the "detect and update model" strategy. For drift detection, it extends the traditional CUSUM test [61] to a pdf-free form. This detection method is then integrated with a kNN classifier [42]. When a concept drift is detected, old instances (more than the last T samples) are removed from the case base. In later work, the authors of [11], [45] extended this algorithm to handle recurrent concepts by computing and comparing current concept to previously stored concepts. NEFCS [2] is another kNN-based adaptive model. A competence model-based drift detection algorithm [3] was used to locate drift instances in the case base and distinguish them from noise instances and a redundancy removal algorithm, Stepwise Redundancy Removal (SRR), was developed to remove redundant instances in a uniform way, guaranteeing that the reduced case base would still preserve enough information for future drift detection.

5.2 Model ensemble for recurring drift

In the case of recurring concept drift, preserving and reusing old models can save significant effort to retrain a new model for recurring concepts. This is the core idea of using ensemble methods to handle concept drift. Ensemble methods have received much attention in stream data mining research community in recent years. Ensemble methods comprise a set of base classifiers that may have different types or different parameters. The output of each base classifier is combined using certain voting rules to predict the newly arrived data. Many adaptive ensemble methods have been developed that aim to handle concept drift by extending classical ensemble methods or by creating specific adaptive voting rules. An example is shown in Fig. 14, where new base classifier is added to the ensemble when concept drift occurs.

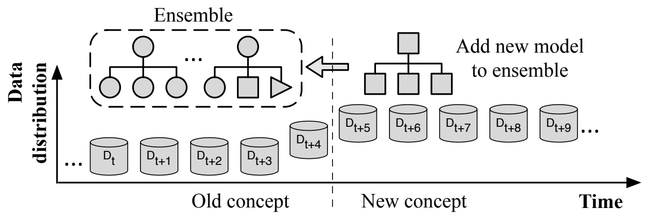


Fig. 14. A new base classifier is added to the ensemble when a concept drift occurs.

Bagging, Boosting and Random Forests are classical ensemble methods used to improve the performance of single classifiers. They have all been extended for handling streaming data with concept drift. An online version of the bagging method was first proposed in [62] which uses each instance only once to simulate the batch mode bagging. In a later study [63], this method was combined with the ADWIN drift detection algorithm [31] to handle concept drift. When a concept drift is reported, the newly proposed method, called Leveraging Bagging, trains a new classifier on the latest data to replace the existing classifier with the worst performance. Similarly, an adaptive boosting method was developed in [64] which handles concept drift by monitoring prediction accuracy using a hypothesis test, assuming that classification errors on non-drifting data should follow Gaussian distribution. In a recent work [35], the Adaptive Random Forest (ARF) algorithm was proposed, which extends the random forest tree algorithm with a concept drift detection method, such as ADWIN [31], to decide when to replace an obsolete tree with a new one. A similar work can be found in [65], which uses Hoeffding bound to distinguish concept drift from noise within decision trees.

Besides extending classical methods, many new ensemble methods have been developed to handle concept drift using novel voting techniques. **D**ynamic **W**eighted **M**ajority (DWM) [66] is such an ensemble method that is capable of adapting to drifts with a simple set of weighted voting rules. It manages base classifiers according to the performance of both the individual classifiers and the global ensemble. If the ensemble misclassifies an instance, DWM will train a new base classifier and add it to ensemble. If a base classifier misclassifies an instance, DWM reduces its weight by a factor. When the weight of a base classifier drops below a user defined threshold, DWM removes it from the ensemble. The drawback of this method is that the adding classifier process may be triggered too frequently, introducing performance issues on some occasions, such as when gradual drift occurs. A well-known ensemble method, Learn++NSE [67], mitigates this issue by weighting base classifiers according to their prediction error rate on the latest batch of data. If the error rate of the youngest classifier exceeds 50%, a new classifier will be trained based on the latest data. This method has several other benefits: it

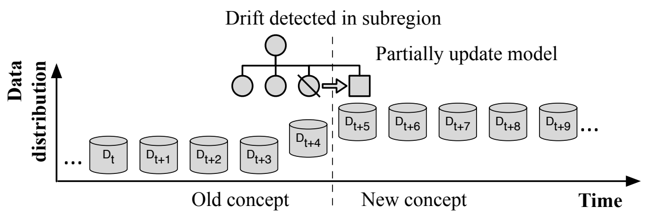


Fig. 15. A decision tree node is replaced with a new one as its performance deteriorates when a concept drift occurs in a subregion.

can easily adopt almost any base classifier algorithm; it does not store history data, only the latest batch of data, which it only uses once to train a new classifier; and it can handle sudden drift, gradual drift, and recurrent drift because underperforming classifiers can be reactivated or deactivated as needed by adjusting their weights. Other voting strategies than standard weighted voting have also been applied to handle concept drift. Examples include hierarchical ensemble structure [68], [69], short term and long term memory [13], [70] and dynamic ensemble sizes [71], [72].

A number of research efforts have been made that focus on developing ensemble methods for handling concept drift of certain types. **A**ccuracy **U**pdate **E**nsemble (AUE2) [73] was proposed with an emphasis on handling both sudden drift and gradual drift equally well. It is a batch mode weighted voting ensemble method based on incremental base classifiers. By doing re-weighting, the ensemble is able react quickly to sudden drift. All classifiers are also incrementally trained with the latest data, which ensures that the ensemble evolves with gradual drift. The Optimal Weights Adjustment (OWA) method [74] achieves the same goal by building ensembles using both weighted instances and weighted classifiers for different concept drift types. The authors of [75] considered a special case of concept drift — class evolution — the phenomenon of class emergence and disappearance. Recurring concepts are handled in [76], [77], which monitor concept information to decide when to reactivate previously stored obsolete models. [78] is another method that handles recurring concepts by refining the concept pool to avoid redundancy.

5.3 Adjusting existing models for regional drift

An alternative to retraining an entire model is to develop a model that adaptively learns from the changing data. Such models have the ability to partially update themselves when the underlying data distribution changes, as shown in Fig. 15. This approach is arguably more efficient than retraining when the drift only occurs in local regions. Many methods in this category are based on the decision tree algorithm because trees have the ability to examine and adapt to each sub-region separately.

In a foundational work [79], an online decision tree algorithm, called **V**ery **F**ast **D**ecision **T**ree classifier (VFDT) was proposed, which is especially tailored for high speed data streams. It uses Hoeffding bound to limit the number of instances required for node splitting. This method has become very popular because of its several distinct advantages: 1) it only needs to process each instance once and does not store instances in memory or disk; 2) the tree itself only consumes a small amount of space and does not grow with the number of instances it processes unless there is new information in the data; 3) the cost of tree maintenance is very low. An extended version, called CVFDT [80], was later proposed to handle concept drift. In CVFDT, a sliding window is maintained to hold the latest data. An alternative sub-tree is trained based on the window and its performance is monitored. If the alternative sub-tree outperforms its original counterpart, it will be used for future prediction and the original obsolete sub-tree will be pruned. VFDTc [81] is another attempt to make improvements to VFDT with several enhancements: the ability to handle numerical attributes; the application of naive Bayes classifiers in tree leaves and the ability to detect and adapt to concept drift. Two node-level drift detection methods were proposed based on monitoring differences between a node and its sub-nodes. The first method uses classification error rate and the second directly checks distribution difference. When a drift is detected on a node, the node becomes a leaf and its descending sub-tree is removed. Later work [82], [83] further extended VFDTc using an adaptive leaf strategy that chooses the best classifier from three options: majority voting, Naive Bayes and Weighted Naive Bayes.

Despite the success of VFDT, recent studies [84], [85] have shown that its foundation, the Hoeffding bound, may not be appropriate for the node splitting calculation because the variables it computes, the information gain, are not independent. A new online decision tree model [86] was developed based on an alternative impurity measure. The paper shows that this measure also reflects concept drift and can be used as a replacement measure in CVFDT. In the same spirit, another decision tree algorithm (IADEM-3) [87] aims to rectify the use of Hoeffding bound by computing the sum of independent random variables, called *relative frequencies*. The error rate of sub-trees are monitored to detect drift and are used for tree pruning.

6 Evaluation, Datasets and Benchmarks

Section 6.1 discusses the evaluation systems used for learning algorithms handling concept drift. Section 6.2 introduces synthetic datasets, which used to simulate specific and controllable types of concept drift. Section 6.3 describes real-world datasets, which used to test the overall performance in a real-life scenario.

6.1 Evaluation Systems

The evaluation systems is an important part for learning algorithms. Some evaluation methodologies used in learning under concept drift have been mentioned in [8]. We enrich this previous research by reviewing the evaluation systems from three aspects: 1) validation methodology, 2) evaluation metrics, and 3) statistical significance, and each evaluation is followed by its computation equation and usage introduction.

Validation methodology refers to the procedure for a learning algorithm to determine which data instances are used as the training set and which are used as the testing set. There are three procedures peculiar to the evaluation for learning algorithms capable of handling concept drift:

1) *holdout*, 2) *prequential*, and 3) *controlled permutation*. In the scenario of a dataset involving concept drift, *holdout* should follow the rule: when testing a learning algorithm at time t, the holdout set represents exactly the same concept at that time t. Unfortunately, it is only applied on synthetic datasets with predefined concept drift times. *Prequential* is a popular evaluation scheme used in streaming data. Each data instance is first used to test the learning algorithm, and then to train the learning algorithm. This scheme has the advantage that there is no need to know the drift time of concepts, and it makes maximum use of the available data. The prequential error is computed based on an accumulated sum of a loss function between the prediction and observed label: $S = \sum_{t=1}^{n} f(\hat{y}_t, y_t)$. There are three prequential error rate estimates: a landmark window (interleaved-test-then-train), a sliding window, and a forgetting mechanism [88]. *Controlled permutation* [89] runs multiple test datasets in which the data order has been permutated in a controlled way to preserve the local distribution, which means that data instances that were originally close to one another in time need to remain close after a permutation. Controlled permutation reduces the risk that their prequential evaluation may produce biased results for the fixed order of data in a sequence.

Evaluation metrics for datasets involving concept drift could be selected from traditional accuracy measures, such as precision/recall in retrieval tasks, mean absolute scaled error in regression, or root mean square error in recommender systems. In addition to that, the following measures should be examined: 1) *RAM-hours* [90] for the computation cost of the mining process; 2) *Kappa statistic* $\kappa = \frac{p-p_{\text{ran}}}{1-p_{\text{ran}}}$ [91] for classification taking into account class imbalance, where p is the accuracy of the classifier under consideration (reference classifier) and p_{ran} is the accuracy of the random classifier; 3) *Kappa-Temporal statistic* $\kappa_{per} = \frac{p-p_{\text{per}}}{1-p_{\text{per}}}$ [92] for the classification of streaming data with temporal dependence, where p_{per} is the accuracy of the persistent classifier (a classifier that predicts the same label as previously observed); 4) *Combined Kappa statistic* $\kappa^{+} = \sqrt{\max(0,\kappa)\max(0,\kappa_{\text{per}})}$ [92], which combines the κ and κ_{per} by taking the geometric average; 5) *Prequential AUC* [93]; and 6) the Averaged Normalized Area Under the Curve (NAUC) values for Precision-Range curve and Recall-Range curve [53], for the classification of streaming data involving concept drift. Apart from evaluating the performance of learning algorithms, the accuracy of the concept drift detection method/algorithm can be accessed according to the following criteria: 1) *true detection rate*, 2) *false detection rate*, 3) *miss detection rate*, and 4) delay of detection [22].

Statistical significance is used to compare learning algorithms on achieved error rates. The three most frequently used statistical tests for comparing two learning algorithms [94], [95] are: 1) McNemar test [96]: denote the number of data instances misclassified by the first classifier and correctly classified by the second classifier by a, and denote b in the opposite way. The McNemar statistic is computed as $M = \text{sign}(a-b) \times (a-b)^2/(a+b)$ to test whether two classifiers perform equally well. The test follows the χ^2 distribution; 2) Sign test: for N data instances, denote the number of data instances misclassified by the first classifier and correctly classified by the second classifier by B and the number of ties by T. Conduct one-sided sign test by computing $p = \sum_{k=B}^{N-T} \binom{N-T}{k} 0.5^k \times 0.5^{N-T-k}$. If p less than a significant level, then the second classifier is better than the first classifier. and 3) Wilcoxon's sign-rank test: For testing two classifiers on N datasets, let $x_{i,1}$ and $x_{i,2}$ $(i = 1, \ldots, N)$ denote the measurements. The number of ties is T and $N_r = N - T$. The test statistic $W = \sum_{i=1}^{N_r}(\text{sign}(x_{i,1} - x_{i,2}) \times R_i)$ where R_i is the rank ordered by $|x_{i,1} - x_{i,2}|$ increasingly. Two classifiers perform equally is rejected if $|W| > W_{\text{critical},N_r}$ (two-sided), where W_{critical,N_r} can be acquired from the statistical table. All three tests are non-parametric. The Nemenyi test [97] is used to compare more than two learning algorithms. It is an appropriate test for comparing all learning algorithms with multiple datasets, based on the average rank of algorithms over all datasets. The Nemenyi test consists of the following: two classifiers are performing differently if the corresponding average ranks differ by at least the critical difference $\text{CD} = q_\alpha\sqrt{k(k+1)/6N}$, where k is the number of learners, N is the number of datasets, and critical values q_α are based on the Studentized range statistic divided by $\sqrt{2}$. Other tests can be used to compare learning algorithms with a control algorithm [97].

6.2 Synthetic datasets

We list several widely used synthetic datasets for evaluating the performance of learning algorithms dealing with concept drift. Since data instances are generated by predefined rules and specific parameters, a synthetic dataset is a good option for evaluating the performance of learning algorithms in different concept drift scenarios. The dataset provider, the number of instances (#Insts.), the number of attributes (#Attrs.), the number of classes (#Cls.), types of drift (Types), sources of drift (Sources), and used by references, are listed in TABLE 3.

6.3 Real-world datasets

In this section, we collect several publicly available real-world datasets, including real-world datasets with synthetic drifts. The dataset provider, the number of instances (#Insts.), the number of attributes (#Attrs.), the number of classes (#Cls.), and used by references, are shown in TABLE 4.

Most of these datasets contain temporal concept drift spanning over different period range - e.g. daily (Sensor [108]), seasonally (Electricity [109]) or yearly (Airlines [104], NOAA weather [67]). Others include geographical (Covertype [106]) or categorical (Poker-Hand [106]) concept drift. Certain datesets, mainly text based, are targeting at specific drift types, such as sudden drift (Email_data [110]), gradural drift (Spam assassin corpus [111]), recurrent drift (Usenet [112]) or novel class (KDDCup'99 [106], ECUE drift dataset 2 [113])

These datasets provide realistic benchmark for evaluating differnent concept drift handling methods. There are, however, two limitations of real world data sets: 1) the groud truth of precise start and end time of drifts is unknown; 2) some real datasets may include mixed drift types. These limitations make it difficult to evaluate methods for

TABLE 3
List of synthetic datasets for performance evaluation of learning under concept drift.

	Dataset	#Insts.	#Attrs.	#Cls.	Types	Sources	Used by references
1	STAGGER [1]	Custom	3	2	Sudden	II	[20], [23], [27], [30], [41], [57], [65], [72], [87], [98], [99]
2	SEA [100]	Custom	3	2	Sudden	II	[2], [5], [13], [20], [27], [32], [35], [51], [57], [58], [63], [65], [67], [73], [76], [99]–[102]
3	Rotating hyperplane [80], [103]	Custom	10	2	Gradual; Incremental	II	[2], [13], [21], [27], [30], [32], [35], [36], [41], [51], [58], [59], [63], [71]–[73], [78], [80], [83], [87], [102]
4	Random RBF [104]	Custom	Custom	Custom	Sudden; Gradual; Incremental	III	[13], [20], [21], [26], [27], [29], [30], [35], [41], [50], [63], [67], [72]–[74], [87], [102], [105]
5	Random Tree [79], [104]	Custom	Custom	Custom	Sudden; Reoccurring	II	[27], [35], [73], [82], [84]–[87]
6	LED [106]	Custom	24	10	Sudden	II	[23], [27], [35], [63], [73], [81], [82], [87], [99], [102]
7	Waveform [106]	Custom	40	3	Sudden	II	[18], [27], [78], [81]–[83], [87], [102]
8	Sine [20]	Custom	2	2	Sudden	II	[20], [21], [26], [29], [72], [107]
9	Circle [20]	Custom	2	2	Gradual	III	[20], [21], [26], [30], [41], [72], [101], [107]
10	Rotating chessboard [67]	Custom	2	2	Gradual	II	[13], [45], [51], [67], [107]

understanding the drift, and could introduce bias when comparing different machine learning models.

7 The Concept Drift Problem in Other Research Areas

We have observed that handling the concept drift problem is not a standalone research subject but has a large number of indirect usage scenarios. In this section, we adopt this new perspective to review recent developments in other research areas that benefit from handling the concept drift problem.

7.1 Class imbalance

Class imbalance is a common problem in stream data mining in addition to concept drift. Research effort has been made to develop effective learning algorithms to tackle both problems at same time. [117] presented two ensemble methods for learning under concept drift with imbalanced class. The first method, Learn++.CDS, is extended from Learn++.NSE and combined with the Synthetic Minority class Oversampling Technique (SMOTE). The second algorithm, Learn++.NIE, improves on the previous method by employing a different penalty constraint to prevent prediction accuracy bias and replacing SMOTE with bagging to avoid oversampling. ESOS-ELM [118] is another ensemble method which uses Online Sequential Extreme Learning Machine (OS-ELM) as a basic classifier to improve performance with class imbalanced data. A concept drift detector is integrated to retrain the classifier when drift occurs. The author then developed another algorithm [119], which is able to tackle multi-class imbalanced data with concept drift. [120] proposed two learning algorithms OOB and UOB, which build an ensemble model to overcome the class imbalance in real time through resampling and time-decayed metrics. [121] developed an ensemble method which handles concept drift and class imbalance with additional true label data limitation.

7.2 Big data mining

Data mining in big data environments faces similar challenges to stream data mining [122]: data is generated at a fast rate (Velocity) and distribution uncertainty always exists in the data, which means that handling concept drift is also crucial in big data applications. Additionally, scalability is an important consideration because in big data environments, a data stream may come in very large and potentially unpredictable quantities (Volume) and cannot be processed in a single computer server. An attempt to handle concept drift in a distributed computing environment was made by [123] in which an Online Map-Reduce Drift Detection Method (OMR-DDM) was proposed, using the combined online error rate of the parallel classification algorithms to identify the changes in a big data stream. A recent study [124] proposed another scalable stream data mining algorithm, called Micro-Cluster Nearest Neighbor (MC-NN), based on nearest neighbor classifier. This method extends the original Micro-Cluster algorithm [125] to adapt to concept drift by monitoring classification error. This micro-cluster algorithm was further extended to a parallel version using the map-reduce technique in [126] and applied to solve the label-drift classification problem where class labels are not known in advance [127].

7.3 Active learning and semi-supervised learning

Active learning is based on the assumption that there is a large amount of unlabeled data but only a fraction of them can be labeled by human effort. This is a common situation in stream data applications, which are often also subject to the concept drift problem. [115] presented a general framework that combines active learning and concept drift adaptation. It first compares different instance-sampling strategies for labeling to guarantee that the labeling cost will be under budget, and that distribution bias will be prevented. A drift adaptation mechanism is then adopted, based on the DDM detection method [20]. In [128], the authors proposed a new active learning algorithm that primarily aims to avoid bias in the sampling process of choosing instances for labeling. They also introduced a memory loss factor to the model, enabling it to adapt to concept drift.

Semi-supervised learning concerns how to use limited true label data more efficiently by leveraging unsupervised techniques. In this scenario, additional design effort is required to handle concept drift. For example, in [129], the authors applied a Gaussian Mixture model to both labeled and unlabeled data, and assigned labels, which has the ability to adapt to gradual drift. Similarly, [99], [130], [131] are all cluster-based semi-supervised ensemble methods that aim to adapt to drift with limited true label data. The latter

TABLE 4
List of real-world datasets for performance evaluation of learning under concept drift.

	Dataset	#Insts.	#Attrs.	#Cls.	Used by references
1	Airlines [104]	539384	7	2	[4], [5], [35], [73], [102], [114], [115]
2	Covertype [106]	581012	54	7	[13], [23], [35], [36], [59], [63], [73], [81]–[83], [86], [87], [102], [115]
3	Electricity [109]	45312	8	2	[4], [5], [13], [20], [23], [26], [29], [31], [35], [36], [57], [63], [72], [73], [78], [86], [87], [101], [102], [115]
4	Poker-Hand [106]	1025010	10	10	[13], [32], [63], [73], [102]
5	NOAA weather [67]	18159	8	2	[2], [4], [13], [67], [68], [78], [105]
6	Sensor [108]	2219803	5	54	[36], [78]
7	KDDCup'99 [106]	494021	41	23	[35], [47], [65], [69], [74], [84], [86], [99], [102]
8	Usenet1 [112]	1500	99	2	[23], [51], [87]
9	Usenet2 [112]	1500	99	2	[23], [87]
10	Email_data [110]	1500	913	2	[45], [76], [77]
11	Spam_data [110]	9324	499	2	[4], [5], [23], [36], [102], [116]
12	Spam assassin corpus [111]	9324	39916	2	[4], [35], [76], [87]
13	ECUE drift dataset 1 [113]	10983	287034	2	[2], [3]
14	ECUE drift dataset 2 [113]	11905	166047	2	[2], [3]

are also able to recognize recurring concepts. In [132], the author adopted a new perspective on the true label scarcity problem by considering the true labeled data and unlabeled data as two independent non-stationary data generating processes. Concept drift is handled asynchronously on these two streams. The SAND algorithm [133], [134] is another semi-supervised adaptive method which detects concept drift on cluster boundaries. There are also studies [90, 91] that focus on adapting to concept drift in circumstances where true label data is completely unavailable.

7.4 Decision Rules

Data-driven decision support systems need to be able to adapt to concept drift in order to make accurate decisions and decision rules is the main technique for this purpose. [102] proposed a decision rule induction algorithm, Very Fast Decision Rules (VFDR), to effectively process stream data. An extended version, Adaptive VFDR, was developed to handle concept drift by dynamically adding and removing decision rules according to their error rate which is monitored by drift detector. Instead of inducing rules from decision trees, [135] proposed another decision rule algorithm based on PRISM [136] to directly induce rules from data. This algorithm is also able to adapt to drift by monitoring the performance of each rule on a sliding window of latest data. [137] also developed an adaptive decision making algorithm based on fuzzy rules. The algorithm includes a rule pruning procedure, which removes obsolete rules to adapt to changes, and a rule recal procedure to adapt to recurring concepts.

This section by no means attempts to cover every research field in which concept drift handling is used. There are many other studies that also consider concept drift as a dual problem. For example, [138] is a dimension reduction algorithm to separate classes based on least squares linear discovery analysis (LSLDA), which is then extended to adapt to drift; [139] considered the concept drift problem in time series and developed an online explicit drift detection method by monitoring time series features; and [140] developed an incremental scaffolding classification algorithm for complex tasks that also involve concept drift.

8 Conclusions: Findings and Future Directions

We summarize the recent developments of concept drift research, and the following important findings can be extracted:

1) Error rate-based and data distribution-based drift detection methods are still playing a dominant role in concept drift detection research, while multiple hypothesis test methods emerge in recent years;
2) Regarding to concept drift understanding, all drift detection methods can answer "When", but very few methods have the ability to answer "How" and "Where";
3) Adaptive models and ensemble techniques have played an increasingly important role in recent concept drift adaptation developments. In contrast, research of retraining models with explicit drift detection has slowed;
4) Most existing drift detection and adaptation algorithms assume the ground true label is available after classification/prediction, or extreme verification latency. Very few research has been conducted to address unsupervised or semi-supervised drift detection and adaptation.
5) Some computational intelligence techniques, such as fuzzy logic, competence model, have been applied in concept drift;
6) There is no comprehensive analysis on real-world data streams from the concept drift aspect, such as the drift occurrence time, the severity of drift, and the drift regions.
7) An increasing number of other research areas have recognized the importance of handling concept drift, especially in big data community.

Based on these findings, we suggest four new directions in future concept drift research:

1) Drift detection research should not only focus on identifying drift occurrence time accurately, but also need to provide the information of drift severity and regions. These information could be utilized for better concept drift adaptation.
2) In the real-world scenario, the cost to acquire true label could be expensive, that is, unsupervised or semi-su-

pervised drift detection and adaptation could still be promising in the future.

3) A framework for selecting real-world data streams should be established for evaluating learning algorithms handling concept drift.
4) Research on effectively integrating concept drift handling techniques with machine learning methodologies for data-driven applications is highly desired.

We hope this paper can provide researchers with state-of-the-art knowledge on concept drift research developments and provide guidelines about how to apply concept drift techniques in different domains to support users in various prediction and decision activities.

Acknowledgments

The work presented in this paper was supported by the Australian Research Council (ARC) under discovery grant DP150101645. We significantly thank Yiliao Song for her help in preparation of datasets and applications shown in Sections 6.

References

[1] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine Learning*, vol. 23, no. 1, pp. 69–101, 1996.

[2] N. Lu, J. Lu, G. Zhang, and R. Lopez de Mantaras, "A concept drift-tolerant case-base editing technique," *Artif. Intell.*, vol. 230, pp. 108–133, 2016.

[3] N. Lu, G. Zhang, and J. Lu, "Concept drift detection via competence models," *Artif. Intell.*, vol. 209, pp. 11–28, 2014.

[4] A. Liu, Y. Song, G. Zhang, and J. Lu, "Regional concept drift detection and density synchronized drift adaptation," in *Proc. 26th Int. Joint Conf. Artificial Intelligence*. Accept, 2017, Conference Proceedings.

[5] A. Liu, G. Zhang, and J. Lu, "Fuzzy time windowing for gradual concept drift adaptation," in *Proc. 26th IEEE Int. Conf. Fuzzy Systems*. IEEE, 2017, Conference Proceedings.

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

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

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

[9] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary environments: a survey," *IEEE Comput. Intell. Mag.*, vol. 10, no. 4, pp. 12–25, 2015.

[10] J. Gama, "A survey on learning from data streams: current and future trends," *Progress in Artificial Intelligence*, vol. 1, no. 1, pp. 45–55, 2012.

[11] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. d. Carvalho, and J. Gama, "Data stream clustering: A survey," *ACM Comput. Surv.*, vol. 46, no. 1, pp. 1–31, 2013.

[12] J. C. Schlimmer and R. H. Granger Jr, "Incremental learning from noisy data," *Machine learning*, vol. 1, no. 3, pp. 317–354, 1986.

[13] V. Losing, B. Hammer, and H. Wersing, "Knn classifier with self adjusting memory for heterogeneous concept drift," in *Proc. 16th Int. Conf. Data Mining*, 2016, Conference Proceedings, pp. 291–300.

[14] I. Žliobaitė and J. Hollmén, "Optimizing regression models for data streams with missing values," *Machine Learning*, vol. 99, no. 1, pp. 47–73, 2014.

[15] S. Amos, "When training and test sets are different: characterizing learning transfer," *Dataset Shift in Machine Learning*, pp. 3–28, 2009.

[16] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognit.*, vol. 45, no. 1, pp. 521–530, 2012.

[17] M. Basseville and I. V. Nikiforov, *Detection of abrupt changes: theory and application*. Prentice Hall Englewood Cliffs, 1993, vol. 104.

[18] A. Dries and U. Rückert, "Adaptive concept drift detection," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 2, no. 5–6, pp. 311–327, 2009.

[19] C. Alippi and M. Roveri, "Just-in-time adaptive classifiers part i: Detecting nonstationary changes," *IEEE Trans. Neural Networks*, vol. 19, no. 7, pp. 1145–1153, 2008.

[20] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Proc. 17th Brazilian Symp. Artificial Intelligence*, ser. Lecture Notes in Computer Science. Springer, 2004, Book Section, pp. 286–295.

[21] L. Bu, C. Alippi, and D. Zhao, "A pdf-free change detection test based on density difference estimation," *IEEE Trans. Neural Networks Learn. Syst.*, vol. PP, no. 99, pp. 1–11, 2016.

[22] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi, "An information-theoretic approach to detecting changes in multi-dimensional data streams," in *Proc. Symp. the Interface of Statistics, Computing Science, and Applications*. Citeseer, 2006, Conference Proceedings, pp. 1–24.

[23] I. Frias-Blanco, J. d. Campo-Avila, G. Ramos-Jimenez, R. Morales-Bueno, A. Ortiz-Diaz, and Y. Caballero-Mota, "Online and non-parametric drift detection methods based on hoeffding's bounds," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 3, pp. 810–823, 2015.

[24] M. Yamada, A. Kimura, F. Naya, and H. Sawada, "Change-point detection with feature selection in high-dimensional time-series data," in *Proc. 23rd Int. Joint Conf. Artificial Intelligence*, 2013, Conference Proceedings, pp. 1827–1833.

[25] J. Gama and G. Castillo, "Learning with local drift detection," in *Proc. 2nd Int. Conf. Advanced Data Mining and Applications*. Springer, 2006, Conference Proceedings, pp. 42–55.

[26] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method," in *Proc. 4th Int. Workshop Knowledge Discovery from Data Streams*, 2006, Conference Paper.

[27] S. Xu and J. Wang, "Dynamic extreme learning machine for data stream classification," *Neurocomputing*, vol. 238, pp. 433–449, 2017.

[28] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.

[29] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognit. Lett.*, vol. 33, no. 2, pp. 191–198, 2012.

[30] K. Nishida and K. Yamauchi, "Detecting concept drift using statistical testing," in *Proc. 10th Int. Conf. Discovery Science*, V. Corruble, M. Takeda, and E. Suzuki, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, Conference Proceedings, pp. 264–269.

[31] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," in *Proc. 2007 SIAM Int. Conf. Data Mining*, vol. 7. SIAM, 2007, Conference Proceedings, p. 2007.

[32] ——, "Adaptive learning from evolving data streams," in *Proc. 8th Int. Symp. Intelligent Data Analysis*. Springer, 2009, Conference Proceedings, pp. 249–260.

[33] A. Bifet, G. Holmes, B. Pfahringer, and R. Gavaldà, "Improving adaptive bagging methods for evolving data streams," in *Proc. 1st Asian Conf. Machine Learning*, ser. Lecture Notes in Computer Science, Z.-H. Zhou and T. Washio, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, Book Section, pp. 23–37.

[34] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà, "New ensemble methods for evolving data streams," in *Proc. 15th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. ACM, 2009, Conference Proceedings, pp. 139–148.

[35] H. M. Gomes, A. Bifet, J. Read, J. P. Barddal, F. Enembreck, B. Pfharinger, G. Holmes, and T. Abdessalem, "Adaptive random forests for evolving data stream classification," *Machine Learning*, 2017.

[36] J. Shao, Z. Ahmadi, and S. Kramer, "Prototype-based learning on concept-drifting data streams," in *Proc. 20th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. 2623609: ACM, 2014, Conference Proceedings, pp. 412–421.

[37] D. Kifer, S. Ben-David, and J. Gehrke, "Detecting change in data streams," in *Proc. 30th Int. Conf. Very Large Databases*, vol. 30. VLDB Endowment, 2004, Conference Proceedings, pp. 180–191.

[38] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Statistical change detection for multi-dimensional data," in *Proc. 13th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. San Jose, California, USA: ACM, 2007, Conference Paper, pp. 667–676.

[39] A. A. Qahtan, B. Alharbi, S. Wang, and X. Zhang, "A pca-based change detection framework for multidimensional data streams," in *Proc. 21th Int. Conf. on Knowledge Discovery and Data Mining*. ACM, 2015, Conference Proceedings, pp. 935–944.

[40] F. Gu, G. Zhang, J. Lu, and C.-T. Lin, "Concept drift detection based on equal density estimation," in *Proc. 2016 Int. Joint Conf. Neural Networks*. IEEE, 2016, Conference Proceedings, pp. 24–30.

[41] L. Bu, D. Zhao, and C. Alippi, "An incremental change detection test based on density difference estimation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, pp. 1–13, 2017.

[42] C. Alippi and M. Roveri, "Just-in-time adaptive classifiers part ii: designing the classifier," *IEEE Trans. Neural Networks*, vol. 19, no. 12, pp. 2053–2064, 2008.

[43] C. Alippi, G. Boracchi, and M. Roveri, "A just-in-time adaptive classification system based on the intersection of confidence intervals rule," *Neural Networks*, vol. 24, no. 8, pp. 791–800, 2011.

[44] ——, "Just-in-time ensemble of classifiers," in *Proc. 2012 Int. Joint Conf. Neural Networks*. IEEE, 2012, Conference Proceedings, pp. 1–8.

[45] ——, "Just-in-time classifiers for recurrent concepts," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 24, no. 4, pp. 620–634, 2013.

[46] W. Heng and Z. Abraham, "Concept drift detection for streaming data," in *Proc. 2015 Int. Joint Conf. Neural Networks*, 2015, Conference Proceedings, pp. 1–9.

[47] Y. Zhang, G. Chu, P. Li, X. Hu, and X. Wu, "Three-layer concept drifting detection in text data streams," *Neurocomputing*, vol. 260, pp. 393–403, 2017.

[48] L. Du, Q. Song, L. Zhu, and X. Zhu, "A selective detector ensemble for concept drift detection," *The Computer Journal*, vol. 58, no. 3, pp. 457–471, 2014.

[49] B. I. F. Maciel, S. G. T. C. Santos, and R. S. M. Barros, "A lightweight concept drift detection ensemble," in *Proc. 27th IEEE Int. Conf. on Tools with Artificial Intelligence*. IEEE, 2015, pp. 1061–1068.

[50] C. Alippi, G. Boracchi, and M. Roveri, "Hierarchical change-detection tests," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 28, no. 2, pp. 246–258, 2017.

[51] S. Yu and Z. Abraham, "Concept drift detection with hierarchical hypothesis testing," in *Proc. 2017 SIAM Int. Conf. Data Mining*. SIAM, 2017, Conference Proceedings, pp. 768–776.

[52] H. Raza, G. Prasad, and Y. Li, "Ewma model based shift-detection methods for detecting covariate shifts in non-stationary environments," *Pattern Recognit.*, vol. 48, no. 3, pp. 659–669, 2015.

[53] S. Yu, X. Wang, and J. C. Principe, "Request-and-reverify: Hierarchical hypothesis testing for concept drift detection with expensive labels," *arXiv preprint arXiv:1806.10131*, 2018.

[54] P. M. Gonçalves Jr, S. G. de Carvalho Santos, R. S. Barros, and D. C. Vieira, "A comparative study on concept drift detectors," *Expert Systems with Applications*, vol. 41, no. 18, pp. 8144–8156, 2014.

[55] F. Pukelsheim, "The three sigma rule," *The American Statistician*, vol. 48, no. 2, pp. 88–91, 1994.

[56] A. Tsymbal, M. Pechenizkiy, P. Cunningham, and S. Puuronen, "Dynamic integration of classifiers for handling concept drift," *Information Fusion*, vol. 9, no. 1, pp. 56–68, 2008.

[57] S. H. Bach and M. Maloof, "Paired learners for concept drift," in *Proc. 8th Int. Conf. Data Mining*, 2008, Conference Proceedings, pp. 23–32.

[58] D. Liu, Y. Wu, and H. Jiang, "Fp-elm: An online sequential learning algorithm for dealing with concept drift," *Neurocomputing*, vol. 207, pp. 322–334, 2016.

[59] D. Han, C. Giraud-Carrier, and S. Li, "Efficient mining of high-speed uncertain data streams," *Applied Intelligence*, vol. 43, no. 4, pp. 773–785, 2015.

[60] S. G. Soares and R. Araújo, "An adaptive ensemble of on-line extreme learning machines with variable forgetting factor for dynamic system prediction," *Neurocomputing*, vol. 171, pp. 693–707, 2016.

[61] B. F. J. Manly and D. Mackenzie, "A cumulative sum type of method for environmental monitoring," *Environmetrics*, vol. 11, no. 2, pp. 151–166, 2000.

[62] N. C. Oza and S. Russell, "Experimental comparisons of online and batch versions of bagging and boosting," in *Proc. 7th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. 502565: ACM, 2001, Conference Proceedings, pp. 359–364.

[63] A. Bifet, G. Holmes, and B. Pfahringer, "Leveraging bagging for evolving data streams," in *Proc. 2010 Joint European Conf. Machine Learning and Knowledge Discovery in Databases*. Springer, 2010, Conference Proceedings, pp. 135–150.

[64] F. Chu and C. Zaniolo, "Fast and light boosting for adaptive mining of data streams," in *Proc. 8th Pacific-Asia Conf. Knowledge Discovery and Data Mining*, H. Dai, R. Srikant, and C. Zhang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, Book Section, pp. 282–292.

[65] P. Li, X. Wu, X. Hu, and H. Wang, "Learning concept-drifting data streams with random ensemble decision trees," *Neurocomputing*, vol. 166, pp. 68–83, 2015.

[66] J. Z. Kolter and M. A. Maloof, "Dynamic weighted majority: An ensemble method for drifting concepts," *Journal of Machine Learning Research*, 2007.

[67] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Trans. Neural Networks*, vol. 22, no. 10, pp. 1517–31, 2011.

[68] X.-C. Yin, K. Huang, and H.-W. Hao, "De2: Dynamic ensemble of ensembles for learning nonstationary data," *Neurocomputing*, vol. 165, pp. 14–22, 2015.

[69] P. Zhang, J. Li, P. Wang, B. J. Gao, X. Zhu, and L. Guo, "Enabling fast prediction for ensemble models on data streams," in *Proc. 17th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. San Diego, California, USA: ACM, 2011, Conference Paper, pp. 177–185.

[70] Y. Xu, R. Xu, W. Yan, and P. Ardis, "Concept drift learning with alternating learners," in *Proc. 2017 Int. Joint Conf. Neural Networks*, 2017, Conference Proceedings, pp. 2104–2111.

[71] L. Pietruczuk, L. Rutkowski, M. Jaworski, and P. Duda, "A method for automatic adjustment of ensemble size in stream data mining," in *Proc. 2016 Int. Joint Conf. Neural Networks*, 2016, Conference Proceedings, pp. 9–15.

[72] S.-C. You and H.-T. Lin, "A simple unlearning framework for online learning under concept drifts," in *Proc. 20th Pacific-Asia Conf. Knowledge Discovery and Data Mining*. Springer, 2016, Conference Proceedings, pp. 115–126.

[73] D. Brzezinski and J. Stefanowski, "Reacting to different types of concept drift: The accuracy updated ensemble algorithm," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 25, no. 1, pp. 81–94, 2014.

[74] P. Zhang, X. Zhu, and Y. Shi, "Categorizing and mining concept drifting data streams," in *Proc. 14th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. Las Vegas, Nevada, USA: ACM, 2008, Conference Paper, pp. 812–820.

[75] Y. Sun, K. Tang, L. L. Minku, S. Wang, and X. Yao, "Online ensemble learning of data streams with gradually evolved classes," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 6, pp. 1532–1545, 2016.

[76] J. Gama and P. Kosina, "Recurrent concepts in data streams classification," *Knowledge and Information Systems*, vol. 40, no. 3, pp. 489–507, 2013.

[77] J. B. Gomes, M. M. Gaber, P. A. Sousa, and E. Menasalvas, "Mining recurring concepts in a dynamic feature space," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 25, no. 1, pp. 95–110, 2014.

[78] Z. Ahmadi and S. Kramer, "Modeling recurring concepts in data streams: a graph-based framework," *Knowledge and Information Systems*, 2017.

[79] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proc. 6th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. ACM, 2000, Conference Proceedings, pp. 71–80.

[80] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proc. 7th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. San Francisco, California: ACM, 2001, Conference Paper, pp. 97–106.

[81] J. Gama, R. Rocha, and P. Medas, "Accurate decision trees for mining high-speed data streams," in *Proc. 9th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. ACM, 2003, Conference Proceedings, pp. 523–528.

[82] H. Yang and S. Fong, "Incrementally optimized decision tree for noisy big data," in *Proc. 1st Int. Workshop Big Data, Streams and Heterogeneous Source Mining Algorithms, Systems, Programming*

Models and Applications. Beijing, China: ACM, 2012, Conference Paper, pp. 36–44.

[83] ——, "Countering the concept-drift problems in big data by an incrementally optimized stream mining model," *Journal of Systems and Software*, vol. 102, pp. 158–166, 2015.

[84] L. Rutkowski, M. Jaworski, L. Pietruczuk, and P. Duda, "Decision trees for mining data streams based on the gaussian approximation," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 108–119, 2014.

[85] L. Rutkowski, L. Pietruczuk, P. Duda, and M. Jaworski, "Decision trees for mining data streams based on the mcdiarmid's bound," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 6, pp. 1272–1279, 2013.

[86] L. Rutkowski, M. Jaworski, L. Pietruczuk, and P. Duda, "A new method for data stream mining based on the misclassification error," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 5, pp. 1048–1059, 2015.

[87] I. Frías-Blanco, J. d. Campo-Ávila, G. Ramos-Jiménez, A. C. P. L. F. Carvalho, A. Ortiz-Díaz, and R. Morales-Bueno, "Online adaptive decision trees based on concentration inequalities," *Knowledge-Based Systems*, vol. 104, pp. 179–194, 2016.

[88] J. Gama, R. Sebastião, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Machine Learning*, vol. 90, no. 3, pp. 317–346, 2012.

[89] I. Žliobaitė, "Controlled permutations for testing adaptive learning models," *Knowledge and Information Systems*, vol. 39, no. 3, pp. 565–578, 2014.

[90] A. Bifet, G. Holmes, B. Pfahringer, and E. Frank, "Fast perceptron decision tree learning from evolving data streams," in *Proc. 14th Pacific-Asia Conf. Knowledge Discovery and Data Mining*, M. J. Zaki, J. X. Yu, B. Ravindran, and V. Pudi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, Book Section, pp. 299–310.

[91] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[92] I. Žliobaitė, A. Bifet, J. Read, B. Pfahringer, and G. Holmes, "Evaluation methods and decision theory for classification of streaming data with temporal dependence," *Machine Learning*, vol. 98, no. 3, pp. 455–482, 2015.

[93] D. Brzezinski and J. Stefanowski, "Prequential auc for classifier evaluation and drift detection in evolving data streams," in *Proc. 3rd Int. Workshop New Frontiers in Mining Complex Patterns*, A. Appice, M. Ceci, C. Loglisci, G. Manco, E. Masciari, and Z. W. Ras, Eds. Cham: Springer International Publishing, 2014, Book Section, pp. 87–101.

[94] A. Bifet, G. d. F. Morales, J. Read, G. Holmes, and B. Pfahringer, "Efficient online evaluation of big data stream classifiers," in *Proc. 21th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. Sydney, NSW, Australia: ACM, 2015, Conference Paper, pp. 59–68.

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

[96] Q. McNemar, "Note on the sampling error of the difference between correlated proportions or percentages," *Psychometrika*, vol. 12, no. 2, pp. 153–157, 1947.

[97] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, no. Jan, pp. 1–30, 2006.

[98] J. Z. Kolter and M. A. Maloof, "Using additive expert ensembles to cope with concept drift," in *Proc. 22nd Int. Conf. Machine Learning*. Bonn, Germany: ACM, 2005, Conference Paper, pp. 449–456.

[99] X. Wu, P. Li, and X. Hu, "Learning from concept drifting data streams with unlabeled data," *Neurocomputing*, vol. 92, pp. 145–155, 2012.

[100] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-scale classification," in *Proc. Seventh ACM Int. Conf. Knowledge Discovery and Data Mining*. 502568: ACM, 2001, Conference Proceedings, pp. 377–382.

[101] R. Fok, A. An, and X. Wang, "Mining evolving data streams with particle filters," *Comput. Intell.*, vol. 33, no. 2, pp. 147–180, 2017.

[102] P. Kosina and J. Gama, "Very fast decision rules for classification in data streams," *Data Mining and Knowledge Discovery*, vol. 29, no. 1, pp. 168–202, 2015.

[103] H. Wang, W. Fan, P. S. Yu, and J. Han, "Mining concept-drifting data streams using ensemble classifiers," in *Proc. 9th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. Washington, D.C.: ACM, 2003, Conference Paper, pp. 226–235.

[104] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "Moa: Massive online analysis," *Journal of Machine Learning Research*, vol. 99, pp. 1601–1604, 2010.

[105] V. M. Souza, D. F. Silva, J. Gama, and G. E. Batista, "Data stream classification guided by clustering on nonstationary environments and extreme verification latency," in *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 2015, pp. 873–881.

[106] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[107] M. Harel, S. Mannor, R. El-Yaniv, and K. Crammer, "Concept drift detection through resampling," in *Proc. 31st Int. Conf. Machine Learning*, 2014, Conference Proceedings, pp. 1009–1017.

[108] X. Zhu, "Stream data mining repository," 2010. [Online]. Available: http://www.cse.fau.edu/~xqzhu/stream.html

[109] M. Harries and N. S. Wales, "Splice-2 comparative evaluation: Electricity pricing," 1999.

[110] I. Katakis, G. Tsoumakas, and I. Vlahavas, "Tracking recurring contexts using ensemble classifiers: an application to email filtering," *Knowledge and Information Systems*, vol. 22, no. 3, pp. 371–391, 2009.

[111] I. Katakis, G. Tsoumakas, E. Banos, N. Bassiliades, and I. Vlahavas, "An adaptive personalized news dissemination system," *Journal of Intelligent Information Systems*, vol. 32, no. 2, pp. 191–212, 2008.

[112] I. Katakis, G. Tsoumakas, and I. P. Vlahavas, "An ensemble of classifiers for coping with recurring contexts in data streams," in *18th European Conf. Artificial Intelligence*, 2008, Conference Proceedings, pp. 763–764.

[113] S. J. Delany, P. Cunningham, A. Tsymbal, and L. Coyle, "A case-based technique for tracking concept drift in spam filtering," *Knowledge-Based Systems*, vol. 18, no. 4–5, pp. 187–195, 2005.

[114] L.-Y. Wang, C. Park, K. Yeon, and H. Choi, "Tracking concept drift using a constrained penalized regression combiner," *Comput. Stat. Data Anal.*, vol. 108, pp. 52–69, 2017.

[115] I. Zliobaite, A. Bifet, B. Pfahringer, and G. Holmes, "Active learning with drifting streaming data," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 25, no. 1, pp. 27–39, 2014.

[116] G. Song, Y. Ye, H. Zhang, X. Xu, R. Y. K. Lau, and F. Liu, "Dynamic clustering forest: An ensemble framework to efficiently classify textual data stream with concept drift," *Information Sciences*, vol. 357, pp. 125–143, 2016.

[117] G. Ditzler and R. Polikar, "Incremental learning of concept drift from streaming imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2283–2301, 2013.

[118] B. Mirza, Z. Lin, and N. Liu, "Ensemble of subset online sequential extreme learning machine for class imbalance and concept drift," *Neurocomputing*, vol. 149, pp. 316–329, 2015.

[119] B. Mirza and Z. Lin, "Meta-cognitive online sequential extreme learning machine for imbalanced and concept-drifting data classification," *Neural Networks*, vol. 80, pp. 79–94, 2016.

[120] S. Wang, L. L. Minku, and X. Yao, "Resampling-based ensemble methods for online class imbalance learning," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 5, pp. 1356–1368, 2015.

[121] E. Arabmakki and M. Kantardzic, "Som-based partial labeling of imbalanced data stream," *Neurocomputing*, vol. 262, pp. 120–133, 2017.

[122] A. Katal, M. Wazid, and R. H. Goudar, "Big data: Issues, challenges, tools and good practices," in *Proc. 6th Int. Conf. Contemporary Computing (IC3)*, 2013, Conference Proceedings, pp. 404–409.

[123] A. Andrzejak and J. B. Gomes, "Parallel concept drift detection with online map-reduce," in *Proc. 12th Int. Conf. Data Mining Workshops*, 2012, Conference Proceedings, pp. 402–407.

[124] M. Tennant, F. Stahl, O. Rana, and J. B. Gomes, "Scalable real-time classification of data streams with concept drift," *Future Generation Computer Systems*, vol. 75, pp. 187–199, 2017.

[125] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams," in *Proc. 29th Int. Conf. Very Large Databases*, vol. 29. VLDB Endowment, 2003, Conference Proceedings, pp. 81–92.

[126] X. Song, H. He, S. Niu, and J. Gao, "A data streams analysis strategy based on hoeffding tree with concept drift on hadoop system," in *Proc. 4th Int. Conf. Advanced Cloud and Big Data*, 2016, Conference Proceedings, pp. 45–48.

[127] V. Nguyen, T. D. Nguyen, T. Le, S. Venkatesh, and D. Phung, "One-pass logistic regression for label-drift and large-scale clas-

sification on distributed systems," in *Proc. 16th Int. Conf. Data Mining*, 2016, Conference Proceedings, pp. 1113–1118.

[128] W. Chu, M. Zinkevich, L. Li, A. Thomas, and B. Tseng, "Unbiased online active learning in data streams," in *Proc. 17th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. San Diego, California, USA: ACM, 2011, Conference Paper, pp. 195–203.

[129] G. Ditzler and R. Polikar, "Semi-supervised learning in non-stationary environments," in *Proc. 2011 Int. Joint Conf. Neural Networks*, 2011, Conference Proceedings, pp. 2741–2748.

[130] M. J. Hosseini, A. Gholipour, and H. Beigy, "An ensemble of cluster-based classifiers for semi-supervised classification of non-stationary data streams," *Knowledge and Information Systems*, vol. 46, no. 3, pp. 567–597, 2015.

[131] P. Zhang, X. Zhu, J. Tan, and L. Guo, "Classifier and cluster ensembles for mining concept drifting data streams," in *Proc. 10th Int. Conf. Data Mining*, 2010, Conference Proceedings, pp. 1175–1180.

[132] S. Chandra, A. Haque, L. Khan, and C. Aggarwal, "An adaptive framework for multistream classification," in *Proc. 25th ACM Int. on Conf. Information and Knowledge Management*. Indianapolis, Indiana, USA: ACM, 2016, Conference Paper, pp. 1181–1190.

[133] A. Haque, L. Khan, M. Baron, B. Thuraisingham, and C. Aggarwal, "Efficient handling of concept drift and concept evolution over stream data," in *Proc. 32nd Int. Conf. Data Engineering*, 2003, Conference Proceedings, pp. 481–492.

[134] A. Haque, L. Khan, and M. Baron, "Sand: Semi-supervised adaptive novel class detection and classification over data stream," in *30th AAAI Conf. Artificial Intelligence*, 2016, Conference Proceedings, pp. 1652–1658.

[135] T. Le, F. Stahl, M. M. Gaber, J. B. Gomes, and G. D. Fatta, "On expressiveness and uncertainty awareness in rule-based classification for data streams," *Neurocomputing*, vol. 265, pp. 127–141, 2017.

[136] J. Cendrowska, "Prism: An algorithm for inducing modular rules," *Int. J. Man Mach. Stud.*, vol. 27, no. 4, pp. 349–370, 1987.

[137] M. Pratama, S. G. Anavatti, M. Joo, and E. D. Lughofer, "pclass: An effective classifier for streaming examples," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 2, pp. 369–386, 2015.

[138] Y.-R. Yeh and Y.-C. F. Wang, "A rank-one update method for least squares linear discriminant analysis with concept drift," *Pattern Recognit.*, vol. 46, no. 5, pp. 1267–1276, 2013.

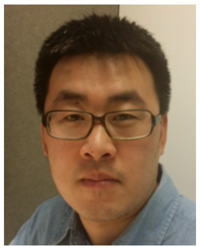
[139] R. C. Cavalcante, L. L. Minku, and A. L. I. Oliveira, "Fedd: Feature extraction for explicit concept drift detection in time series," in *Proc. 2016 Int. Joint Conf. Neural Networks*, 2016, Conference Proceedings, pp. 740–747.

[140] M. Pratama, J. Lu, E. Lughofer, G. Zhang, and S. Anavatti, "Scaffolding type-2 classifier for incremental learning under concept drifts," *Neurocomputing*, vol. 191, pp. 304–329, 2016.

Jie Lu is a Distinguished Professor, Director of Centre for Artificial Intelligence, and Associate Dean Research with in the Faculty of Engineering and Information Technology at the University of Technology Sydney. Her research interests lie in the area of decision support systems, concept drift, fuzzy transfer learning, and recommender systems. She has published 10 research books and 400 papers, won 8 Australian Research Council discovery grants and 20 other grants. She serves as Editor-In-Chief for KBS and IJCIS, and delivered 16 keynotes in international conferences.

Anjin Liu is a Postdoctoral Research Associate in the A/DRsch Centre for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney. He received the BIT degree (Honour) at the University of Sydney in 2012. His research interests include concept drift detection, adaptive data stream learning, multi-stream learning, machine learning and big data analytics

Fan Dong is Research Fellow of Centre for Artificial Intelligence, University of Technology Sydney. He received the dual Ph.D. degree from University of Technology Sydney and Beijing Institute of Technology in 2018. His research interests include concept drift detection, adaptive learning under concept drift and data stream mining.



Feng Gu is a Ph.D. candidate at the Faculty of Engineering and Information Technology, the University of Technology Sydney, NSW, Australia. He received bachelors degree of software engineering at Zhejiang University, China, in 2012. His research interests include stream data mining, adaptive learning under concept drift and evolving data.

João Gama is an Associate Professor at the University of Porto, Portugal. He is also a senior researcher and member of the board of directors of the Laboratory of Artificial Intelligence and Decision Support (LIAAD), a group belonging to INESC Porto. He serves as the member of the Editorial Board of Machine Learning Journal, Data Mining and Knowledge Discovery, Intelligent Data Analysis and New Generation Computing. His main research interest is in knowledge discovery from data streams and evolving data. He has published more than 200 papers and a recent book on Knowledge Discovery from Data Streams. He has extensive publications in the area of data stream learning.

Guangquan Zhang is an Associate Professor, and the Director of Decision System and e-Service Intelligence (DeSI) lab with in the Centre for Artificial Intelligence, in the Faculty of Engineering and Information Technology at the University of Technology Sydney. His main research interests lie in the area of uncertain information processing, fuzzy decision making, concept drift and fuzzy transfer learning. He has published 4 monographs and over 400 papers in refereed journals, conference proceedings and book chapters. He has won 7 Australian Research Council discovery grants and guest edited many special issues for international journals.