Overview

Classification systems in dynamic environments: an overview

Felipe Azevedo Pinage,[1]* Eulanda Miranda dos Santos[1] and João Manuel Portela da Gama[2]

Data mining and machine learning algorithms can be employed to perform a variety of tasks. However, since most of these problems may depend on environments that change over time, performing classification tasks in dynamic environments has been a challenge in data mining research domain in the last decades. Currently, in the literature, the most common strategies used to detect changes are based on accuracy monitoring, which relies on previous knowledge of the data in order to identify whether or not correct classifications are provided. However, such a feedback can be infeasible in practical problems. In this work, we present a comprehensive overview of current machine learning/data mining approaches proposed to deal with dynamic environments problems. The objective is to highlight the main drawbacks and open issues, as well as future directions and problems worthy of investigation. In addition, we provide the definitions of the main terms used to represent this problem in the literature, such as concept drift and novelty detection.

How to cite this article:
WIREs Data Mining Knowl Discov 2016, 6:156–166. doi: 10.1002/widm.1184

INTRODUCTION

The design of classification systems robust to deal with dynamic environments has attracted considerable attention in machine learning and data mining. In real-world applications, some changes occur in the environments along with time. This problem, named as concept drift, has a direct impact on the performance of classification systems, since the classification systems tend to decrease their effectiveness, i.e., high recognition rates may not be achieved.

Some real-world problems may present dynamic environments and the application of adaptive classification systems is very important. For example, in filtering anti-spam, the features that characterize a spam can evolve over time. Besides, important features used to classify spam may be irrelevant in the future.[1] Thus, the anti-spam filter needs a mechanism to detect changes in order to adapt itself to new patterns of spam. In the literature, we can find detection methods to different applications, such as e-mail filtering,[2] fraud detection,[3] intrusion detection in computer networks,[4] and topic ranking in twitter.[5]

There are many studies in the literature that propose new methods to design classification systems that are able to detect changes and adapt its knowledge without compromise the system accuracy. However, several methods have focused on either detecting changes based on monitoring the success rate of the system or retraining classifiers without explicitly detecting changes. In the first case, it is necessary to reduce the performance of the system suddenly in order to detect changes, which certainly implies damage to the system. In addition, these approaches rely on the assumption that there is an oracle able to indicate whether or not the classification system predicts the correct label for the unknown samples. The main disadvantage of the second group of approaches is the computational cost involved, since the system updates constantly, even if changes do not occur.

*Correspondence to: felipepinage@icomp.ufam.edu.br

[1]Institute of Computing, Federal University of Amazonas, Manaus, Brazil

[2]Institute for Systems Engineering and Computers, Faculty of Economics, University of Porto, Porto, Portugal

Conflict of interest: The authors have declared no conflicts of interest for this article.

Moreover, in the literature related to classification systems in dynamic environment, many different terms are used coming from different fields of research for the same problem, or the same terms for different problems. In this way, such a diversity of terms may increase the difficulty of understanding each term correctly. For instance, terms such as Novelty Detection, Concept Drift, and One-Class Classification.

This work presents an overview on classification systems applied to dynamic environments. The objective is to explain the main concepts and terms widely used in the literature. In addition, the most commonly encountered strategies are summarized and discussed in order to highlight the main drawbacks detected on current solutions. Finally, we present some issues worthy of investigation.

This overview is organized as follows: Second section presents the most common terms used in research dealing with dynamic environment problems. The main events that occur in data streams are described in third section. Then, in the fourth section, the most popular and recently proposed learning methods used to solve these problems are discussed. Last section concludes this overview by pointing out possible open issues.

UNDERSTANDING THE PROBLEMS

Several different concepts may be related to changing environmental problems. In this section, the main concepts are presented aiming on describing their definitions and the relation among each other.

Novelty Detection

One of the main critical challenges in the literature when using classification systems in changing environments is called novelty detection. According to Miljkovic,[6] novelty refers to abnormal patterns embedded in a large amount of normal data, or when the data do not fit the expected behavior. Traditionally, novelty detection is related to statistical approaches for outlier detection, which can be based on monitoring the unconditional probability distribution.[1,7] According to Kuncheva,[1] in case of unlabeled data, one simple statistical scheme to detect novelties works on comparing the probability estimate $p(x)$ to a fixed threshold θ, i.e., when $p(x) > \theta$, x is classified based on knowledge obtained during the training step. Otherwise, x may be assumed as a novel object.

In recent work,[8] the authors advocate that often in novelty detection problems only few labels or even none are available. In this way, it is possible to use semi-supervised or unsupervised classification systems. In the context of novelty detection using supervised learning, there available only the knowledge about normal patterns. Thus, the novelties are assumed to be those data not clustered with the normal data, but which are spread in low-density regions.[9,10] Moreover, according to Faria et al.,[11] novelty detection aims to detect emergent patterns and then incorporate them into the normal model. Finally, it is important to distinguish novelty detection from outlier detection, given that the first is related to data distribution and system accuracy decreasing.[7]

Concept Drift

In the machine learning community, the term concept is employed to define the overall distribution of the data used to perform classification, regression, or unsupervised problems in a certain point of time.[12] Usually, it is expected an existence of a stable underlying data generating mechanism, i.e., the concept does not evolve over time. However, as mentioned in the introduction, it has been shown that the learning context (target environment) changes over time in many real-world problems. In this case, researchers have referred to this problem as concept drift.

Therefore, concept drift occurs when data distributions change over time unexpectedly and in unpredictable ways. Widmer[13] defines concept drift as follows: 'In many real-world domains, the context in which some concepts of interest depend may change, resulting in more or less abrupt and radical changes in the definition of the target concept. The change in the target concept is known as concept drift.'

The change of underlying unknown probability distribution, which represents the concept drift, can be defined such as $P_j(x, \omega) \neq P_k(x, \omega)$, where x represents a data instance, ω represents the class, and the change occurs from time t_j to time t_k, where $t_j < t_k$. According to Hee Ang et al.,[14] this means that, in a changing environment, an optimal prediction function for $P_j(x, \omega)$ is no longer optimal for $P_k(x, \omega)$. Moreover, concept drifts are the changes that may compromise the classification accuracy.

Hence, a very important challenge arises when it is observed that the learning concepts start to drift. According to Bose et al.,[15] concept drift solutions should focus on two main directions: how to detect drifts (changes) and how to adapt the predictive model to drifts. These are no trivial tasks because there are different types of changes and the classification system should be robust to ones and sensitive to others. Many algorithms have been developed to

handle concept drift and some of them will be described in the section *Current Solutions for Concept Drifts*.

In addition, the concept drift is the consequence of context change, which is directly related to the features and can be either hidden (called hidden contexts) or explicit. Harries and Sammut[16] define context as follows: 'Context is any attribute whose values tend to be stable over contiguous intervals of time when a hidden attribute occurs.'

One-Class Classification

Here, only one class is well sampled (normal data), while samples from other classes (abnormal data) are not available. In One-Class Classification, we know only the probability density $p(x|\omega_T)$, where x represents a data instance and ω_T is the normal class. According to Le et al.,[17] real-world applications are easier and cheaper collecting normal data, while the abnormal data are expensive and are not always available. The problem focus on making a description of a normal set of objects, as well as on detecting objects that do not belong to the learned description.

MAIN EVENTS IN DATA STREAMS

As in data streams arrive a massive quantity of examples, it is very common that the occurrence of events become a challenge in the classification tasks. Adae and Berthold[18] say that an *event* can be any irregularity in the data behavior, i.e., the current observations are not related to previous concepts. These events may be divided in two categories: (1) anomalies and (2) drifts.

Anomalies

According to Chandola et al.,[19] anomalies refers to patterns in data that do not conform to expect behavior but they are not incorporated to the normal model after their detection, because they do not represent a new concept. In literature, the most common types of anomalies mentioned are: noise and rare event.

- Noise: Meaningless data that cannot be interpreted correctly and should not be taken into account on classification tasks, but can be used to improve system robustness for the underlying distribution.[20] A difficult problem in handling concept drift is distinguishing between true concept drift and noise. An ideal learner should combine robustness to noise and sensitivity to concept drift as much as possible.[21]
- Rare event: This is classified as an outlier. Assuming that these events are rare, they can be dealt with as abnormal data but discarded by the system. However, a concise group of examples classified as outliers should be considered as a novelty, since those events are no longer rare.[22]

Drifts

Here, there are the types of concept drifts. Changes may compromise the classification accuracy due to the appearance of new concepts (gradual, incremental, and abrupt) or reappearance of previous concepts (recurring concepts).

- Gradual: Here, a concept $C1$ is gradually replaced by a new concept $C2$.[23,24] Therefore, the new concept takes over almost imperceptibly, leading to a period of uncertainty between two stable states. Since the change occurs between two consecutive time points t_1 and t_2, i.e., there is a subspace of the whole instance space whose concepts are different from the remaining data, because both concepts coexist in such a period of mixed distributions. These changes are usually detected through strategies based on time windows that scan (sweep) the training data.
- Incremental: When the concept evolves slowly over time. Some researchers use the terms incremental and gradual as the same type of change. However, according to Brzezinski[25] and Bose et al.,[15] a change is assumed to be incremental when variables slowly change their values over time, but there are no examples of two distributions mixed. The old concept disappears slowly until be completely replaced by the new concept.
- Abrupt: Also called sudden concept drift, it occurs when the source distribution at time t, denoted S_t, is suddenly replaced by a different distribution in S_{t+1}. In other words, a concept $C1$ is substituted by concept $C2$, and $C1$ disappears exactly at the moment of this replacement.[2] Several methods designed to cope with abrupt changes use falling confidence of classification to detect a change occurrence.
- Recurring concepts: Concepts that disappear but may reappear in the future, i.e., temporary changes, which are reverted after some time. This happens especially due to the fact that several hidden contexts may reappear at irregular time intervals.[25] Recurring concepts can occur

in both gradual and abrupt ways. Gomes et al.[26] assume that when a concept reappears, normally the context previously associated with it also reappears.

In the occurrence of any type of changes, there are many strategies to treat them. These solutions are discussed in the next section.

CURRENT SOLUTIONS FOR CONCEPT DRIFTS

This section presents the most relevant and recent studies whose focus is on handling concept drift. These studies are divided into three categories: drift detectors, ensemble classifiers, and unsupervised methods. The drift detectors are described first. Afterwards, ensemble-based methods are discussed. Finally, the unsupervised methods are described.

Drift Detectors

Drift detectors is a category of methods, which employ statistical tests to monitor whether or not the class distribution is stable over time and to reset the decision model when a concept drift is detected. All drift detectors discussed in this section are based on single classifiers. Consequently, the decision model must be updated after drift detection. In addition, these algorithms usually detect drifts based on online classification error rate.

Strategies based on classification error are motivated by probably approximately correct (PAC) learning model,[27] which assumes that, if the distribution of the examples is stationary, the error rate of the learning algorithm will decrease as the number of examples increases. Thus, an increase of this error rate suggests a change in class distribution and a probably outdated current model.

The Drift Detection Method (DDM), proposed by Gama et al.,[28] defines two thresholds: warning level and drift level. The first level is reached if condition (1) is attained, while the drift level is achieved when condition (2) is satisfied. The values p and s represent the error rate of the learning algorithm and its standard deviation, respectively. The registers $p_{\min}$ and $s_{\min}$ are defined during the training phase, and are updated if after each incoming example i, the current register $p_i + s_i$ is lower than $p_{\min} + s_{\min}$.

$$p_i + s_i \geq p_{\min} + 2*s_{\min} \quad (1)$$

$$p_i + s_i \geq p_{\min} + 3*s_{\min} \quad (2)$$

For instance, given that the error rate of the actual model reaches the warning level at example k_n, while the drift level is reached at example k_p, in DDM, it is assumed that the concept changes at k_p and a new context is declared between k_n and k_p. In the adaptation process, the new decision model should be generated using only the new context, i.e., the same classifier is retrained using examples stored between k_n and k_p.

The main drawback to this strategy is that DDM is critically affected by the velocity of the changes. Consequently, if a very slow gradual change takes place, the system will not be able to detect it. In order to overcome this drawback, Baena et al.[29] proposed the Early Drift Detection Method (EDDM). EDDM relies on the assumption that the distance between two consecutive errors will increase by improving the predictions of the decision model.

Similar to DDM, two thresholds are defined when using EDDM, also called warning level and drift level. EDDM calculates the distance (p') between two consecutive errors and their standard deviation (s'), and stores the maximum values of (p') and (s') to register the point where the distance between two errors is maximum ($p'_{\max} + 2 * s'_{\max}$). The warning level is reached when the result of the formula (3) is lower than α (set to 0,95), and the drift level is reached when the same formula (3) is lower than β (set to 0,9).

$$(p'_i + 2*s'_i)/(p'_{\max} + 2*s'_{\max}) \quad (3)$$

On the one hand, the thresholds must be used to monitor the decrease on the distance between two errors. On the other, the adaptation process is basically the same as used in DDM, i.e., the decision model is updated using only the new context, ranging from warning and drift levels.

EDDM starts the search for concept drifts after calculating 30 classification errors, due to the fact that the authors intended to estimate the distance distribution between two consecutive errors in order to compare it with further distributions. The results attained by EDDM were better than the results provided by DDM in some databases. In addition, EDDM was able to early detect gradual changes even when the changes were very slow. Even though, EDDM was not robust enough to noisy dataset.

Another important drift detector is the Detection Method Using Statistical Testing (STEPD), proposed by Nishida and Yamauchi.[30] STEPD is based on two accuracies: the recent one and the overall one. The recent accuracy is calculated using a recent set of examples, called W, while the overall accuracy is calculated using the whole set of

examples, except for the recent W examples. This detector relies on two assumptions: (a) if the accuracy of a classifier for recent W examples is equal to the overall accuracy, then the target concept is stationary and (b) a significant decrease on recent accuracy suggests concept drift.

STEPD compares the measure T, defined in Eq. (4), to the percentile of standard normal distribution in order to obtain the observed level (P) of significance. Moreover, it defines two levels of significance as thresholds (here, also called warning and drift levels). The algorithm starts by storing examples when P is lower than the warning level and retrain the classifier when P is lower than the drift level using examples stored from the warning to the drift level.

$$T(r_0, r_r, n_0, n_r) = \frac{\left|\frac{r_0}{n_0} - \frac{r_r}{nr}\right| - 0.5\left(\frac{1}{n_o} + \frac{1}{n_r}\right)}{\sqrt{\hat{p}\left(1-\hat{p}\right)\left(\frac{1}{n_o} + \frac{1}{n_r}\right)}} \qquad (4)$$

where, r_0 is the number of correct classifications considering overall examples n_0, except the recent W examples, r_r is the number of correct classifications among W examples n_r, and $\hat{p} = (r_0 + r_r)/(n_0 + n_r)$.

According to Nishida and Yamauchi,[30] in comparison to EDDM and DDM, STEPD presented the highest performances for abrupt changes and noises. However, EDDM detected gradual changes better than STEPD, while DDM successfully detected abrupt changes, but produced the slowest detection speed.

It is important to observe that all drift detectors described in this section receive the incoming data in a stream and are based on single classifiers, which are replaced after a detection of concept drift. Generally, handling concept drift using single classifiers is not very effective especially due to the following two reasons. First, after training a classifier, its knowledge will not adapt to changes unless the classifier is retrained. Second, if the classifier is retrained after each time period, it will forget the previously learned concepts, which may lead to catastrophic forgetting,[31] especially when the environment presents recurring changes.

Classifier ensembles have been investigated in the literature as a strategy for avoiding the single classifier problems when coping with changing environment problems. In some works,[32–36] the authors conclude that ensemble classifiers present superior performances than single classifier. These methods update their knowledge base by adding, removing, or updating classifiers, as discussed in the next section.

Ensemble Classifiers

The majority of the algorithms based on ensemble classifiers are passive approaches. The idea of passive approaches is to update the system constantly using new input data without detecting changes, i.e., the detection mechanism is implicit in the method. These methods build ensembles by adding new members as new datasets are provided. The new classifiers replace ensemble members according to different strategies. One possibility is to remove the oldest member, as was done in Streaming Ensemble Algorithm (SEA).[37] Another option is removing the poorest performing member, as in Dynamic Weighted Majority (DWM).[38]

DWM works as follows: (1) if the global prediction is incorrect, then DWM adds a new member; (2) the weight is decreased of members whose prediction is incorrect; (3) DWM removes members with a weight less than the threshold θ.

Sidhu et al[39] proposed an online ensemble approach called Early Dynamic Weighted Majority (ERDWM). The weighted strategy is undertaken using three options: (1) decrease the weight of members whose prediction is incorrect; (2) increase the weight of members whose local prediction is correct but global prediction is incorrect; and (3) no weight update when both local and global predictions are correct.

ERDWM focus on the highest performing classifier members in order to reduce the chances of incorrect global prediction, which is the main problem detected in DWM. In addition, ERDWM reduces the need of creating new classifier members and consequently, it decreases time and memory resources requirements. Even though, Sidhu et al.[39] conclude that ERDWM does not outperform EDDM in terms of memory and execution time. On the other hand, ERDWM is better in retaining previous knowledge to support predictions.

A more recent work using passive approach is presented by Brzezinski and Stefanowski.[23] They propose a method called Accuracy Updated Ensemble (AUE2), which presents a mechanism to achieve good predictions in occurrence of different types of drift at relatively low computational costs. In AUE2, a new ensemble classifier member is created after each incoming data chunk. Thus, the new member replaces the poorest performing member. The remaining ensemble members are updated according to their accuracy. The weighting formula (5) is used to combine information about classifiers accuracy and current class distribution.

$$\omega_{ij} = \frac{1}{MSE_r + MSE_{ij} + \epsilon} \qquad (5)$$

where MSE_{ij} denotes the estimate of the prediction error of each classifier on each data chunk, while MSE_r represents the mean square error of a randomly predicting classifier. MSE_r is used as a reference point to the current class distribution. Finally, ϵ indicates a small positive value added to avoid division by zero. Equation (5) is used to update the weight of the remaining classifier members.

In addition, Brzezinski and Stefanowski[23] assume that the most recent incoming data chunk B_i is the best representation of the current and near-future data distribution. Consequently, a classifier C', trained on B_i, is assumed to be the best possible (or perfect) classifier. The weight of C' is assigned as follows

$$\omega_{C'} = \frac{1}{MSE_r + \epsilon} \tag{6}$$

According to the authors, AUE2 achieves higher classification accuracy than its predecessors (Accuracy Weighted Ensemble—AWE and SEA) in the presence of slow gradual drifts. Besides, ensemble members can be retrained, which makes AUE2 less dependent on chunk size and it allows using smaller chunks without compromise its accuracy. Finally, to solve the problem of memory usage, AUE2 sets a memory usage limit (threshold) that, when exceeded, decreases the amount of classifier members.

The works summarized so far have focused on dealing with concept drift by either explicitly detecting drifts using single classifiers or implicitly detecting drifts using ensemble of classifiers. However, as mentioned before, approaches based on single classifiers may be prone to catastrophic forgetting. In terms of passive approaches, their main drawback is the high computational cost involved, whatever the learning method used, ensemble or single classifiers, since the system updates constantly even if changes do not occur. An alternative to these previous methods is to use classifier ensembles to detect drifts explicitly.

Following this idea, Minku and Yao[12] proposed the Diversity for Dealing with Drift (DDD). This method processes each example at a time and maintains ensembles with different diversity levels in order to deal with concept drift. Basically, DDD generates a pool of classifiers using online bagging,[40] as follows: whenever a training example is available, it is presented N (defined by Poisson distribution) times for each base learner, and the classification is performed by unweighted majority vote, as in offline bagging. Then, the classifier members are separated into two subsets of classifiers: (1) low diversity and (2) high diversity classifiers.

It is important to note that there is no generally accepted formal definition of diversity yet. The researchers still investigate how diversity must be measured and the real meaning of this measure. Johansson et al.[41] suggest that diversity is almost an axiom based on the assumption that the classifier members must be diverse to assure that the ensemble is most likely to present good generalization. Since there is no consensus about which proposed diversity measure is the best one, DDD measures diversity using Q statistic,[42] recommend by Kuncheva and Whitaker,[43] due to its simplicity and easy interpretation. Minku and Yao[12] consider that high/low diversity refers to low/high average Q statistic.

The aim of dividing ensemble members into high/low diversity ensembles is the assumption that high and low diversities are related to ensemble accuracy. According to the authors, the accuracy of the ensembles may be similar (not the same) or very different as a consequence of severity and speed of each type of drift being and respectively. For instance, Minku and Yao[12] observed that high diversity ensembles achieve better accuracy rates when dealing with low severity and high speed drifts.

DDD operates in two modes: before and after drift detection. In the first mode, the low diversity ensemble and the high diversity ensemble are generated using incoming examples. Then, the after drift detection mode is triggered when there is no convergence about the concept, i.e., DDD monitors the low diversity ensemble using a drift detector, namely EDDM. In this last mode, the low/high diversity ensembles generated in the first mode are assigned as old low/high diversity ensembles and the first mode is reactivated in order to create new low/high diversity ensembles.

In general, DDD focus on learning the new concept using information learned from the old concept, i.e., by training the old high diversity ensemble on the new concept, leading to reduce its diversity. Minku and Yao[12] present experiments using artificial and real-world data. The attained results show that DDD usually achieves similar or even better accuracy than EDDM.

Another category of DDMs relies on unsupervised approaches. As described in the next section, clustering strategies and similarity measures are the main concerns of these methods.

Unsupervised Methods

Being different from the two categories mentioned before, the development of this category of methods intends to handle concept drift in a distribution of

unlabelled data. In such context, there are many data stream problems using clustering solutions.

The method proposed by Fanizzi et al.[44] focus on two problems: concept drift (known concepts changing) and novelty detection (changes to unknown concepts). An isolated cluster in the search space represents this last problem. In their method, for each cluster, the maximum distance between its instances and its medoid is computed to establish a decision boundary for each cluster. The union of the boundaries of all clusters is called global decision boundary. The new unknown incoming examples that fall outside this global decision boundary are assumed as no 'normal' data and need a further analysis. In this way, these examples are stored in a short-term memory for new clusters grouping, which might indicate concept drift or novelty detection.

Another work based on unsupervised drift detection is found in Otey and Parthasarathy.[45] In this work, the authors calculate the dissimilarity between two data windows ($\overline{X}$ and $\overline{Y}$) considering three components: distance, rotation, and variance. For the distance component, its dissimilarity D_{dist} is computed by means of Euclidean distance between the centroids of each dataset ($\mu_{\overline{X}}$ and $\mu_{\overline{Y}}$), according to the following expression

$$D_{\text{dist}}(\overline{X},\overline{Y}) = |\mu_{\overline{X}} - \mu_{\overline{Y}}| \tag{7}$$

For the rotation component, its dissimilarity D_{rot} is defined as the sum of the angles between the components (Eq. (8)). Since the columns of X and Y are the principal components of the datasets $\overline{X}$ and $\overline{Y}$, respectively. It follows that the diagonal of the matrix X^TY is the cosine of the angles between the corresponding principal components:

$$D_{rot}(\overline{X},\overline{Y}) = \text{trace}\left(cos^{-1}\left(abs\left(X^TY\right)\right)\right) \tag{8}$$

For the variance component, its dissimilarity D_{var} is defined by the symmetric relative entropy (SRE) between the distributions of the random variables $V_{\overline{X}}$ and $V_{\overline{Y}}$, as shown in Eq. (9):

$$D_{\text{var}}(\overline{X},\overline{Y}) = \text{SRE}\left(V_{\overline{X}}, V_{\overline{Y}}\right) \tag{9}$$

Finally, Otey and Parthasarathy[40] define the resultant dissimilarity D_{final} according to equation bellow:

$$D_{\text{final}}(\overline{X},\overline{Y}) = D_{\text{dist}} * D_{\text{rot}} * D_{\text{var}} \tag{10}$$

It is worth noting that, even though the method proposed by Otey and Parthasarathy[45] is applicable to detect drifts, learning process is not involved. In addition, this method deals with incoming data in chunks. The authors, however, suggest an alternative incremental form of anomaly and change detection. This incremental method may calculate $D_{\text{final}}(\overline{X},\overline{X}\cup\{x\})$, where x denotes the first sample following the window. In this way, it is possible to verify how much D_{final} may increase when the data point x is included. This measure may indicate a concept drift.

A Comparative Analysis of the Current Methods

All methods described in this section are summarized in Table 1. This table highlights how these methods are divided according to number of classifiers, incremental or nonincremental learning, and active or passive approaches. In addition, Table 1 also presents for which type of changes each method performs better, as well as whether or not the method needs labeled data to work with. Finally, for the active methods, the measure used for drift detection is mentioned too.

DDM, EDDM, and STEPD represent the same configuration of approaches. The main difference between these three drift detectors is the statistical test employed. These methods are based on single classifier, which is replaced after drift detection. Moreover, incoming data arrive in a stream, updating the current decision model incrementally (online learning). When warning level is reached, the samples update a kind of alternative decision model. However, alternative model only replaces the current decision model when drift level is reached. Since DDM, EDDM, and STEPD pass by the same sample just once, these methods are assumed to be online.[46]

We also consider DDM, EDDM, and STEPD as active methods, since the drift detection is explicit in their strategies. However, due to the fact that these methods are online, every incoming sample is added to the decision models (current or alternative), i.e., the system does not update only after drift detection. Actually, the system is updated as the incoming samples arrive.

These methods are robust to abrupt and gradual drifts. However, EDDM performs better when gradual drifts are very slow because it is based on distance between error occurrences. We have conducted simple experiments using these drift detectors over two classical artificial datasets: (1) SINE1, whose data show abrupt drifts and (2) CIRCLE, a gradual drift dataset. The following measures were

TABLE 1 | Compilation of Related Work Reported in Literature Grouped According to the Approach of Generic Solutions for Dynamic Environments Problems

Method	Classifiers	Learning	Data	Strategy	Detection Based on	Well Performed to
DDM[28]	Single	Online	Labeled	Active	Error monitoring	Abrupt/Gradual drift and noise
EDDM[29]	Single	Online	Labeled	Active	Error monitoring	Abrupt/Gradual (slow) drift
STEPD[30]	Single	Online	Labeled	Active	Error monitoring	Abrupt/Gradual drift and noise
SEA[37]	Ensemble	Batch	Labeled	Passive	—	Abrupt drift and noise
DWM[38]	Ensemble	Online	Labeled	Passive	—	Noise
ERDWM[39]	Ensemble	Online	Labeled	Passive	—	Recurring concepts and noise
AUE2[23]	Ensemble	Batch	Labeled	Passive	—	Abrupt/Gradual drift, recurring concept, and noise
DDD[12]	Ensemble	Incremental	Labeled	Active	It depends on detection method used	Abrupt/Gradual (slow) drift and noise
Fanizzi et al.[44]	Single	Batch	Unlabeled	Active	Dissimilarity	—
Otey & Parthasarathy[45]	—	—	Unlabeled	Active	Dissimilarity	Abrupt drift and outlier

DDM, Drift Detection Method; EDDM, Early Drift Detection Method; STEPD, Detection Method Using Statistical Testing; SEA, Streaming Ensemble Algorithm; DWM, Dynamic Weighted Majority; ERDWM, Early Dynamic Weighted Majority; AUE2, Accuracy Updated Ensemble; DDD, Diversity for Dealing with Drift.

TABLE 2 | Evaluation of the Most Common Drift Detectors

	SINE1				CIRCLE			
	Preq. Error	Delay	TD	FD	Preq. Error	Delay	TD	FD
DDM	0.0537	11	9	0	0.0487	32	3	0
EDDM	0.0540	14	9	0	0.0581	20	3	0
STEPD	0.0709	34	9	12	0.0932	0	0	0

DDM, Drift Detection Method; EDDM, Early Drift Detection Method; STEPD, Detection Method Using Statistical Testing; TD, true detection; FD, false detection.

evaluated: prequential error; detection delay (number of examples after drift occurrence and before detection); and number of true detection (TD) and false detection (FD). The results are presented in Table 2.

As can be observed in Table 2, DDM attained the lowest prequential error in both datasets, even when the detection delay was higher. The reason for this behavior is that DDM employs a statistical test, which selects better examples for the next concept. On the other hand, methods based on using a set of recent examples, such as STEPD, are less sensitive to gradual drifts.

The methods based on ensemble classifiers are better in maintaining previous knowledge than methods based on single classifier. These methods also use incremental learning divided into online learning, when the incoming data arrive as stream (DWM, ERDWM, and DDD), and one-pass learning, when incoming data arrive as batch (SEA and AUE2). Except for DDD, all the methods based on ensemble classifiers mentioned in this chapter use passive strategies to handle concept drifts. These methods are robust on reacting to new concepts. In addition, due to ensemble of classifiers, they are also robust on reacting to recurrent concepts. However, in period of stable concepts, they update the system unnecessarily.

In despite of the fact that DDD uses active strategy, this method needs another method to detect changes. However, as mentioned above, since the authors used EDDM only at the change detection phase, DDD is considered an incremental learning method, as shown in Table 1. Besides, DDD allows choosing a drift detector, which passes by each sample only once.

The remaining methods are based on unlabeled data. In Fanizzi et al.,[44] even though incoming data arrive on stream, first this method waits to form a cluster. Then, it integrates the created cluster to the model. Since it presents explicit drift detection, this method is assumed as an active strategy. The method proposed by Otey and Parthasarathy[45] is totally based on data distribution. Therefore, it does not use classifier (and learning). In addition, this method

TABLE 3 | Configuration for an Ideal Concept Drift Detection Method

Classifiers	Learning	Data	Strategy	Detection Not Based on
Ensemble	Online	Unlabeled	Active	Error monitoring

presents active strategy for data stream. It is better on detecting abrupt drifts because changes may occur from one window to another one. The incremental form of change detection suggested by the authors may handle gradual drifts.

Challenges and Open Issues

Passive strategies are practically infeasible in real applications, due to the following reasons. First, to be able to react to all changes, a system based on passive strategy must be updated in short time intervals, leading to high computational cost. Second, if the system is updated in large time intervals, some changes may not be noticed by the system.

These drawbacks allow us to believe that the best moment for system update is after change detection. In this way, the system will not spend an unnecessary computational cost, and all relevant changes will be noticed. Therefore, active strategies may be considered better than passive strategies, because they are based on explicit detection of changes.

In addition, as confirmed in this survey, ensemble of classifiers achieves better performance on handling many types of drifts, when compared to single classifiers. However, most of the ensemble-based techniques available in the literature are passive strategies. The exception is DDD. Nevertheless, as mentioned before, this method needs a drift detector.

In terms of active methods, the approaches discussed here work based on error monitoring or dissimilarity between incoming data. The error monitoring-based methods need an operator feedback to indicate when the error rate increases, i.e., it is necessary to know the true labels of the data. On the other hand, in several changing environment problems, such as spam filtering, the true labels are not always available. In this context, unsupervised methods take advantage, since drifts are detected on unlabeled data. Thus, error rate monitoring is not necessary.

Another important point is that, during the adaptation process, all systems described in this section follow a standard process: active strategies based on single classifiers replace the classifiers after drift detection using the most recent data to update their models, while ensemble-based, for both active and passive strategies, create new ensemble members using the most current data. What makes the difference in the adaptation process of ensemble-based methods is the identification of the right moment to replace old members, such as intended by Minku and Yao[12] in DDD method.

Finally, there are specific procedures for evaluating the performance of adaptive methods. Since streaming data evolve over time, one solution is to keep evaluating the model in different times (or incrementally) to see how the model improves.[47] Besides learning performance evaluation, there are some criteria for change detection evaluation listed by Gama et al.[47]: probabilities of true change detection and false alarms and delay of detection. This change detection evaluation should be computed on synthetic data where drifts are known.

These interesting observations help us to highlight that in practical problems data may arrive in a stream, with no fully labels available, the classification system needs to quick react to different types of drifts and previous concepts may reappear. Hence, to be able to deal with these issues, an ideal concept DDM would present the configuration shown in Table 3. The use of ensemble classifiers may provide better classification performances and maintain previous knowledge. In terms of online learning, the objective is to increase sensibility to drifts and the nonreutilization of the data. Since unlabeled data is a consequence of practical problems, a solution may be based on unsupervised or semi-supervised approaches. An active strategy is important to avoid high computational costs and unnecessary system updates. Finally, a method not based on error monitoring would avoid error increasing and would be well suited to unlabeled data.

Therefore, there are many open problems on employing classification systems to deal with concept drift. Based on this review, it is possible to observe the main problems detected on current solutions, the most successful directions for future contributions, as well as hypothesis for future work.

CONCLUSIONS

This work intended to review several solutions commonly applied to handle concept drifts in order to point out the main drawbacks and advantages of the current solutions. In addition, we focused on

clarifying the meaning of several terms usually used in the literature devoted to classification problems in dynamic environments, such as novelty detection, concept drift, and one-class classification.

Moreover, we emphasize that each change must be dealt with according to its characteristics: noises must be ignored; rare events must be considered as outliers; gradual, incremental, and abrupt changes must be detected to allow the system to be updated; and recurrent changes should not be forgotten. Thus, a highly accurate and reliable drift detector is expected to be robust to noises and rare events, and sensitive to the other changes.

Finally, this review intended to indicate some issues not addressed in the literature and future directions worthy of investigation, such as handling different types of drifts using no accuracy monitoring and no unnecessary updates.

FURTHER READING

Katakis I, Tsoumakas G, Vlahavas I. Tracking recurring contexts using ensemble classifiers: an application to email filtering. *Knowl Inf Syst* 2010, 22:371–391.

Marsland S. Novelty detection in learning systems. *Neural Comput Surv* 2003, 3:157–195.

REFERENCES

1. Kuncheva LI. Classifier ensembles for changing environments. In: *Multiple Classifier Systems*. Lecture Notes in Computer Science, vol. 3077. Berlin, Heidelberg: Springer; 2004, 1–15.
2. Kmieciak, M, Stefanowski, J. Handling sudden concept drift in enron messages data stream. Draft version of a paper finally published in T Morzy, M Gorawski, Wrembel R, A Zgrzywa (Eds.). *Technologie przetwarzania danych. Mat. III KNTPD Conference*, Poznan, WNT Press, 21–23 April, 2010, 284–296.
3. Wang, H Fan, W Yu, PS, Han, J. Mining concept-drifting data streams using ensemble classifiers. In: *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03)*, Washington DC, USA. August 24–27, 2003, 226–253.
4. Spinosa, EJ, de Carvalho, ACPLF, Gama, J. Cluster-based novel concept detection in data streams applied to intrusion detection in computer networks. In: *Proceedings of the ACM Symposium on Applied Computing (SAC'08)*, ACM, Fortaleza, Brazil. March, 2008, 976–980.
5. Lifna, CS, Vijayalakshmi, M. Identifying concept-drift in twitter streams. In: *Proceedings of ICACTA*, Mumbai, India, 2015. March 26–27, 2015, Vol. 45, 86–94.
6. Miljkovic, D. Review of novelty detection methods. In: *Proceedings of MIPRO*, Opatija, Croatia. May 24–28, 2010.
7. Markou M, Singh S. Novelty detection: a review. Part 1: statistical approaches. *Signal Process* 2003, 83:2481–2497.
8. Morsier, F, Borgeaud, M, Kuchler, C, Gass, V, Thiran J. Semi-supervised and unsupervised novelty detection using nested support vector machines. In: *Proceeding of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Munich, Germany. July 22–27, 2012.
9. Camps-Valls G, Bruzzone L. *Kernel Methods for Remote Sensing Data Analysis*. Chichester: Wiley Online Library; 2009.
10. Muñoz-Marí J, Bovolo F, Gómez-Chova L, Bruzzone L, Camp-Valls G. Semisupervised one-class support vector machines for classification of remote sensing data. *IEEE Trans Geosci Remote Sens* 2010, 48:3188–3197.
11. Faria ER, Gonçalves IJCR, de Carvalho ACPLF, Gama J. Novelty detection in data streams. *Artif Intell Rev* 2016, 45:235–269. doi:10.1007/s10462-015-9444-8.
12. Minku L, Yao X. DDD: a new ensemble approach for dealing with concept drift. *IEEE Trans Knowl Data Eng* 2012, 24:619–633.
13. Widmer, G. Combining robustness and flexibility in learning drifting concepts. In: *Proceedings of the 11th European Conference on Artificial Intelligence*, Wiley & Sons, Chichester, August 8–12, 1994, 468–472.
14. Hee Ang H, Gopalkrishnan V, Zliobaite I, Pechenizkiy M, Hoi SCH. Predictive handling of asynchronous concept drifts in distributed environments. *IEEE Trans Knowl Data Eng* 2013, 25:2343–2355. doi:10.1109/TKDE.2012.172.
15. Bose RPJC, van der Aalst WMP, Zliobaite I, Pechenizkiy M. Dealing with concept drifts in process mining. *IEEE Trans Neural Netw Learn Syst* 2014, 25:154–171.
16. Harries MB, Sammut C, Horn K. Extracting hidden context. *Mach Learn* 1998, 32:101–126.
17. Le T, Tran D, Nguyen P, Ma W, Sharma D. Multiple distribution data description learning method for novelty detection. In: *Proceedings of International Joint*

Conference on Neural Networks, San Jose, CA, July 31–August 5, 2011, 2321–2326.

18. Ada I, Berthold MR. EVE: a framework for event detection. In: *Evolving Systems*, vol. 4. Berlin Heidelberg: Springer; 2013, 61–70.
19. Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. *ACM Comput Surv* 2009, 41:1–72.
20. Kuncheva, LI. Classifier ensembles for detecting concept change in streaming data: overview and perspectives. In: *2nd Workshop SUEMA 2008*, *ECAI*, Patras, Greece, July, 2008, 5–10.
21. Widmer G, Kubat M. Learning in the presence of concept drift and hidden contexts. *Mach Learn* 1996, 23:69–101.
22. Gama J. *Knowledge Discovery from Data Streams*. 1st ed. Boca Raton, FL: CRC Press Chapman Hall; 2010.
23. Brzezinski D, Stefanowski J. Reacting to different types of concept drift: the accuracy updated ensemble algorithm. *IEEE Trans Neural Netw Learn Syst* 2014, 25:81–94.
24. Tsymbal A, Pechenizkiy M, Cunningham P, Puuronen S. Dynamic integration of classifiers for handdling concept drift. *Inf Fusion* 2008, 9:56–68.
25. Brzezinski, D. Mining data streams with concept drift. Master's thesis, *School: Poznan University of Technology*, Poznan, 2010.
26. Gomes JB, Gaber MM, Sousa PAC, Menasalvas E. Mining recurring concepts in a dynamic feature space. *IEEE Trans Neural Netw Learn Syst* 2014, 25:95–110.
27. Mitchell T. *Machine Learning*. New York: McGraw Hill; 1997.
28. Gama J, Castillo G. Learning with local drift detection. In: *Advances in Artificial Intelligence*. Lecture Notes in Computer Science, vol. 3171. Berlin/Heidelberg: Springer; 2004, 286–295. doi:10.1007/978-3-540-28645-5_29.
29. Baena-Garcia M, Del Campo-Ávila J, Fidalgo R, Bifet A. Early drift detection method. In: *ECML PKDD 2006 Workshop on Knowledge Discovery from Data Streams*. Berlin: Springer; 2006, 77–86.
30. Nishida K, Yamauchi K. Detecting concept drift using statistical testing. In: Corruble V, Takeda M, Suzuki E, eds. *Discovery Science*. Lecture Notes in Computer Science, vol. 4755. Berlin/ Heidelberg: Springer; 2007, 264–269.
31. Chen, H, Ma, S, Jiang, K. Detecting and adapting do drifting concepts. In: *Proceedings of 9th International Conference on Fuzzy Systems and Knowledge Discovery*, Chongqing, May 29–31, 2012, 775–779.
32. Altinçay H. Ensembling evidential k-nearest neighbor classifiers through multi-modal perturbation. *Appl Soft Comput* 2007, 7:1072–1083.
33. Ruta, D, Gabrys, B. Neural network ensembles for time series prediction. In: *Proceedings of the International Joint Conference on Neural Networks (IJCNN'2007)*, IEEE Press, Orlando, Florida, USA. August 12–17, 2007, 1204–1209.
34. Tremblay G, Sabourin R, Maupin P. Optimizing nearest neighbor in random subspaces using a multi-objective genetic algorithm. In: *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04)*, ICPR '04, IEEE Computer Society, Washington, DC, USA. August 23–26, 2004, Vol. 01, 208–211.
35. Valentini, G. Ensemble methods based on bias-variance analysis. PhD thesis, *Genova University*, 2003.
36. Zhang P, Zhu X, Shi Y. Categorizing and mining concept drifting data streams. In: *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, ACM, New York, NY, 2008, 812–820.
37. Street, WN, Kim, Y. A Streaming Ensemble Algorithm (SEA) for large-scale classification. In: *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA. August 26–29, 2001, 377–382.
38. Kolter JZ, Maloof MA. Dynamic weighted majority: an ensemble method for drifting concepts. *J Mach Learn Res* 2007, 8:2755–2790.
39. Sidhu P, Bhatia M, Bindal A. A novel online ensemble approach for concept drift in data streams. In: *Proceedings of IEEE Second International Conference on Image Information Processing (ICIIP)*, Shimla, 2013.
40. Oza, NC, Russell, S. Experimental comparisons of online and batch versions of bagging and boosting. In: *Proceedings of the ACM SIGKDD International Conference Knowledge Discovery and Data Mining*, San Francisco, CA, USA. August 26–29, 2001, 359–364.
41. Johansson, U, Lofstrom, T, Niklasson, L. The importance of diversity in neural network ensembles—an empirical investigation. In: *International Joint Conference on Neural Networks*, 2007, 661–666.
42. Yule G. On the association of attributes in statistics. *Philos Trans R Soc Lond A* 1900, 194:257–319.
43. Kuncheva LI, Whitaker CJ. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Mach Learn* 2003, 51:181–207.
44. Fanizzi N, Amato C, Esposito F. Conceptual clustering: concept formation, drift and novelty detection. In: *The Semantic Web: Research and Applications*. Lecture Notes in Computer Science, vol. 5021. Tenerife, Canary Islands, Spain, June 1–5, 2008, 318–332.
45. Otey, M, Parthasarathy, S. A dissimilarity measure for comparing subsets of data: application to multivariate time series. In: *Proceedings of the ICDM Workshop on Temporal Data Mining*, 2005.
46. Ditzler G, Polikar R. Incremental learning of concept drift from streaming imbalanced data. *IEEE Trans Knowl Data Eng* 2013, 25:10.
47. Gama J, Zliobaite I, Bifet A, Pechenizkiy M, Bouchachia A. A survey on concept drift adaptation. *J ACM Comput Surv* 2014, 46, Article No 44, 1–37.

