Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)
Survey Track

Recurrent Concept Drifts on Data Streams

Nuwan Gunasekara[1], Bernhard Pfahringer[1], Heitor Murilo Gomes[2], Albert Bifet[1,3], Yun Sing Koh[4]

[1]AI Institute, University of Waikato
[2]Victoria University of Wellington
[3]LTCI, Télécom Paris, IP Paris
[4]School of Computer Science, University of Auckland
ng98@students.waikato.ac.nz, {abifet,bernhard}@waikato.ac.nz, heitor.gomes@vuw.ac.nz, y.koh@auckland.ac.nz

Abstract

In an era where machine learning permeates every facet of human existence, and data evolves incessantly, the application of machine learning models transcends mere data processing. It involves navigating constant changes exemplified by the phenomenon of concept drift, which often affects model performance. These drifts can be recurrent due to the cyclic nature of the underlying data generation processes, which could be influenced by recurrent phenomena such as weather and time of the day. Stream Learning on data streams with recurrent concept drifts attempts to learn from such streams of data. The survey underscores the significance of the field and its practical applications, delving into nuanced definitions of machine learning for data streams afflicted by recurrent concept drifts. It explores diverse methodological approaches, elucidating their key design components. Additionally, it examines various evaluation techniques, benchmark datasets, and available software tailored for simulating and analysing data streams with recurrent concept drifts. Concluding, the survey offers insights into potential avenues for future research in the field.

1 Introduction

With the emergence of Industry, 4.0, more and more processes are monitored digitally, thus continuously generating tremendous quantities of data [Dreyfus *et al.*, 2022]. Data accessibility enables the implementation of impactful online data-driven machine learning models [Dreyfus *et al.*, 2022]. However, these data-driven models are impacted by concept drifts: input distribution shifts in the underlying data [Dreyfus *et al.*, 2022]. Concept drifts can differ due to impact, transition type, reach of change, recurrence, and blips/outliers/noise [Gunasekara *et al.*, 2023].

The literature describes real and virtual concept drifts considering *impact*. The former affects the decision boundary of the model. The latter does not influence the decision boundary. Hence, the model is unaffected [Gunasekara *et al.*, 2023]. Concerning *transition type*, the drifts are categorized into sudden (abrupt), gradual, and incremental drifts, considering the evolution of the relationship between features and the target and the speed of change. In sudden or abrupt drifts, the current data distribution changes to a new one within a short period [Gunasekara *et al.*, 2023]. For gradual drifts, this transition happens gradually [Suárez-Cetrulo *et al.*, 2023]. Here, one could observe instances from both distributions for a certain period. The transition time is very long for incremental drifts [Gunasekara *et al.*, 2023]. With *recurrent concept drifts*, a particular data distribution reoccurs in the stream after a certain period. Recurrent concept drifts could also have different transition types going from one distribution to another. Literature also categorizes concept drifts considering their *reach of change*: drifts that affect all of the features are regarded as global drifts, and drifts that affect some of the features are called local drifts [Suárez-Cetrulo *et al.*, 2023]. *Random blips/outliers/noise* are situations where, for a very short period, few instances do not belong to the current distribution popup in the stream [Suárez-Cetrulo *et al.*, 2023].

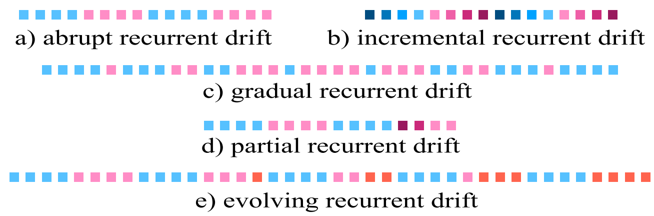

Figure 1: Recurrent concept drifts by concept transition.

This survey mainly focuses on recurrent concept drifts. Due to the cyclic nature of many physical phenomena, Stream Learning (SL) methods that can handle recurrent concept drifts were used in many real-world applications. A SL method that supports recurrent concept drifts was used to detect air quality using readings of neighboring sensors [Halstead *et al.*, 2022b]. Here, inference in air quality data was complex due to spatial non-linearities and abrupt temporal changes. Spatiotemporal relationships, environmental and contextual features such as meteorological conditions (wind), urban activity (traffic and heater use), and points of interest (locations of factories) were not available for the model. Thus, the model was required to adapt to these recurrent contexts dynamically. Furthermore, SL with recurrent concept drifts support were used in two energy optimization problems [Wu *et al.*, 2023]. The first, tries to optimize between energy storage and energy usage during tariff periods. The second one attempts to predict HVAC (heating, ventilation, and air conditioning) electricity, considering factors such as temperature and humidity. Factors such as seasonality and human activity induce recurrent concept drifts in both cases. In another real-world problem, per-concept models were used to predict the outlet temperature in a Vertical Roller Mill system with recurring concepts due to regulating strategies [Sun *et al.*, 2021]. Depending on the regulating strategy, previous concepts could reemerge. In another instance where species activity registered on a sensor that changed depending on the time of the day, custom models related to each species produced better results in determining the species than a generic model [Moreira dos Reis *et al.*, 2018]. Furthermore, recent research on data-driven model maintenance in industry 4.0 highlights the importance of SL on data streams with recurrent concept [Dreyfus *et al.*, 2022].

Considering the above and many real-world applications, this survey focuses on SL for data streams with recurrent concept drifts. The survey by [Suárez-Cetrulo *et al.*, 2023] mainly focuses on SL in general and explains different methods to handle recurrent concept drifts. However, it did not discuss evaluation methods, benchmark datasets, and software implementations for SL on data streams with recurrent concept drifts. A recent survey [Gunasekara *et al.*, 2023] looks at Online Continual Learning (OCL) from a Stream Learning perspective. Though it refers to [Suárez-Cetrulo *et al.*, 2023] for the latest SL work for recurrent concept drifts, the main emphasis there is how SL could improve Online Continual Learning. Hence, it misses a thorough investigation of SL on data streams with recurrent concept drifts. This survey attempts to fill the above-mentioned survey gaps by explaining design components, evaluation methods, benchmark datasets, and software implementations for SL on data streams with recurrent concept drifts. The survey also dwells into similar settings such as concept evolution with recurrent classes for SL and Online Continual Learning to find intersections and differences with those research fields.

This paper is structured as follows. First, we provide definitions of recurrent concept drifts. Considering their design components we explain popular SL methods for recurrent concept drifts in the following section. In the subsequent sections, we explore different evaluation methods, open-source software, benchmark datasets and future directions. Concluding remarks are presented in the final section.

2 Problem Statement

Consider an online learning algorithm A learning from an online (potentially infinite) non-Independent and Identically Distributed (IID) data stream. Here, the data stream can be defined as a stream of unknown distributions $D = D_1, D_2, \ldots, D_N$ over $X \times Y$, where X and Y are input and output random variables. The transition of $D_j \rightarrow D_{j+1}$ could happen such that $D_j \neq D_{j+1}$, to result in a *concept drift* [Gama *et al.*, 2014]. Specifically, it is a drift in the conditional probability distributions, where $P_j(y|X) \neq P_{j+1}(y|X)$ [Gama *et al.*, 2014; Gao *et al.*, 2020]. Furthermore, if $D_{j-k} = D_{j+1}$, this new distribution can be considered a recurrence of a k-th previous distribution. Such a drift is identified as a *recurrent concept drift* in literature [Suárez-Cetrulo *et al.*, 2023].

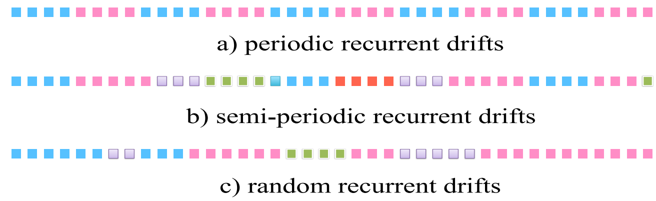

Figure 2: Recurrent concept drifts by time of recurrence.

Recurrent concept drifts could also have different transition types going from one distribution to another. Figure 1a illustrates *abrupt recurrent concept drifts*. Here, the distribution shifts from one concept to the other are abrupt. On the other hand, figure 1b explains *incremental recurrent concept drifts*, where instances slowly change over to the new concept from the previous concept, and this incremental transition of concepts is present throughout the recurrent drifts. With *gradual recurrent concept drifts*, concepts change gradually. During the drift period, instances from both concepts appear in the stream for some time. This is illustrated in figure 1c with instances from the blue concept appearing in the stream with the pink concept until it completely switches over to the pink concept. Then, again, instances slowly transition into the blue concept while instances from the pink concept gradually disappear from the stream. *Partial recurrent concept drifts* are when a given recurrent concept is only seen partially at the subsequent recurrence. For example, in figure 1d, the pink concept is partially present in the subsequent recurrence, while the blue concept is fully present in the next recurrence. Figure 1e explains an *evolving recurrent concept drift* where a given recurrent concept slowly evolves into a new concept over a period of time. Here, one can see the pink recurrent concept slowly evolving into the orange concept throughout the recurrences.

On the other hand, the recurrence of a given concept could have many forms, depending on the time of recurrence. As shown in figure 2a, recurrences follow a periodic pattern in *periodic recurrent drifts*. Seasonal weather changes are a good example of periodic recurrent drifts, where each season occurs around the same time of the year. In *semi-periodic recurrent drifts*, only certain concepts reoccur periodically. For example, in figure 2b, only the blue concept reoccurs regularly, while other concepts follow a random recurrence pattern. The recurrence pattern for each concept is hard to determine in a data stream with *random recurrent drifts*. This is illustrated with random recurrences for each concept in figure 2c.

Researchers working in the field should be aware of the different intricacies in recurrent concept drifts, considering drift transition type and concept recurrence pattern which is highlighted in figures 1 and 2.

3 Handling Recurrent Concept Drifts

The literature identifies many methods to detect drifts in the input distribution: *methods based on differences between two distributions*, *methods based on sequential analysis* and *methods based on statistical process control* [Gunasekara *et al.*, 2023]. Works by [Khamassi *et al.*, 2018] and [Gama *et al.*, 2014] contain thorough reviews of drift detectors for Stream Learning. Once a drift is detected, they discuss many methods to identify recurrent concepts. *Conceptual equivalence* assumes when two classifiers behave similarly in prediction for a time window, both classifiers describe the same concept [Yang *et al.*, 2006]. The idea of *concept similarity* was initially proposed in REDLLA [Li *et al.*, 2012] to detect recurring drifts in the absence of labeled data. Though the authors did not name this as such, the approach aimed to recognize similar concepts using Euclidean distances between concept clusters representing different concepts. Different distance measures for concept similarity are discussed in other research [Angel *et al.*, 2016; Gomes *et al.*, 2013]. RCD [Gonçalves Jr and De Barros, 2013] holds a pool of buffers and compares the current buffer against all to find similar concepts using a multivariate non-parametric statistical test.

Managing a *concept history* [Yang *et al.*, 2006; Alippi *et al.*, 2013], which contains a classifier for each concept, is popular among the research community. This idea of a *concept history* is described using different terminology like *pool of classifiers* [Hosseini *et al.*, 2013], *concept list* [Li *et al.*, 2012] and *concept repository* [Wu *et al.*, 2021; Wu *et al.*, 2022] in SL for data streams with recurrent concept drifts.

3.1 Explicit Handling of Recurrences

Some SL methods that deal with recurrent concept drifts build a classifier for each data batch and use a repository/ensemble management technique to maintain the repository/ensemble. These methods do not employ a drift detector to detect drifts. LEARN++.NSE trains a new classifier for each batch of data and combines the classifiers using weighted voting [Elwell and Polikar, 2011]. For recurring concepts, it assumes that the weights of learners relating to the recurrent concept will increase, and therefore, the final prediction will consider relevant knowledge. PMRCD also builds a classifier for each data batch. The method maintains a pool of ensembles for each concept [Hosseini *et al.*, 2012]. A maximum number of concepts and classifiers were assumed to handle the memory limitations. The authors proposed a mechanism to manage the classifier pool by merging and splitting concepts. A similar approach of maintaining a classifier pool and selecting the best ensemble was discussed in Dynse [Almeida *et al.*, 2018]. ASE also considered dynamically handling the ensemble size for recurrent concept drifts [Duda *et al.*, 2017]. Considering the current data chunk, it evaluates whether adding an item to the ensemble increases the accuracy for the entire data stream. The method uses Kullback-Leibler discrepancy to measure the suitability of the elements. GraphPool [Ahmadi and Kramer, 2018] maintains a pool of concepts by applying a merging mechanism whenever necessary: after receiving a new batch of data, it extracts a concept representation considering the correlation among features. Then, it compares the current representation to the representations in the pool using a statistical multivariate likelihood test. GraphPool merges all the corresponding concepts if they are similar. Like PMRCD, it also uses Naïve Bayes classifiers. Furthermore, GraphPool maintains the transition among concepts via a first-order Markov chain. This information is used in prediction. It can use either a single classifier or a weighted majority vote among classifiers for prediction.

3.2 Meta Learning

Independently of the number of base learners, some SL methods for recurrent concept drifts act as a wrapper algorithm to determine the best model or models for the current concept after detecting a drift. Thus, they act as meta-learners [Suárez-Cetrulo *et al.*, 2023]. RCD creates a new classifier for each concept and stores a sample of data used to build it [Gonçalves Jr and De Barros, 2013]. When it detects a new concept, the algorithm compares the incoming data to previous ones using a non-parametric multivariate statistical test to verify if both contexts come from the same distribution. If so, the corresponding classifier is reused. Concept Profiling Framework (CPF) [Anderson *et al.*, 2016] handles recurring concepts using a classifier pool. It evaluates which model to reuse when a concept drift is detected using conceptual equivalence with classification accuracy. New models are added into the pool in case of a drift and pruned considering their reuse frequency. CPF is shown to be very effective on synthetic data with clear recurrent drifts but fails to outperform RCD on real-world benchmarks [Anderson *et al.*, 2019]. One of the limitations of CPF is that it relies on a fixed buffer size to determine what model to reuse. This makes it slower to react to drift than the other approaches [Anderson *et al.*, 2019]. An enhanced version of CPF was proposed in ECPF [Anderson *et al.*, 2019]. It trains both a new and used classifier on new data and retains the more accurate classifier when concept drift occurs. ECPF creates a copy of the old classifier so that training on new data does not impact the old classifier. PEARL [Wu *et al.*, 2022] utilizes an exact technique and a probabilistic graphical model with Lossy Counting to replace drifted trees in an ensemble with relevant trees from a repository. The exact technique uses pattern matching to find the set of drifted trees that co-occurred in the past for predictions. At the same time, the probabilistic graphical model captures tree replacements among recurrent concept drifts and replaces the exact technique when stable.

3.3 Concept Clusters

Unsupervised concept representations and distance metrics for concept similarity were also used by the SL community working on data streams with recurrent concept drifts to identify recurrences [Li *et al.*, 2012; Wu *et al.*, 2012; Gomes *et al.*, 2012]. In REDLLA [Li *et al.*, 2012], when growing a tree, the k-Means clustering algorithm produces concept clusters and labels unlabeled data at the leaves. The deviation between current and historic concept clusters was used to identify concept drifts and recurring concepts. A similar approach was proposed in SUN [Wu *et al.*, 2012]. There instead of k-Means, authors used k-Modes as the clustering algorithm. A framework using the Context Spaces Model with Context Information was proposed in ContexTrac to represent different concepts [Gomes *et al.*, 2012]. However, they did not elaborate on a method to extract Context Information from a concept. A clustering-based semi-supervised framework, ESCR [Zheng *et al.*, 2021], uses Jensen Shannon divergence on classification confidence score [Haque *et al.*, 2016] to detect recurrent concept drifts. It detects recurring concept drift by looking for any significant change in classifier confidence scores. Then, it determines the possibility of recurring concept drift via Jensen-Shannon divergence by calculating the distance between two confidence score distributions. ESCR performed better than REDLLA and RCD on some synthetic datasets. However, its performance was poor compared to other baselines when the dataset contains irrelevant attributes. CDMSE [Li *et al.*, 2021] works with missing labeled data. There, the predicted class labels by an ensemble model were partitioned into clusters for each data chunk to infer their class labels. Then a concept drift detection method based on the divergence of distributions between adjoining data chunks was used to distinguish recurring concept drifts. The method performed slightly better than SUN and REDLLA on different percentages of unlabeled data. CCP [Katakis *et al.*, 2010] is a very early method that used data stream clustering for SL on data streams with recurrent concept drifts. It proposes a general framework for classifying data streams by exploiting stream clustering to build and update an ensemble of incremental classifiers dynamically. Data stream clustering framework UClust [Namitha and Santhosh Kumar, 2020] was proposed to handle unlabeled data streams with recurrent concepts. Clusters detected through CluStream [Aggarwal *et al.*, 2003] were used to detect drifts and identify concept recurrences. In the CDCMS framework, clustering in the model space was used to build a diverse ensemble and identify recurring concepts [Chiu and Minku, 2020]. The authors argue that diversity accelerates adaptation to different types of drifts when the new concept is similar to the past concepts.

3.4 Drift Prediction

Some methods attempt to predict the next drift or the concept, considering the recurrent nature of the concept's appearance in the stream. These methods try to either proactively influence the drift detection mechanism or re-actively correct the detection signal by the drift detector. MM-PRec [Angel *et al.*, 2016] trains a meta-learner that uses a Hidden Markov Model to predict when a drift will happen and the most suitable concept for each situation if it is recurrent. To measure concept similarity, the authors used a function based on fuzzy logic. The extra computing required to train the meta-model was identified as the method's main drawback. Predictive Change Confidence Function (PCCF) for modeling recurrent changes and predicting change points was derived using the average time between changes and its standard deviation [Maslov *et al.*, 2016].The PCCF models recurrent streams as convolutions of Gaussian distributions of the time intervals between changes. The method can be used to post-process a detection by a drift detector or dynamically adjust the sensitivity of a drift detector. Later, BLPA [Maslov *et al.*, 2017] used PCCF to improve Bayesian Online Change Point Detector (BOCPD) [Adams and MacKay, 2007] for recurrent concept drifts. BOCPD was also used to develop Change Point Recurrence Distribution (CPRD) as an empirical estimate of the recurrent behavior of observed change points [Reich *et al.*, 2019a; Reich *et al.*, 2019b]. ProSeed [Chen *et al.*, 2016] uses a probabilistic network that uses stream volatility patterns to predict future changes. Like PCCF, this method also works independently of the drift detection technique. ProSeed was incorporated into the drift detector SEED [Huang *et al.*, 2014] to yield a proactive drift detector. Experimental results showed that ProSeed performed better than reactive drift detectors for data streams with reoccurring volatility trends. The same approach was used in ProChange [Koh *et al.*, 2018] to improve a drift detector using Hellinger distance to detect virtual drifts and Hoeffding inequality to detect real drifts for unlabeled transactional data. Metadata Drift Predictor (MDP) proposes a dynamically adapting drift detector using drift-related metadata clustering [Anderson *et al.*, 2018]. MDP allows the drift detector to be more sensitive when metadata is similar to past drifts and more conservative when metadata is dissimilar. In their empirical evaluations, MDP performed more accurately compared to ProSeed. Nacre proposes a framework that contains a recurrent drift classifier, a sequence predictor, and a drift coordinator for smooth adaptation of recurrent concept drifts [Wu *et al.*, 2021]. The recurrent drift classifier maintains a concept repository for previously learned concepts. The drift sequence predictor predicts the next drift point based on the previous drift intervals. The drift coordinator manipulates the recurrent drift classifier and the drift

Sec	Method	Year	DD	DP	LM	LN/A	MetaL	MetaF	Clust	CEqSim	Ens	CPool
3.1	LEARN++*	2011									X	X
	PMRCD	2012									X	X
	Dynse	2018									X	X
	ASE	2017								X	X	
	GraphPool	2018								X	X	X
3.2	RCD	2013	X				X			X	X	X
	CPF	2016	X				X			X		X
	ECPF	2019	X				X			X		X
	PEARL	2022	X				X				X	X
3.3	REDLLA	2012	X		X				X	X		X
	SUN	2012	X		X				X	X		X
	ContexTrac	2012	X						X	X		X
	ESCR	2021	X		X				X	X	X	
	CDMSE	2021	X		X				X		X	
	CCP	2010						X	X		X	
	UClust	2020	X		X	X		X	X	X		X
	CDCMS	2020	X						X		X	
3.4	MM-PRec	2016	X	X			X			X		X
	PCCF	2016		X								
	BLPA	2017	X	X								
	CPRD	2019	X	X								
	ProSeed	2016	X	X								
	ProChange	2018	X	X	X							
	MDP	2018	X	X				X	X			
	Nacre	2021	X	X							X	
3.5	SELeCT	2022						X				X
	FiCSUM	2023	X					X				

Table 1: Design components of the proposed methods for recurrent concept drifts in each section (Sec). DD: Drift Detection, DP: Drift Prediction, LM: Labels Missing, LN/A: Labels Not Available, MetaL: Meta Learning, MetaF: Meta Features, Clust: Clustering CEqSim: Conceptual Equivalence/Concept Similarity, Ens: Ensemble, CPool: Concept Pool. LEARN++*: LEARN++.NSE

sequence predictor to adapt to drifts proactively. Nacre uses PEARL [Wu *et al.*, 2022] as the recurrent drift classifier.

3.5 Meta Features

Apart from performance statistics such as accuracy, error rate, kappa, area under the curve, or classifier confidence score, some methods use different meta-features to improve SL on data streams with recurrent concept drifts. CCP stores the running mean and standard deviation for each numeric attribute and the fraction of instances for each bin in the case of nominal attributes over a sliding window. These features are identified as 'Conceptual Vectors' in CCP. GraphPool takes this idea further by storing a covariance matrix of attributes for each class. Cluster feature vector [Zhang *et al.*, 1996] was used to store information about each cluster in UClust. Authors of FiCSUM [Halstead *et al.*, 2023a] argue that no single meta-feature can fully represent a concept. They propose a general framework for combining various meta-features into a single representation. They propose a method for efficiently computing, storing, and querying an arbitrary meta-feature set as a single representation. They also propose a method for dynamic learning of meta-features that distinguishes concepts for a given dataset. According to the authors, FiCSUM enables feature selection methods, such as mutual information, to be applied to concept representation meta-features [Halstead *et al.*, 2023a]. The idea was initially proposed in 2021 however, in the initial version, it was identified that introducing irrelevant meta-features may reduce performance. A new data structures which facilitates feature selection to learn the relevant meta-features was introduced in the later version[Halstead *et al.*, 2023a]. SELeCT [Halstead *et al.*, 2022a] maintains a distinct internal state for each classifier. It uses meta-features described in FiCSUM to describe a state. Considering the previous states, it selects the best classifier for the current state of the stream. SELeCT contains three components: a method for representing a concept as a system state, a method for computing state priors and likelihoods, and a continuous state selection statistical test to select the active state for an incoming observation.

Table 1 summarises how the above-mentioned-design components were used in different SL methods for data streams with recurrent concept drifts. It further shows that some solutions employ many techniques to address the issue, whereas others use only a few. We can also observe that techniques involving clustering and concept meta-features have gained traction in recent years.

4 Evaluation

It is crucial to correctly evaluate a newly proposed method or a drift detector to understand its strengths and weaknesses. Except for typical prequential evaluation and data stream cross-validation along with accuracy, kappa, and other metrics [Gunasekara *et al.*, 2023], novel techniques proposed for SL on data streams with recurrent concept drifts require evaluation methods that demonstrate their suitability for such data streams. The literature explains a few evaluation methods for SL on recurrent concept drifts. They mainly fall into three categories: i) evaluation methods that evaluate the general performance of a method on a data stream with recurrent concept drifts, ii) evaluation methods that evaluate the model selection for each concept on a data stream with recurrent concept drifts, iii) evaluation methods which evaluate drift detection methods for data streams with recurrent concept drifts.

4.1 Evaluating Relative Performance

Other than reporting typical prequential evaluation of a newly proposed classifier A on a data stream with recurrent concept drifts, this evaluation compares the performance of classifier A against a baseline classifier B. Early work proposed a variant of Q statistic [Gama *et al.*, 2013]: $Q_i = log(B_{err_i}/A_{err_i})$, where the index i refers to the time-stamp, and A_{err} and B_{err} refers to the error rate of the classifiers under comparison [Gama and Kosina, 2014]. In more recent work, Cumulative Accuracy Gain (CAG) [Wu *et al.*, 2022]: $\sum((accuracy(A) - accuracy(B))$ was proposed to evaluate a new method for recurrent concept drifts. CAG represents a sum of accuracy differences against a baseline at a given sample frequency. Apart from accuracy, CAG evaluation could also be applied using kappa statistics. In both the above evaluation methods, the base classifier B selection greatly influences the final evaluation.

4.2 Evaluating Model Selection for Each Concept

Most of the SL methods proposed for data streams with recurrent concept drifts keep a pool of classifiers/buffers for each concept and attempt to retrieve the most suitable one from the pool for a given concept. To evaluate this, three measurements of context identification are discussed in the literature [Halstead *et al.*, 2021]. Proposed measurements consider concepts that are previously known and their context and model are use as nominal and discrete for a given dataset. Here, context refers to an underlying condition that results in a concept. For example, a stream may be in a given context A for a period of observations and then swap to context B over the subsequent period. At the same time, a system may use model A it has created over a period of time and then swap to model B.

$F1_c$ measures the context linkage considering model use in these cases. It measures the strength of the relationship between each $<$ model, context $>$ pair, e.g. $<$ Model A, rain $>$, using recall (the fraction of rain samples which occurred under Model A) and precision (the fraction of samples under Model A where it was also raining). For given s and c, where s is the time steps a given model was active and c is the time steps a given underlying context was present. The authors calculated the recall(R) and precision(P) of s on c: $R(s,c) = \frac{|\{t|t \in s \text{ and } t \in c\}|}{|c|}$, $P(s,c) = \frac{|\{t|t \in s \text{ and } t \in c\}|}{|s|}$ and further calculated the F1 score: $F1(s,c) = 2\frac{R(s,c)P(s,c)}{R(s,c)+P(s,c)}$.

The above is used to calculate three transparency measures. **i) Average Context $\boldsymbol{F1_c}$**: is the maximal $F1$ score obtained considering the cooccurrence of a single model and each context, averaged across all underlying contexts: $F1_c = \sum_{c \in C} \frac{1}{|C|} \max_s F1(s,c)$. For any given underlying context, this measures the approximate average $F1$ score a single model hopes to achieve. **ii) Average System $\boldsymbol{F1_s}$**: is the maximal $F1$ score obtained by considering the cooccurrence of each model and a single context, averaged across all models, weighted by length of active time: $F1_s = \sum_{s \in S} \frac{|s|}{|S|} \max_c F1(s,c)$. This measures the average F1 score a model chosen randomly has on its closest context match, given only the system's model history. **iii)** $\boldsymbol{s \approx c}$: the number of models which achieve a recall and precision above 80% on one underlying context. These models have the potential to predict underlying contexts.

According to the authors, $F1_c$ and $F1_s$ have similar properties to the standard F1 measure, which ranges from 0 to 1. The $s \approx c$ measure ranges from 0 to the maximum number of underlying concepts present in the stream. A high $F1_c$ indicates that changes in concept were successfully identified. In FiCSUM, $F1_c$ is identified as C-F1 [Halstead *et al.*, 2023a].

4.3 Evaluating Drift Detection on Synthetic Data

MDP explains an evaluation framework for drift detection on synthetic datasets [Anderson *et al.*, 2018]. As one can specify when the true drifts occur, they can be compared to when the drifts are detected. False positive rate FP is the proportion of drifts detected when no drift occurred since the last drift detection. Here, the true positive rate (TP) = 1 - FP. The false negative rate FN is defined as the proportion of true drifts followed by another drift before a drift detection. These measures are compared to highlight the tradeoff between Type I (FP) and Type II (FN) errors in each method. The authors also considered the drift detection delay, the mean number of instances between an actual drift, and the subsequent drift detection. Similar evaluations on drift detection and drift detection delay are also presented in other work as well [Maslov *et al.*, 2016; Huang *et al.*, 2014; Koh *et al.*, 2018].

5 Open Source Software

Most of the methods discussed in section 3 have a GitHub fork of a popular SL framework like Massive Online Analysis (MOA), scikit-multiflow, or River with the implementation. LEARN++.NSE is available as a learner in MOA. PCCF and ProChange are available as separate Python programs. MOA implementations of CPF, ECPF, MDP, and PEARL are available as separate repositories. scikit-ika, which is based on scikit-multiflow, contains implementations of PEARL and Nacre. FALL [Halstead *et al.*, 2023b] is a modular adaptive learning platform based on River. Thus it allows one to use different classifiers and drift detectors from River. It contains modules to calculate some of the meta-features discussed in the section 3.5 such as the ones used in FiCSUM. FALL also implements the selection mechanism and C-F1 context evaluation explained in section 4.2.

6 Benchmark Datasets

Some of the commonly used synthetic data generators such as SEA, Hyperplane, Agrawal, Random Tree, LED and different concept drift simulators are available in MOA. Although real-world datasets Electricity and Sensor [Angel *et al.*, 2016] are widely used in SL literature for data streams with recurrent concept drifts, their exact concept recurrences are unclear.

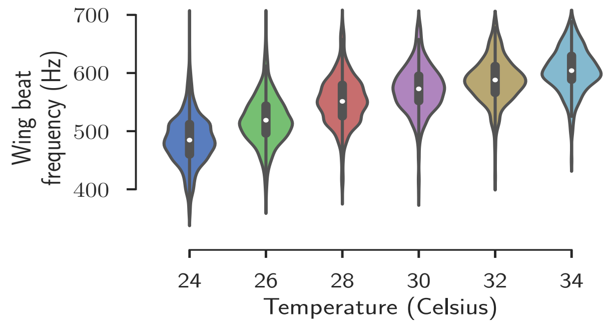

Figure 3: Different temperature contexts which influence the wing-beat frequency of Aedes aegypti mosquitoes in Aedes-Culex dataset. Source: [Moreira dos Reis *et al.*, 2018].

On the other hand, real-world datasets Aedes-Culex, Aedes-Sex, Arabic-Digit, CMC, Handwritten-QG, and Wine contain clear recurring contexts [Moreira dos Reis *et al.*, 2018]. For example, figure 3 shows the different temperature contexts that influence the wing-beat frequency of Aedes aegypti mosquitoes in the Aedes-Culex dataset. FiCSUM uses these contexts to identify resulting concepts. The building electricity demand simulation dataset used in [Wu *et al.*, 2023] also contains recurrent concepts.

7 Future Directions

Considering the recent developments in SL and machine learning in general, this section aims to explore possible intersections of those fields with SL on data streams with recurrent concept drifts.

7.1 Unlabeled Data and Concept Evolution

Apart from one recent work [Namitha and Santhosh Kumar, 2020], there has not been much attention paid to unlabeled data streams with recurrent concepts among the research community. Considering that most real-world data is unlabeled, this could be an exciting research direction the SL community working on recurrent concept drifts could peruse.

Section 2 definition of a recurrent concept drift at concept $j+1$ only considers the recurrent concepts where $P_j(y|X) \neq P_{j+1}(y|X)$. On the other hand, *concept evolution* only considers evolving target variables. Here at $j+1$, new set of classes emerge compared to j: $P_j(y) \neq P_{j+1}(y)$ [Gao *et al.*, 2020]. Here, the distribution shift is in the label data instead of the input data. Apart from completely new (previously not seen) classes at $j+1$, [Masud *et al.*, 2011; Gao *et al.*, 2020] considers concept evolution with the reappearance of any previously seen class that was not available at j. There are only a few works in this area of concept evolution with recurrent classes. Most of the techniques discussed in section 3 could be applied in this setting. Thus, we see this as a natural expansion of current SL on data streams with recurrent concept drifts.

7.2 Online Continual Learning

Online Continual Learning also considers online learning of tasks with different label/input distributions [Gunasekara *et*

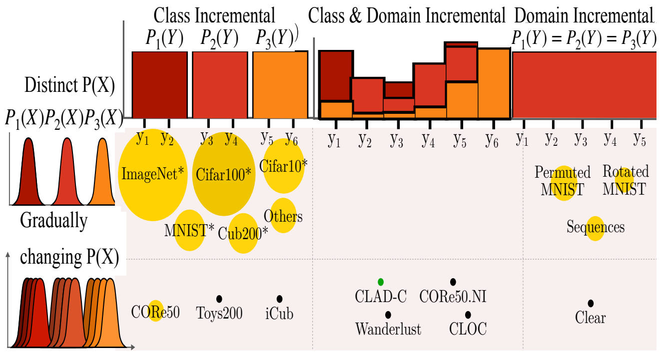

Figure 4: Continual Learning Datasets [Verwimp *et al.*, 2023].

al., 2023]. But in OCL, task reoccurrence is not guaranteed. Here, the learning system is forced to remember old tasks. Hence, after training on a given task, instances from all the previous tasks are considered in the evaluation [Gunasekara *et al.*, 2023]. Thus, forgetting of previous tasks after learning a new task can be calculated.

OCL considers two settings with different types of distribution shifts: Online Domain Incremental Continual Learning (ODICL) and Online Class Incremental Continual Learning (OCICL). ODICL is similar to SL on data streams with concept drifts where distribution shifts are in the input data [Gunasekara *et al.*, 2023]. On the other hand, OCICL focuses online learning on tasks with different label distributions. It is similar to SL with concept evolution. Considering the similarity in two settings: SL and OCL [Gunasekara *et al.*, 2023] explains how some of the ideas from SL on recurrent concept drifts could be useful in OCL. Specifically, drift detection and prediction to detect and predict tasks, model repository management techniques to manage a pool of Neural Network (NN)s and model selection techniques to select a Neural Network for prediction on instances from a past task.

On the other hand, some of the real-world OCL datasets such as: CORe50 and Online Domain Incremental Continual Learning version of CORe50 [Gunasekara *et al.*, 2022] and many of the datasets discussed in [Verwimp *et al.*, 2023] can be used to evaluate SL models for data streams with recurrent concept drifts. Figure 4 contains a summary of different real-world OCL datasets. As per figure 4, some datasets contain clear distribution shifts going from one task to the other, and in most of those datasets, the start and end of the task are clearly defined to support the OCL evaluation. Such real-world datasets can be very useful in evaluating SL methods proposed for data streams with recurrent concept drifts.

7.3 Evaluation for Recurrent Concept Drifts

The main limitation of the performance evaluation methods discussed in section 4.1 for SL on recurrent concept drifts is that they require a baseline method. Thus, the performance of the proposed algorithm is dependent upon the performance of the baseline method. Here, one could choose a general SL algorithm as the baseline to evaluate an algorithm specifically proposed for recurrent concept drifts. This does not allow us to evaluate how accurately the proposed algorithm performs on recurrent concept drifts. On the other hand, model selection evaluation methods discussed in section 4.2 only consider selecting the correct model for the relevant concept. They do not give us an indication of the model's predictive performance.

Novel evaluation methods that specifically address a given algorithm's suitability when learning from a data stream with recurrent concept drifts could be an area that the research community could further explore. When evaluating the performance of a new algorithm, it is best to avoid a baseline algorithm, as evaluation is a bit subjective to the performance of the selected baseline. Possible inspirations could come from Online Continual Learning evaluations [Gunasekara *et al.*, 2023]. Online Domain Incremental Continual Learning specifically is quite similar to Stream Learning [Gunasekara *et al.*, 2023]. Different tasks with distribution shifts in the input data appear in the stream. The evaluation method is aware of the drift points of the stream. After learning a new task, the OCL method is evaluated against separate test sets of all the already learned tasks. Apart from accuracy, one interesting metric discussed in OCL is forgetting [Gunasekara *et al.*, 2023]. Here, the current task's performance of the OCL method is compared against previous tasks' best performance, with lower forgetting indicating a model improvement over past tasks.

For data streams with recurrent concept drifts, if the start and end points of a concept and its relevant recurrent concept are known, then one can track the performance of recurrent concept explicitly. This process can provide useful insights for each recurrent concept. It can help identify which recurrent concept performs best with which proposed algorithm. Similar to current stream learning evaluation methods, this new evaluation can be applied with accuracy, kappa, and other evaluation metrics discussed in SL literature [Gunasekara *et al.*, 2023]. One could also use a similar metric as forgetting to evaluate the model improvement over different reincarnations of the same concept.

8 Conclusions

This survey focuses on Stream Learning methods for data streams with recurrent concept drifts. It highlights the importance of SL on data streams with recurrent concept drifts considering concept recurrence in the data streams generated by the increasing digitization processes. The survey explains the currently available methods, their design components, evaluation methods, benchmarks, and the availability of software implementations. It highlights the need for evaluation methods that do not use a baseline algorithm. Furthermore, it inquires into Online Continual Learning and Concept Evolution with recurrent classes to find possible intersections and directions for future research.

Though we see the importance of explainability in recurrent concept drifts, where one explains the reasons for recurrences, explainable Stream Learning itself is still in its infancy. We hope future work in explainable Stream Learning would lay the foundations for explaining recurrent concept drifts.

References

[Adams and MacKay, 2007] Ryan Prescott Adams and David JC MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

[Aggarwal *et al.*, 2003] Charu C Aggarwal, S Yu Philip, Jiawei Han, and Jianyong Wang. A framework for clustering evolving data streams. In *VLDB*. Elsevier, 2003.

[Ahmadi and Kramer, 2018] Zahra Ahmadi and Stefan Kramer. Modeling recurring concepts in data streams: a graph-based framework. *Knowl Inform Syst*, 2018.

[Alippi *et al.*, 2013] Cesare Alippi, Giacomo Boracchi, and Manuel Roveri. Just-in-time classifiers for recurrent concepts. *IEEE Trans. Neural Netw. Learn. Syst.*, 2013.

[Almeida *et al.*, 2018] Paulo RL Almeida, Luiz S Oliveira, Alceu S Britto Jr, and Robert Sabourin. Adapting dynamic classifier selection for concept drift. *Expert Syst Appl*, pages 67–85, 2018.

[Anderson *et al.*, 2016] Robert Anderson, Yun Sing Koh, and Gillian Dobbie. Cpf: Concept profiling framework for recurring drifts in data streams. In *AJCAI*, pages 203–214. Springer, 2016.

[Anderson *et al.*, 2018] Robert Anderson, Yun Sing Koh, and Gillian Dobbie. Predicting concept drift in data streams using metadata clustering. In *IJCNN*, pages 1–8. IEEE, 2018.

[Anderson *et al.*, 2019] Robert Anderson, Yun Sing Koh, Gillian Dobbie, and Albert Bifet. Recurring concept meta-learning for evolving data streams. *Expert Syst Appl*, page 112832, 2019.

[Angel *et al.*, 2016] Abad Miguel Angel, Gomes Joao Bartolo, and Menasalvas Ernestina. Predicting recurring concepts on data-streams by means of a meta-model and a fuzzy similarity function. *Expert Syst Appl*, 2016.

[Chen *et al.*, 2016] Kylie Chen, Yun Sing Koh, and Patricia Riddle. Proactive drift detection: Predicting concept drifts in data streams using probabilistic networks. In *2016 IJCNN*, pages 780–787. IEEE, 2016.

[Chiu and Minku, 2020] Chun Wai Chiu and Leandro L Minku. A diversity framework for dealing with multiple types of concept drift based on clustering in the model space. *IEEE Trans. Neural Netw. Learn. Syst.*, pages 1299–1309, 2020.

[Dreyfus *et al.*, 2022] Paul-Arthur Dreyfus, Antoine Pélissier, Foivos Psarommatis, and Dimitris Kiritsis. Data-based model maintenance in the era of industry 4.0: A methodology. *Journal of Manufacturing Systems*, pages 304–316, 2022.

[Duda *et al.*, 2017] Piotr Duda, Maciej Jaworski, and Leszek Rutkowski. On ensemble components selection in data streams scenario with reoccurring concept-drift. In *SSCI*, pages 1–7. IEEE, 2017.

[Elwell and Polikar, 2011] Ryan Elwell and Robi Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, pages 1517–1531, 2011.

[Gama and Kosina, 2014] Joao Gama and Petr Kosina. Recurrent concepts in data streams classification. *Knowl Inform Syst*, pages 489–507, 2014.

[Gama *et al.*, 2013] Joao Gama, Raquel Sebastiao, and Pedro Pereira Rodrigues. On evaluating stream learning algorithms. *Machine learning*, pages 317–346, 2013.

[Gama *et al.*, 2014] João Gama, Indrė Žliobaitė, Albert Bifet, M Pechenizkiy, and A Bouchachia. A survey on concept drift adaptation. *CSUR*, pages 1–37, 2014.

[Gao *et al.*, 2020] Yang Gao, Swarup Chandra, Yifan Li, Latifur Khan, and Thuraisingham Bhavani. Saccos: A semi-supervised framework for emerging class detection and concept drift adaption over data streams. *IEEE Trans Knowl Data Eng*, 34(3):1416–1426, 2020.

[Gomes *et al.*, 2012] João Bártolo Gomes, Pedro AC Sousa, and Ernestina Menasalvas. Tracking recurrent concepts using context. *Intell Data Anal*, pages 803–825, 2012.

[Gomes *et al.*, 2013] Joao Bartolo Gomes, Mohamed Medhat Gaber, Pedro AC Sousa, and Ernestina Menasalvas. Mining recurring concepts in a dynamic feature space. *IEEE Trans. Neural Netw. Learn. Syst.*, 2013.

[Gonçalves Jr and De Barros, 2013] Paulo Mauricio Gonçalves Jr and Roberto Souto Maior De Barros. Rcd: A recurring concept drift framework. *Pattern Recognition Letters*, pages 1018–1025, 2013.

[Gunasekara *et al.*, 2022] Nuwan Gunasekara, Heitor Gomes, Albert Bifet, and Bernhard Pfahringer. Adaptive neural networks for online domain incremental continual learning. In *DS 2022*, pages 89–103. Springer, 2022.

[Gunasekara *et al.*, 2023] Nuwan Gunasekara, Bernhard Pfahringer, Heitor Murilo Gomes, and Albert Bifet. Survey on online streaming continual learning. In *IJCAI*, pages 6628–6637, 2023.

[Halstead *et al.*, 2021] Ben Halstead, Yun Sing Koh, P Riddle, R Pears, M Pechenizkiy, and Albert Bifet. Recurring concept memory management in data streams: exploiting data stream concept evolution to improve performance and transparency. *DM and KD*, pages 796–836, 2021.

[Halstead *et al.*, 2022a] Ben Halstead, Yun Sing Koh, Patricia R, M Pechenizkiy, and Albert Bifet. A probabilistic framework for adapting to changing and recurring concepts in data streams. In *DSAA*, pages 1–10. IEEE, 2022.

[Halstead *et al.*, 2022b] Ben Halstead, Yun Sing Koh, Patricia Riddle, Russel Pears, Mykola Pechenizkiy, Albert Bifet, Gustavo Olivares, and Guy Coulson. Analyzing and repairing concept drift adaptation in data stream classification. *Machine Learning*, pages 3489–3523, 2022.

[Halstead *et al.*, 2023a] Ben Halstead, Yun Sing Koh, Patricia Riddle, Mykola Pechenizkiy, and Albert Bifet. Combining diverse meta-features to accurately identify recurring concept drift in data streams. *ACM Trans Knowl Discov Data*, pages 1–36, 2023.

[Halstead *et al.*, 2023b] Ben Halstead, Yun Sing Koh, Patricia Riddle, Mykola Pechenizkiy, and Albert Bifet. Fall: A modular adaptive learning platform for streaming data. In *ICDE*, pages 3619–3622. IEEE, 2023.

[Haque *et al.*, 2016] Ahsanul Haque, Latifur Khan, and Michael Baron. Sand: Semi-supervised adaptive novel class detection and classification over data stream. In *AAAI*, 2016.

[Hosseini *et al.*, 2012] Mohammad Javad Hosseini, Zahra Ahmadi, and Hamid Beigy. New management operations on classifiers pool to track recurring concepts. In *DaWaK*, pages 327–339. Springer, 2012.

[Hosseini *et al.*, 2013] Mohammad Javad Hosseini, Zahra Ahmadi, and Hamid Beigy. Using a classifier pool in accuracy based tracking of recurring concepts in data stream classification. *Evolving Systems*, pages 43–60, 2013.

[Huang *et al.*, 2014] David Tse Jung Huang, Yun Sing Koh, Gillian Dobbie, and Russel Pears. Detecting volatility shift in data streams. In *ICDM*, pages 863–868. IEEE, 2014.

[Katakis *et al.*, 2010] Ioannis Katakis, Grigorios Tsoumakas, and Ioannis Vlahavas. Tracking recurring contexts using ensemble classifiers: an application to email filtering. *Knowl Inform Syst*, pages 371–391, 2010.

[Khamassi *et al.*, 2018] Imen Khamassi, Moamar Sayed-Mouchaweh, Moez Hammami, and Khaled Ghédira. Discussion and review on evolving data streams and concept drift adapting. *Evolving systems*, pages 1–23, 2018.

[Koh *et al.*, 2018] Yun Sing Koh, David Tse Jung Huang, Chris Pearce, and Gillian Dobbie. Volatility drift prediction for transactional data streams. In *ICDM*, pages 1091–1096. IEEE, 2018.

[Li *et al.*, 2012] Peipei Li, Xindong Wu, and Xuegang Hu. Mining recurring concept drifts with limited labeled streaming data. *TIST*, pages 1–32, 2012.

[Li *et al.*, 2021] Peipei Li, Man Wu, Junhong He, and Xuegang Hu. Recurring drift detection and model selection-based ensemble classification for data streams with unlabeled data. *New Generat Comput*, pages 341–376, 2021.

[Maslov *et al.*, 2016] Alexandr Maslov, Mykola Pechenizkiy, Indrė Žliobaitė, and Tommi Kärkkäinen. Modelling recurrent events for improving online change detection. In *SDM*, pages 549–557. SIAM, 2016.

[Maslov *et al.*, 2017] Alexandr Maslov, Mykola Pechenizkiy, Yulong Pei, Indre Žliobaitė, Alexander S, Tommi K, and Jaakko H. Blpa: Bayesian learn-predict-adjust method for online detection of recurrent changepoints. In *IJCNN*, pages 1916–1923. IEEE, 2017.

[Masud *et al.*, 2011] Mohammad M Masud, Tahseen M Al-Khateeb, Latifur Khan, Charu Aggarwal, Jing Gao, Jiawei Han, and Bhavani Thuraisingham. Detecting recurring and novel classes in concept-drifting data streams. In *ICDM*, pages 1176–1181. IEEE, 2011.

[Moreira dos Reis *et al.*, 2018] Denis Moreira dos Reis, André Maletzke, Diego F Silva, and Gustavo EAPA Batista. Classifying and counting with recurrent contexts. In *SIGKDD*, pages 1983–1992, 2018.

[Namitha and Santhosh Kumar, 2020] K Namitha and G Santhosh Kumar. Learning in the presence of concept recurrence in data stream clustering. *Big Data*, pages 1–28, 2020.

[Reich *et al.*, 2019a] Christian Reich, Christina Nicolaou, Ahmad Mansour, and Kristof Van Laerhoven. Bayesian estimation of recurrent changepoints for signal segmentation and anomaly detection. In *EUSIPCO*. IEEE, 2019.

[Reich *et al.*, 2019b] Christian Reich, Christina Nicolaou, Ahmad Mansour, and Kristof Van Laerhoven. Detection of machine tool anomalies from bayesian changepoint recurrence estimation. In *INDIN*. IEEE, 2019.

[Sun *et al.*, 2021] Linjin Sun, Yangjian Ji, Mingrui Zhu, Fu Gu, Feng Dai, and Ke Li. A new predictive method supporting streaming data with hybrid recurring concept drifts in process industry. *Computers and Industrial Engineering*, 161:107625, 2021.

[Suárez-Cetrulo *et al.*, 2023] Andrés L. Suárez-Cetrulo, David Quintana, and Alejandro Cervantes. A survey on machine learning for recurring concept drifting data streams. *Expert Syst Appl*, page 118934, 2023.

[Verwimp *et al.*, 2023] Eli Verwimp, Kuo Yang, Sarah Parisot, Lanqing Hong, Steven McDonagh, Eduardo P, Matthias De Lange, and Tinne T. Clad: A realistic continual learning benchmark for autonomous driving. *Neural Networks*, pages 659–669, 2023.

[Wu *et al.*, 2012] Xindong Wu, Peipei Li, and Xuegang Hu. Learning from concept drifting data streams with unlabeled data. *Neurocomputing*, pages 145–155, 2012.

[Wu *et al.*, 2021] Ocean Wu, Yun Sing Koh, G Dobbie, and T Lacombe. Nacre: Proactive recurrent concept drift detection in data streams. In *IJCNN*, pages 1–8. IEEE, 2021.

[Wu *et al.*, 2022] Ocean Wu, Yun Sing Koh, Gillian Dobbie, and T Lacombe. Probabilistic exact adaptive random forest for recurrent concepts in data streams. *Int. J. Data Sci. Anal.*, pages 1–16, 2022.

[Wu *et al.*, 2023] Haitao Wu, Dolgintseva Elizaveta, Anastasia Zhadan, and Ovanes Petrosian. Forecasting online adaptation methods for energy domain. *Eng Appl Artif Intell*, page 106499, 2023.

[Yang *et al.*, 2006] Ying Yang, Xindong Wu, and Xingquan Zhu. Mining in anticipation for concept change: Proactive-reactive prediction in data streams. *DM and KD*, pages 261–289, 2006.

[Zhang *et al.*, 1996] Tian Zhang, R Ramakrishnan, and M Livny. Birch: an efficient data clustering method for very large databases. *ACM sigmod record*, pages 103–114, 1996.

[Zheng *et al.*, 2021] Xiulin Zheng, Peipei Li, Xuegang Hu, and Kui Yu. Semi-supervised classification on data streams with recurring concept drift and concept evolution. *Knowl Base Syst*, page 106749, 2021.