Machine Learning (2020) 109:533–568
https://doi.org/10.1007/s10994-019-05835-w

Handling concept drift via model reuse

Peng Zhao[1] · **Le-Wen Cai**[1] · **Zhi-Hua Zhou**[1]

Received: 2 May 2019 / Revised: 16 July 2019 / Accepted: 6 September 2019 / Published online: 10 October 2019

Abstract
In many real-world applications, data are often collected in the form of a stream, and thus the distribution usually changes in nature, which is referred to as *concept drift* in the literature. We propose a novel and effective approach to handle concept drift via model reuse, that is, reusing models trained on previous data to tackle the changes. Each model is associated with a weight representing its reusability towards current data, and the weight is adaptively adjusted according to the performance of the model. We provide both generalization and regret analysis to justify the superiority of our approach. Experimental results also validate its efficacy on both synthetic and real-world datasets.

Keywords Concept drift · Model reuse · Non-stationary environments

1 Introduction

With the rapid development in data collection technology, it is of great importance to analyze and extract knowledge from a vast number of data. However, data are commonly in a streaming form and are usually collected from non-stationary environments, and thus they are evolving in nature. In other words, the joint distribution between the input feature and the target label will change, which is also referred to as *concept drift* in the literature (Gama et al. 2014). If we simply ignore the distribution change when learning from the evolving data stream, the performance will dramatically drop, which is not empirically and theoretically desirable for these tasks. Consequently, the concept drift problem has become one of the most challenging issues for data stream learning and has drawn researchers' attention to design practically effective and theoretically sound algorithms.

Editors: Kee-Eung Kim and Jun Zhu.

✉ Zhi-Hua Zhou
zhouzh@lamda.nju.edu.cn

Peng Zhao
zhaop@lamda.nju.edu.cn

Le-Wen Cai
cailw@lamda.nju.edu.cn

[1] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

However, data stream with concept drift is essentially almost impossible to learn (predict) if there is no assumption on the distribution change. That is, if the underlying distribution changes arbitrarily or even adversarially, there is no hope to learn a good model to make the prediction. We share the same assumption with most of previous works, that is, *there contains some useful knowledge for the future prediction in previous data*. No matter sliding window based approaches (Klinkenberg and Joachims 2000; Bifet and Gavaldà 2007; Kuncheva and Zliobaite 2009), forgetting based strategies (Koychev 2000; Klinkenberg 2004; Zhao at al. 2019) or ensemble based methods (Kolter and Maloof 2005, 2007; Sun et al. 2018), they share the same assumption, whereas the difference is how to exploit and utilize the knowledge in previous data.

Another issue is that most previous works on handling concept drift focus on the algorithm design, only a few works consider the theoretical property (Helmbold and Long 1994; Crammer et al. 2010; Mohri and Medina 2012). There are some works proposing algorithms along with theoretical analysis, for instance, (Kolter and Maloof 2005) provides mistake and loss bounds and guarantees that the performance of the proposed approach is relative to the performance of the base learner. (Harel et al. 2014) detects concept drift via resampling and provides bounds on differentiates based on stability analysis. Nevertheless, seldom have clear theoretical guarantees or justifications on why and how to utilize knowledge in previous data to fight with concept drift.

In this paper, we propose a novel and effective approach for handling CONcept Drift via mOdel Reuse, or Condor. It consists of two modules, `ModelUpdate` module aims at exploiting knowledge from previous models to help build the new model, while `WeightUpdate` module adaptively assigns weights for previous models according to their performance, where the weights represent the reusability of previous models towards current data. We theoretically justify the advantage of the ModelUpdate module from the aspect of generalization analysis, showing that our approach of model reuse can benefit from a good weighted combination of previous models. Meanwhile, the WeightUpdate module guarantees that the weights will finally concentrate on the better-fit models and thus provides a good weighted combination of previous models as the initialization for ModelUpdate module to train the new model. Therefore, they together make our proposed algorithm successful. Besides, we investigate the overall performance of the whole data stream by the dynamic regret analysis. Experiments on both synthetic and real-world datasets validate the superiority of our approach, and the empirical studies demonstrate the effectiveness of the model reuse mechanism for handling concept drift.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 proposes our approach. Sections 4 and 5 present theoretical analysis. Section 6 reports experimental results. Finally, we conclude the paper and discuss future work in Sect. 7.

2 Related work

Concept Drift The phenomenon of concept drift has been well-recognized in recent researches (Gama et al. 2014; Gomes et al. 2017). The capability of handling concept drift is one of fundamental requirements of *learnware* (Zhou 2016), and also a crucial step towards robust and reliable artificial intelligence (Dietterich 2017, 2019).

Basically, if there is not any structural information about the data stream, we shall not expect to learn from historical data and make any meaningful prediction. Therefore, a common assumption on the concept drift in the stream as aforementioned is that there contains useful knowledge in previous data. Particularly, most previous works assume that the nearby

data items include more useful information concerning the current data, and thus researchers propose plenty of approaches based on the *sliding window* and *forgetting* mechanisms. Sliding window based approaches maintain the nearest data items and discard old items, with a fixed or adaptive window size (Klinkenberg and Joachims 2000; Kuncheva and Zliobaite 2009). Forgetting based approaches do not explicitly drop old items but downweight previous data items according to their age (Koychev 2000; Klinkenberg 2004).

Another important category falls into the *ensemble* based approaches, they adaptively add or delete base classifiers and dynamically adjust weights when dealing with the evolving data stream. A series of work borrows the idea from boosting (Schapire 1990) and online boosting (Beygelzimer et al. 2015) by endowing the learning system with the capability to cope with non-stationarity via dynamically adjusting weights of classifiers. Take a few representatives, dynamic weighted majority (DWM) dynamically creates and removes weighted experts in response to distribution changes (Kolter and Maloof 2003, 2007). Additive expert ensemble (AddExp) adaptively adjusts the additive expert pool and provides mistake bounds as theoretical guarantees (Kolter and Maloof 2005). Learning in non-stationary environments (Learn^{++}.NSE) trains one new classifier for each batch of data and then combines these classifiers (Elwell and Polikar 2011). Hybrid Forest (Rad and Haeri 2019) considers to adaptively charge to the suitable classifier by the multiple classifier selection according to the 'hybrid power' quantity defined therein. Plenty of approaches are proposed to learn from evolving data stream; we refer the reader to comprehensive surveys (Gama et al. 2014; Gomes et al. 2017). For boosting and ensemble approaches, the reader is recommended books (Schapire and Freund 2012; Zhou 2012).

Our approach is kind of similar to DWM and AddExp on the surface. We all maintain a model pool and adjust weights to penalty models with poor performance. However, we differ from the model update procedure, and they ignore to leverage previous knowledge and reuse models to help build a new model and update the model pool. Besides, our weight update strategies are also different.

Model Reuse Model reuse is an important learning problem, also named as model transfer, hypothesis transfer, or learning from auxiliary classifiers. The basic setting is that one desires to reuse pre-trained models to help further model building, especially when the data are too scarce to train a fair model directly. A series of works lies in the idea of biased regularization, which leverages previous models as the biased regularizer in addition to empirical risk minimization, and achieves a good performance in plenty of scenarios (Duan et al. 2009; Tommasi et al. 2010, 2014). (Reddi et al. 2015) adopts such techniques in the covariate shift to reduce the variance introduced by the standard reweighted empirical risk minimization. There are also some other attempts and applications, for instance, (Segev et al. 2017) develops model reuse by random forests, (Ye et al. 2018) reuses models for transferring the invariant meta feature representation between heterogeneous feature space by semantic mapping and (Wu et al. 2019) proposes a novel model reuse method for multi-party learning by optimizing the global behavior of an ensemble of heterogeneous local models. Besides, (Li et al. 2013) applies model reuse technique to adapt different performance measures. Apart from algorithm design, theoretical foundations are recently established by algorithmic stability (Kuzborskij and Orabona 2013), Rademacher complexity (Kuzborskij and Orabona 2017) and transformation functions (Du et al. 2017).

Our paper proposes to handle concept drift problem via utilizing model reuse learning. The idea of exploiting knowledge by reusing previous model is reminiscent of some past works coping with concept drift by transfer learning, like the temporal inductive transfer (TIX) approach (Forman 2006) and the diversity and transfer-based ensemble learning (DTEL)

approach (Sun et al. 2018). Both of them are batch-style approaches, that is, they need to receive a batch of data each time, whereas ours can update either in an incremental style or a batch update mode. TIX concatenates the predictions from previous models into the feature of next data batch as the new data, and a new model is learned from the augmented data batch. DTEL chooses decision tree as the base learner and builds a new tree by directly using the latest data batch along with "fine-tuning" previous models by a direct tree structural adaptation. It maintains a fixed size model pool with the selection criteria based on diversity measurement. They both do not depict the reusability of previous models, which is carried out by WeightUpdate module in our approach. Last but not the least important, our approach is proposed with sound theoretical guarantees, in particular, we carry out a generalization justification on why and how to reuse previous models. Nevertheless, theirs are not theoretically clear in general.

3 Proposed approach

We consider the scenario that data are coming one by one sequentially, and there may emerge concept drift in the data stream. We propose a novel algorithm, Condor, to handle the potential concept drift. The high-level idea is to adapt the non-stationarity by extracting the useful knowledge in previous data. Therefore, we propose to adopt the model reuse technique to handle concept drift. To realize this goal, we design two core modules: `ModelUpdate` module and `WeightUpdate` module, both are essential in reusing knowledge in previous data and automatically adapting the non-stationary environments. In the following, we give detailed descriptions of the proposed algorithm.

Condor periodically updates the model when achieving the maximum update period p, which is a data-dependent parameter, reflecting the inherent extent of fluctuation and non-stationarity. Meanwhile, we also adopt a concept drift detector $\mathfrak{D}$ in the background in case of abrupt changes. When the maximum update period is achieved, or the abrupt change is detected, instead of resetting the model pool and incrementally training a new model, we aim at reusing knowledge in previous models in a weighted manner to enhance the overall performance. Therefore, there are two issues to address:

- Suppose each previous model is associated with a proper weight. How to reuse these models weightedly to train the new model?
- How to obtain a proper weight for each previous model? Note that the weight should represent model's "reusability" towards current data.

To address the above two issues, our approach consists of the following two important modules.

(1) `ModelUpdate` by model reuse: we reuse previous models to build the new model and update model pool, by making use of the biased regularization technique for multiple model reuse learning.
(2) `WeightUpdate` by expert advice: The weight of each previous model is updated according to its performance on the current data, in an exponential weighted average manner.

We present descriptions of these two modules in next two subsections.

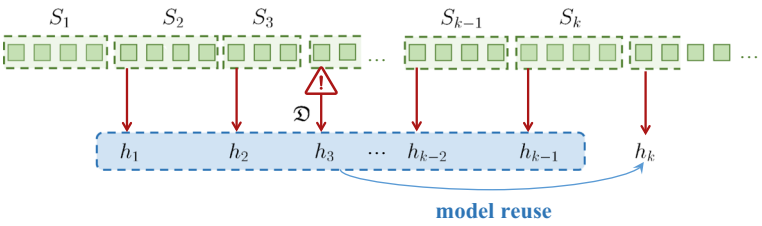


Fig. 1 Illustration of main idea: our approach periodically conducts the model update, and adopts the drift detector $\mathfrak{D}$ in the background in case of the abrupt change. During the model update, on one hand, we utilize the data items in current epoch S_k; on the other hand, we reuse knowledge from previous models ($\{h_1, h_2, \ldots, h_{k-1}\}$) via model reuse

3.1 Model update by model reuse

During the model update stage, we propose to reuse previous models weightedly to adapt the current data epoch. Note that in this subsection, the weight of each previous model is supposed to be given in advance, and the weight update procedure will be specified in the next subsection.

Consider the kth model update as illustrated in Fig. 1, we desire to use previous models in the model pool $H = \{h_1, \ldots, h_{k-1}\}$ along with the current data epoch S_k to train a new model h_k. With a slight abuse of notations, we denote $S_k = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$.

We first adopt linear classifier as the base model and reusing previous models via *biased regularization* technique (Schölkopf et al. 2001; Tommasi et al. 2014). We remark that one may use kernel methods, and Nyström method as well as random Fourier features to transform the kernelized problem into a linear one, details can be found in the paper (Yang et al. 2012).

When the base model is linear classifier, the new model will be obtained according to the following regularized empirical risk minimization,

$$\hat{\mathbf{w}}_k = \arg\min_{\mathbf{w}} \left\{ \frac{1}{m} \sum_{i=1}^{m} \ell\left(\langle \mathbf{w}, \mathbf{x}_i \rangle, y_i\right) + \mu \Omega(\mathbf{w}, \mathbf{w}_p) \right\}, \tag{1}$$

where $\ell : \mathbb{R} \times \mathcal{Y} \mapsto \mathbb{R}_+$ is the non-negative loss function, $\mu > 0$ is a positive trade-off regularization coefficient. Besides, we denote by $\mathcal{H}$ the hypothesis set, and $\Omega : \mathcal{H} \times \mathcal{H} \mapsto \mathbb{R}_+$ is a model reuse regularizer specifying the final model to be built upon the previous models, satisfying $\Omega(\mathbf{w}_p, \mathbf{w}_p) = 0$ with a typical choice as $\|\mathbf{w} - \mathbf{w}_p\|^2$. And $\mathbf{w}_p$ is the linear weighted combination of previous models, namely, $\mathbf{w}_p = \sum_{j=1}^{k-1} \beta_j \hat{\mathbf{w}}_j$, where β_j is the weight associated with previous model h_j, representing the reusability of each model on data in current epoch.

Since data in a relatively stationary epoch are usually scarce in evolving data stream, model reuse mechanism is quite useful as it reduces the sample complexity by reusing previous models as a basis, which will be theoretically investigated by generalization analysis in Theorem 1.

For simplicity, in this paper, we choose the square loss with ℓ_2 regularization in practical implementation, essentially, Least Square Support Vector Machine (LS-SVM) (Suykens et al. 2002). Meanwhile, the optimal solution of 1 can be expressed in the form of $\hat{\mathbf{w}}_k = \sum_{i=1}^{m} \alpha_i \mathbf{x}_i$, where the coefficients $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_m]^{\mathrm{T}}$ can be obtained by analyzing its optimality condition. Specifically, it is sufficient to solve the following linear Karush-Kuhn-Tucker (KKT)

Algorithm 1 Condor

Require: Data stream $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)\}$. Epoch size (maximum update period) p; model pool size K; step size η; Drift detector $\mathfrak{D}(\delta, \mathbf{x}, y)$ with threshold δ.
Ensure: Prediction $\hat{y}_t$, where $t = 1, \ldots, T$; and returned model pool H.
Initialize model h_1 on first (or a couple of) data items;
Initialize model pool $H \leftarrow \{h_1\}$ and weight $\beta_{1,1} \leftarrow 1$;
for $t = 1$ **to** T **do**
  Receive $\mathbf{x}_t$;
  for $k = 1$ **to** $|H|$ **do**
    $\hat{y}_{t,k} \leftarrow h_k(\mathbf{x}_t)$;
  end for
  $\hat{y}_t \leftarrow \sum_{k=1}^{|H|} \beta_{t,k} \hat{y}_{t,k} / \sum_{k=1}^{|H|} \beta_{t,k}$;
  Receive y_t;
  for $k = 1$ **to** $|H|$ **do**
    $\beta_{t+1,k} \propto \beta_{t,k} \exp\{-\eta \ell(\hat{y}_{t,k}, y_t)\}$;
  end for
  if $t \mod p = 0$ **or** $\mathfrak{D}(\delta, \mathbf{x}_t, y_t) > 0$ **then**
    $h \leftarrow \texttt{ModelUpdate}(S_{|H|}, H, \{\beta_1, \ldots, \beta_{|H|}\})$;
    $H \leftarrow H \cup \{h\}$;
    if $|H| > K$ **then**
      Remove the oldest model from H.
    end if
    for $k = 1$ **to** $|H|$ **do**
      Reinitialize the weights: $\beta_{1,k} \leftarrow 1/|H|$;
    end for
  end if
end for

system (Suykens et al. 2002, Chapter 3, pp. 73),

$$\begin{bmatrix} \mathbf{K} + \frac{1}{\mu}\mathbf{I} & \mathbf{1} \\ \mathbf{1} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{y} - \sum_{j=1}^{k-1} \beta_j \hat{\mathbf{y}}_j \\ 0 \end{bmatrix}, \tag{2}$$

where $\mathbf{K}$ is the linear kernel matrix, i.e., $\mathbf{K}_{ij} = \mathbf{x}_i^T \mathbf{x}_j$. Besides, $\mathbf{y}$ and $\hat{\mathbf{y}}_j$ are vectors containing labels of data stream and predictions of the previous jth model, that is, $\mathbf{y} = [y_1, \ldots, y_m]^{\mathrm{T}}$ and $\hat{\mathbf{y}}_j = [\langle \hat{\mathbf{w}}_j, \mathbf{x}_1 \rangle, \ldots, \langle \hat{\mathbf{w}}_j, \mathbf{x}_m \rangle]^{\mathrm{T}}$. The KKT system can be solved by the Gaussian elimination method. For larger datasets, in order to alleviate memory cost and reduce computational complexity, the use of iteration methods are recommended, for instance, the conjugate gradient iterative methods (Suykens et al. 2002, Chapter 3.4.1, pp. 86).

If the concept drift occurs very frequently or data stream accumulates for a long time, the size of the model pool will explode supposing there is no model retirement mechanism. Therefore, we set the maximum of model pool size as K. Apparently, we can keep K of all models with the largest diversity as done in the work of Sun et al. (2018). For simplicity, we only keep the latest K ones in the model pool.

Remark 1 The biased regularization model reuse learning is not limited in the binary classification task, and can be extended to the multi-class scenario. We defer notations and corresponding theoretical analyses in Sect. 5. Meanwhile, our framework can generalize to other base models with suitable model reuse mechanisms. For instance, we can use the decision tree as the base classifier, along with the suitable model reuse strategy. More extensions will be investigated in the future work.

3.2 Weight update by expert advice

During the weight update stage, we propose to update the weights such that the weight can represent the reusability of the model towards current data scratch. To this end, we update the weight of each model by *prediction with expert advice* technique (Cesa-Bianchi et al. 1997; Cesa-Bianchi and Lugosi 2006). The intuitive idea is to adaptively adjust the weight distribution of previous models according to their performance in order to reflect their reusability.

Specifically, when the ModelUpdate stage finishes, the weight distribution in the model pool H will reinitialize. We adopt a uniform initialization: $\beta_{1,k} = 1/|H|$, for $k = 1, \ldots, |H|$. After the initialization, we update the weight as follows: we first receive the new data item $\mathbf{x}_t$, and each previous model will then provide its prediction $\hat{y}_{t,k}$. The final prediction $\hat{y}_t$ is made based on a weighted combination of those predictions ($\hat{y}_{t,k}$, for $k = 1, \ldots, |H|$). Next, the true label is revealed as y_t, and we will update the weight distribution according to the loss each model suffers, in an exponential weighted manner,

$$\beta_{t+1,k} \propto \beta_{t,k} \exp\{-\eta \ell(\hat{y}_{t,k}, y_t)\}. \tag{3}$$

The weight update procedure in Eq. (3) is simple yet efficient. Next, we show the WeightUpdate mechanism returns a good weight distribution, implying our approach can reuse previous models properly. We have the following observation regarding the weight distribution.

Observation 1 (Weight Concentration) *During the WeightUpdate procedure in epoch S_k,the weights will concentrate on those previous models who suffer a small cumulative loss on S_k.*

Proof By a simple analysis on the ModelUpdate procedure, we know that the weight associated with the jth previous model is proportional to $\beta_{1,j} \exp\{-\eta L_{S_k}^{(j)}\}$, where $L_{S_k}^{(j)} = \sum_{i \in S_k} \ell(h_j(\mathbf{x}_i), y_i)$ and $j = 1, \ldots, k-1$. Therefore, the model suffers a small cumulative loss will be associated with a large weight. □

Notwithstanding its simplicity, this observation plays a vital role in making our approach successful. It guarantees that the algorithm adaptively assigns more weights on better-fit previous models and thus essentially depicts the "reusability" of each model. Therefore, the weight update procedure is particularly useful when there emerge recurring concepts in the data stream. We give empirical evidence as support in Sects. 6.3 and 6.4.

The overall procedure of proposed approach Condor is summarized in Algorithm 1, where line 10-12 mainly conduct the weight update procedure presented in Eq. (3) and the model update procedure shown in line 14 is realized by Eq. (1). From the update procedures, we conclude that the overall space complexity is $O(d(K + p))$ because Condor needs to store p data items in each epoch and K previous models, where d is the dimensionality of the feature. The overall time complexity is $O(p^3 + dp^2 + dpK)$, specifically, the $O(p^3 + dp^2)$ term is mainly devoted in solving the KKT system of Eq. (2), while the $O(dpK)$ term is for the prediction of p instances in each epoch. We remark that the time complexity can be further accelerated by the iterative methods in solving LS-SVM, particularly when the matrix of the KKT system has a small condition number (Shewchuk 1994).

4 Theoretical analysis

In this section, we provide theoretical justifications of the proposed approach Condor. Observation 1 in the last paragraph demonstrates that the WeightUpdate mechanism provides a good weight distribution for the weighted combination of previous models as the initialization. We will further show that the ModelReuse mechanism is capable of taking advantage of this initialization when training the new model. Therefore, they together make our proposed approach successful.

To this end, we present the results in the following two aspects:

- Local analysis: investigate the performance in each epoch, from both generalization and regret aspects.
- Global analysis: examine the regret on the whole data stream globally.

To better present the main results, we defer all the proofs in the appendix.

4.1 Local analysis

Local analysis aims to scrutinize the performance of a particular epoch. On the one hand, we are concerned about the *generalization ability* of the model obtained by the ModelUpdate module. Second, we examine the *cumulative regret* of predictions.

Let us consider the epoch S_k, the ModelUpdate module reuses previous models $h_1, \ldots, h_{k-1}$ to help build the new model h_k, as shown in Fig. 1. To simplify the presentation, we introduce some notations. Suppose the length of data stream S is T, and is partitioned into k epochs, $S_2, \ldots, S_k$.[1] For epoch S_k, we assume the distribution is identical, i.e., S_k is a sample of m_k points drawn i.i.d. according to distribution $\mathcal{D}_k$, where m_k denotes its length.

Generalization analysis

First, we conduct generalization analysis on the ModelUpdate module. Define the risk and empirical risk of the hypothesis (model) h on epoch S_k by

$$R(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_k}[\ell(h(\mathbf{x}), y)], \quad \hat{R}(h) = \frac{1}{m_k}\sum_{i\in S_k}\ell(h(\mathbf{x}_i), y_i).$$

Here, with a slight abuse of notations, we also adopt S_k to denote the index included in the epoch, and $\hat{R}(h)$ instead of $\hat{R}_{S_k}(h)$ for simplicity. The new model h_k is built and updated on epoch S_k via the ModelUpdate module, then we have the following generalization error bound.

Theorem 1 *Assume that the non-negative loss function $\ell : \mathbb{R} \times \mathcal{Y} \mapsto \mathbb{R}_+$ is bounded by $M \geq 0$. Meanwhile, for all $y \in \mathcal{Y}$, $\ell(\cdot, y)$ is L-Lipschitz continuous. Also, assume the regularizer $\Omega : \mathcal{H} \times \mathcal{H} \mapsto \mathbb{R}_+$ is a non-negative and λ-strongly convex function in its first argument w.r.t. a norm $\|\cdot\|$. Given the source model h_p, which is a linear combination of previous models. Then, for any $\delta > 0$, with probability at least $1-\delta$, we have,*[2]

$$R(h_k) - \hat{R}(h_k) \leq L\sqrt{\frac{\epsilon_1}{m}} + 3\sqrt{\frac{\epsilon_2 \log(1/\delta)}{m}} + \frac{3M\log(1/\delta)}{4m},$$

[1] Here, we start from epoch 2, since the first epoch cannot utilize the WeightUpdate procedure.

[2] We use m instead of m_k for simplicity.

where $\epsilon_1 = \frac{2B^2 R_p}{\lambda\mu}$ *and* $\epsilon_2 = \frac{M}{4}R_p + 2LM\sqrt{\frac{\epsilon_1}{m}}$. *Besides,* $B = \sup_{\boldsymbol{x}\in\mathcal{X}}\|\boldsymbol{x}\|_\star$ *and* $R_p = R(h_p) = \mathbb{E}_{(\boldsymbol{x},y)\sim S_k}[\ell(h_p(\boldsymbol{x}), y)]$, *representing the risk of reusing model on the underlying distribution of current data.*

To better present the results, we only keep the leading terms w.r.t. m and R_p, *and obtain that*

$$R(h_k) - \hat{R}(h_k) = O\left(\frac{\tilde{\epsilon}_1}{\sqrt{m}} + \frac{\tilde{\epsilon}_2}{m}\right), \tag{4}$$

where $\tilde{\epsilon}_1 = \left(\sqrt{\frac{R_p}{\lambda\mu}} + \sqrt[4]{\frac{R_p}{\lambda\mu m}}\right)$ *and* $\tilde{\epsilon}_2 = \left(\sqrt{\frac{1}{\lambda}} + \sqrt[4]{\frac{1}{\lambda}}\right)$.

Remark 2 Equation (4) shows that the model returned by the ModelUpdate procedure enjoys an $O(1/\sqrt{m})$ generalization bound under certain conditions. In particular, when the source model (a weighted combination of previous models) is sufficiently good, that is, it has a small risk on the current distribution $\mathcal{D}_k$ (i.e., when $R_p \to 0$), we can obtain an $O(1/m)$ bound, a fast rate convergence guarantee. Such a result is very much desired because the number of data items in each epoch is typically limited. Theorem 1 theoretically justifies the effectiveness of the model reuse mechanism that incorporates the knowledge of previous models to learn the new model on the current data, because such a mechanism can significantly reduce the sample complexity, especially if we can reuse previous models properly. Meanwhile, the WeightUpdate mechanism gives a satisfying weight distribution for reusing previous models, because it guarantees to concentrate more weights on those better-fit models (see Observation 1).

Remark 3 It is noteworthy to mention that the generalization analysis does not require the loss function be necessarily convex. We only assume a bounded and Lipschitz condition (in Theorem 1) for the loss function, along with strongly convexity condition for the regularizer. These two conditions can be easily satisfied by common models. For example, in SVMs we use ℓ_2 regularization, which is 2-strongly convex, and the hinge loss $\ell(z, y) = [1 - yz]_+$, which is 1-Lipschitz continuous. We remark that the bounded condition of the loss function can be achieved with bounded model space and training items. This is a general assumption which also appears in the analysis of support vector machine (Mohri et al. 2018, Theorem 5.10), online learning (Cesa-Bianchi and Lugosi 2006, Theorem 2.2) and learning to rank (Mohri et al. 2018, Theorem 10.1 and Corollary 10.2). The limitation of constraining the model space can be possibly relaxed by the technique of localized complexity (Bartlett et al. 2005; Sridharan et al. 2008), which will be investigated in the future work.

Remark 4 The main techniques in the proof are inspired by Kuzborskij and Orabona (2017), but we differ in two aspects. First, their analysis requires a smoothness condition for loss functions, and thus their results are not suitable under our conditions. Second, we extended the analysis of model reuse to multi-class scenarios, and results are presented in Sect. 5 as an independent part for a clear presentation. In addition, the work of Reddi et al. (2015) utilizes biased regularization technique to reduce the variance of the importance sampling estimator in covariate shift scenarios, and they give the theoretical analysis. We remark that their results focus on the deviation of expected risk between the obtained model and the vanilla importance sampling estimator, while they do not give the generalization error of the obtained model, i.e., $R(h) - \hat{R}(h)$. More importantly, their results do not exhibit a phenomenon of fast-rate convergence when the reusable model is good. On the contrary, we present a fast-rate generalization error analysis, theoretically justifying the advantage of the model reuse mechanism.

Regret analysis

In order to proceed to the regret analysis, we need to introduce more notations. Let L_T be the global cumulative loss on the whole data stream S, namely, $L_T = \sum_{i=1}^{T} \ell(\hat{y}_i, y_i)$. Meanwhile, on epoch S_k, let L_{S_k} be the cumulative loss over data in epoch S_k suffered by our approach, and $L_{S_k}^{(j)}$ as the local cumulative loss over data in epoch S_k suffered by the previous model h_j,

$$L_{S_k} = \sum_{i \in S_k} \ell(\hat{y}_i, y_i), \quad L_{S_k}^{(j)} = \sum_{i \in S_k} \ell(h_j(\mathbf{x}_i), y_i).$$

We adopt the concept of *cumulative regret* (or regret) from online learning (Zinkevich 2003; Cesa-Bianchi and Lugosi 2006; Hazan 2016) as the performance measurement, where regret is defined as the difference between the accumulated loss of the predictions and that of a particular expert.

We demonstrate that our approach suffers a small cumulative loss and is able to benefit from *recurring concept drift* scenarios.

Theorem 2 (Local Regret Cesa-Bianchi and Lugosi (2006)) *Assume that the loss function* $\ell : \mathbb{R} \times \mathcal{Y} \mapsto \mathbb{R}_+$ *is convex in its first argument and takes values in* $[0, 1]$. *When the step size is set as* $\eta = \sqrt{(8 \ln(k-1))/m_k}$, *then we have*

$$\text{Regret}_{S_k} = L_{S_k} - \min_{j=1,\ldots,k-1} L_{S_k}^{(j)} \leq \sqrt{(m_k/2)\ln(k-1)}. \tag{5}$$

Furthermore, suppose we know that there exists a previous model which matches current data quite well. Then, by setting the step size as $\eta = \ln(1 + \sqrt{2\ln(k-1)/L_{j_k^*}})$, *where* $L_{j_k^*} = \min_{j=1,\ldots,k-1} L_{S_k}^{(j)}$ *is the cumulative loss of the best-fit previous model, we have*

$$\text{Regret}_{S_k} = L_{S_k} - L_{j_k^*} \leq \sqrt{2L_{j_k^*}\ln(k-1)} + \ln(k-1). \tag{6}$$

Apparently, the quantity $L_{j_k^*}$ can be only available after all m_k rounds predictions. However, it can be compensated by "doubling trick" (Cesa-Bianchi et al. 1997), letting η change according to the current best previous model.

From the regret bound in the above statement [Eqs. (5) and (6)], we can see that the order of regret bound can be substantially improved from a typical $O(\sqrt{m_k})$ to $O(\ln k)$, independent from the number of data items in the epoch, providing $L_{k^*} \ll \sqrt{m_k}$, that is, the cumulative loss of the best-fit previous model is small.

Remark 5 Theorem 2 implies that if the concept of epoch S_k or a similar concept has emerged previously, our approach enjoys a substantially improved local regret providing a proper step size is chosen (essentially, let the step size be larger). The theory accords to our intuition on why model reuse helps for concept drift data stream. In many situations, although the distribution underlying might change over time, the concepts can be recurring, i.e., disappear and re-appear (Katakis et al. 2010; Gama and Kosina 2014). Thus, the conclusion demonstrates that our approach can benefit from such recurring concepts or similar concepts.

4.2 Global analysis

In this part, we investigate the global behavior of the proposed approach, namely, the performance on the whole data stream.

We remark that in the local regret analysis, Theorem 2 gives the (static) regret analysis, that is, the benchmark in Eqs. (5) and (6) is the cumulative loss of the *fixed* best-fit previous model. This is reasonable since the underlying distribution in each local epoch is considered identical. However, the static regret is not suitable for the global regret analysis. The rationale behind static regret is that the best fixed decision in hindsight is reasonably good over the data stream, nevertheless, the optimal decision is drifting over time in non-stationary environments. We thus adopt the *dynamic regret* (Zinkevich 2003; Besbes et al. 2015) denoted by "D-Regret" as the performance measure, a more stringent metric, which compares the predictions to a *time-varying* comparator sequence.

Theorem 3 (Global Dynamic Regret) *Assume that the loss function* $\ell : \mathbb{R} \times \mathcal{Y} \mapsto \mathbb{R}_+$ *is convex in its first argument and takes the values in* $[0, 1]$. *By setting the epoch size (maximal update period)* $p = \lceil \sqrt[3]{\frac{\ln K}{2}} (T/2V_T)^{2/3} \rceil$ *and the step size in epoch* S_k *as* $\eta_k = \sqrt{(8 \ln K)/m_k}$,

$$\text{D-Regret}_T = \sum_{t=1}^{T} \ell(\hat{y}_t, y_t) - \sum_{t=1}^{T} \ell(h_t^*(\boldsymbol{x}_t), y_t) = O\Big(V_T^{1/3} T^{2/3}\Big), \tag{7}$$

where $h_t^* \in \arg\min_{h \in H} \ell(h(\boldsymbol{x}_t), y_t)$ *is the optimal classifier of this round. Besides,* V_T *is the function variation defined by*

$$V_T = \sum_{t=2}^{T} \sup_{h \in H} |\ell(h(\boldsymbol{x}_{t-1}), y_{t-1}) - \ell(h(\boldsymbol{x}_t), y_t)|. \tag{8}$$

Evidently, the function variation measures the non-stationarity of the data stream.

Remark 6 The regret bound in Theorem 3 is different from traditional (static) regret bounds (Hazan 2016). Essentially, it measures the difference between the global cumulative loss with the sum of local cumulative loss suffered by the best possible model. Therefore, the comparator dynamically changes and is time-varying, which depicts the distribution change in the sequence. It is thus named as *dynamic regret*, more suitable to be the performance measure in non-stationary environments.

Remark 7 The term V_T involved in the bound is called function variation, characterizing the non-stationarity of the data stream essentially. That is, the more non-stationary the data stream is, the larger the value of V_T will be. In this sense, the dynamic regret bound in Eq. (7) is adaptive to non-stationarity of the data stream.

Note that the optimal choice of epoch size p depends on the function variation V_T, which is unfortunately unknown ahead of time. There are at least two ways to eliminate such an undesired dependence: either one knows the variation budget B_T such that $V_T \leq B_T$, then p can be set according to B_T; or one can appeal to doubling trick (Cesa-Bianchi et al. 1997) or grid search (Koolen et al. 2014; Zhang et al. 2018) to replace the unknown quantity by those quantities empirically attainable. We will not go for details as this exceeds the scope of this paper.

5 Multi-class model reuse learning

In this section, we extend the generalization analysis from the binary model reuse learning to the multi-class scenario. All proofs are deferred to "Appendix D".

We first introduce new notations for a clear presentation, as the notations in multi-class learning scenarios are slightly different from those in the binary case.

Let $\mathcal{X}$ denote the input feature space and $\mathcal{Y} = \{1, 2, \ldots, c\}$ denote the target label space. Our analysis acts on the last data epoch $S_k = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{m_k}, y_{m_k})\}$, a sample of m_k points drawn i.i.d. according to distribution $\mathcal{D}_k$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathcal{Y}$ with only a single class from $\{1, \ldots, c\}$. Given the multi-class hypothesis set $\mathcal{H}$, any hypothesis $h \in \mathcal{H}$ maps from $\mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$, and makes the prediction by $\mathbf{x} \mapsto \arg\max_{y \in \mathcal{Y}} h(\mathbf{x}, y)$. This naturally rises the definition of *margin* $\rho_h(\mathbf{x}, y)$ of the hypothesis h at a labeled instance $(\mathbf{x}, y)$,

$$\rho_h(\mathbf{x}, y) = h(\mathbf{x}, y) - \max_{y' \neq y} h(\mathbf{x}, y').$$

Based on the margin and the non-negative loss function $\ell : \mathbb{R} \mapsto \mathbb{R}_+$, we can define the risk and empirical risk of a hypothesis h on epoch S_k as,

$$R(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_k}\left[\mathbf{1}_{\rho_h(\mathbf{x},y)} \leq 0\right], \quad \hat{R}_S(h) = \frac{1}{m_k}\sum_{i \in S_k} \ell(\rho_h(\mathbf{x}_i, y_i)).$$

One should be aware that the definition of the loss function in the multi-class scenario is different from that in the binary classification.

First, we identify the optimization formulation of multi-class biased regularization model reuse learning,

$$\hat{W} = \arg\min_W \left\{ \frac{1}{m}\sum_{i=1}^{m} \ell\left(\rho_{h_W}(\mathbf{x}_i, y_i)\right) + \mu\Omega(W, W_p) \right\}, \tag{9}$$

where $\ell : \mathbb{R} \times \mathcal{Y} \mapsto \mathbb{R}_+$ is a non-negative loss function and $h_W(\mathbf{x}) = W^{\mathrm{T}}\mathbf{x}$, $\mu > 0$ is a positive trade-off regularization coefficient. Besides, $\Omega : \mathcal{H} \times \mathcal{H} \mapsto \mathbb{R}_+$ is a model reuse regularizer specifying the final model to be built upon the previous models, satisfying $\Omega(W_p, W_p) = 0$ with a typical choice as $\|W - W_p\|_F^2$. And W_p is the linear weighted combination of previous models, namely, $W_p = \sum_{j=1}^{k-1} \beta_j \hat{W}_j$, where β_j is the weight associated with previous model h_j, representing the reusability of each model on data in current epoch.

Apart from the non-negative property, we suppose the loss function is *regular* as defined in the work of Lei et al. (2015).

Definition 1 (*Regular Loss, Cf. Definition 2* of Lei et al. 2015) We call a loss function $\ell : \mathbb{R} \mapsto \mathbb{R}$ is L-regular if it satisfies the following properties:

(i) $\ell(t)$ bounds the 0-1 loss from above: $\ell(t) \geq \mathbf{1}_{t \leq 0}$;
(ii) $\ell(t)$ is L-Lipschitz continuous, i.e., $|\ell(t_1) - \ell(t_2)| \leq L|t_1 - t_2|$;
(iii) $\ell(t)$ is decreasing and it has a zero point c_ℓ, i.e., there exists a c_ℓ such that $\ell(c_\ell) = 0$.

Our goal is to provide the generalization analysis, namely, to prove the convergence of risk $R(h)$ to the empirical risk $\hat{R}(h)$, and establish the rate. Since $\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_k}[\hat{R}(h)] \neq R(h)$, thus, we cannot directly utilize concentration inequalities to help. To make this feasible, we need to introduce the risk w.r.t. loss function ℓ,

$$R_\ell(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_k}\left[\ell(\rho_h(\mathbf{x}, y))\right].$$

From property (i) in Definition 1, we know that the risk $R(h)$ is a lower bound of $R_\ell(h)$, that is $R(h) \leq R_\ell(h)$. Thus, we only need to establish generalization bound between $R_\ell(h)$ and $\hat{R}_S(h)$. Evidently, $\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_k}[\hat{R}_S(h)] = R_\ell(h)$, thus we can utilize concentration inequalities again.

In the theoretical analysis, we specify the regularizer as square of Frobenius norm, namely, $\Omega(W, W_p) = \|W - W_p\|_F^2$, and provide the following generalization error bound.

Theorem 4 *Let $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{X}\times\mathcal{Y}}$ be a hypothesis set with $\mathcal{Y} = \{1, 2, \ldots, c\}$. Assume that the non-negative loss function $\ell : \mathbb{R} \mapsto \mathbb{R}_+$ is L-regular and M-bounded. Given the source model h_p, which is a linear combination of previous models. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds,*[3]

$$R(h_{\hat{W}}) - \hat{R}_S(h_{\hat{W}}) \leq 2Lc^2\sqrt{\frac{\epsilon_1}{m}} + 3\sqrt{\frac{\epsilon_2 \log(1/\delta)}{4m}} + \frac{3M\log(1/\delta)}{4m}.$$

where $\epsilon_1 = \frac{B^2 R_p}{2\mu}$ and $\epsilon_2 = M\left(8Lc^2\sqrt{\frac{\epsilon_1}{m}} + R_p\right)$. Besides, $B = \sup_{\boldsymbol{x}\in\mathcal{X}} \|\boldsymbol{x}\|_2$.
To better present the results, we only keep the leading term w.r.t. m and R_p, and we have

$$R(h_k) - \hat{R}_S(h_k) = O\left(\frac{c^2}{\sqrt{m}}\left(\sqrt{\frac{R_p}{\mu}} + \sqrt[4]{\frac{R_p}{\mu m}}\right) + \frac{1}{m}\sqrt[4]{\frac{1}{\mu}}\right), \tag{10}$$

where $R_p = R(h_p) = \mathbb{E}_{(\boldsymbol{x},y)\sim S_k}[\ell(\rho_{h_p}(\boldsymbol{x}, y))]$, representing the risk of reusing model on current distribution.

Remark 8 From Theorem 4, we can see that the main result and conclusion in the multi-class case is very similar to that in the binary case. In Eq. (10), we can see that Condor enjoys an $O(1/\sqrt{m})$ order generalization bound, which is consistent to the common learning guarantees. More importantly, Condor enjoys an $O(1/m)$ order fast rate generalization guarantees, when $R_p \mapsto 0$, namely, when the previous model h_p is highly "reusable" for the current data. This shows the effectiveness of the ModelUpdate module that reuses previous models to help build the new model in multi-class scenarios, since the number of data items in each epoch is usually limited, the model reuses mechanism is able to significantly reduce the sample by utilizing previous models as the basis when constructing the new model.

At last, it is noteworthy to mention that the current generalization error bound admits a quadratic dependence on the number of classes c, one might further sharp this to a radical dependence by applying the vector Rademacher complexity technique (Maurer 2016) along with a more scrutinized analysis.

6 Experiments

In this section, we first present the experimental results on both synthetic and real-world concept drift datasets in Sect. 6.1. Then, we provide the empirical support for the effectiveness of the model reuse mechanism in Sect. 6.2. Next, we justify the efficacy of weight update mechanism by showing the empirical studies of weight concentration phenomenon (Sect. 6.3) and experiments on recurring concept drift datasets (Sect. 6.4). Finally, we conduct the parameter study in Sect. 6.5.

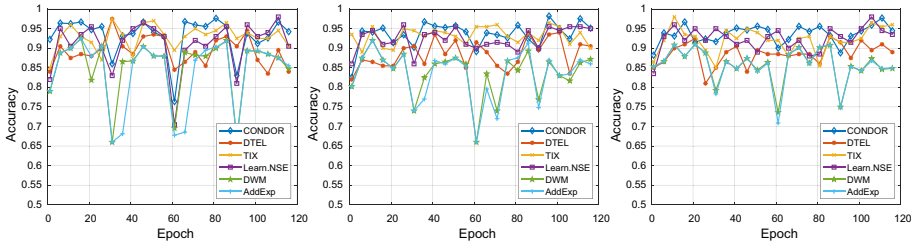
6.1 Results on synthetic and real-world datasets

Contenders We conduct the comparisons with two classes of state-of-the-art concept drift approaches. The first class is the *ensemble category*, including (a) `Learn`$^{++}$`.NSE` (Elwell

[3] We use m instead of m_k for simplicity.

Table 1 Basic statistics of datasets with concept drift

Dataset	# instance	# dim	# class	Dataset	# instance	# dim	# class
SEA200A	24,000	3	2	GEARS-2C-2D	200,000	2	2
SEA200G	24,000	3	2	Usenet-1	1500	100	2
SEA500G	60,000	3	2	Usenet-2	1500	100	2
CIR500G	60,000	3	2	Luxembourg	1900	32	2
SINE500G	60,000	2	2	Spam	9324	500	2
STA500G	60,000	3	2	Email	1500	913	2
1CDT	16,000	2	2	Weather	18,159	8	2
1CHT	16,000	2	2	GasSensor	4450	129	6
UG-2C-2D	100,000	2	2	Powersupply	29,928	2	2
UG-2C-3D	200,000	3	2	Electricity	45,312	8	2
UG-2C-5D	200,000	5	2	Covertype	581,012	54	2



(a) SEA200A **(b)** SEA200G **(c)** SEA500G

Fig. 2 Holdout accuracy comparisons on three synthetic datasets

and Polikar 2011), (b) `DWM` (Kolter and Maloof 2003, 2007) and (c) `AddExp` (Kolter and Maloof 2005), (d) `HybridForest` (Rad and Haeri 2019). The second class is the *transfer category*, including (e) `DTEL` (Sun et al. 2018) and (f) `TIX` (Forman 2006). Essentially, DTEL and TIX also adopt ensemble idea, we classify them into transfer category just to highlight their model transfer strategies.

Settings In the experiments, we set the maximum update period (epoch size) $p = 50$ and model pool size $K = 25$.[4] In case of the abrupt changes, we choose ADWIN algorithm (Bifet and Gavaldà 2007) as the drift detector $\mathfrak{D}$ with default parameter setting reported in their paper.

Synthetic Datasets As it is not realistic to foreknow detailed concept drift information of real-world datasets, like the start, the end of change and so on. We employ six widely used synthetic datasets SEA, CIR, SIN, and STA with their variants into experiments. Besides, another six synthetic datasets are also adopted: 1CDT, 1CHT, UG-2C-2D, UG-2C-3D, UG-2C-5D, and GEARS-2C-2D. Table 1 summarizes their brief statistics. We provide detailed dataset information in "Appendix E.1".

We plot holdout accuracy comparisons over three synthetic datasets: SEA200A, SEA200G, and SEA500G. The holdout accuracy is calculated over testing data generated

[4] Except for Covertype and GasSensor datasets, we set $p = 200$, since Covertype is extremely large with 581,012 data items in total and GasSensor is a multi-class dataset causing a higher sample complexity.

according to the identical distribution as training data at each time stamp. Following (Sun et al. 2018), we manually split the time horizon of each dataset into 120 epochs to have a clear presentation, and all the algorithms will perform the model update after each epoch ends. Note that each synthetic dataset has four stages, in other words, the distribution changes for three times. Specifically, the decision boundary changes for every 30 epochs. From Fig. 2, we can see that all the approaches drop when an abrupt concept drift occurs. Nevertheless, Condor is relatively stable and rises rapidly with more new data items coming, and finally achieves the highest accuracy compared with other approaches, which validates its effectiveness.

Real-world Datasets We adopt ten real-world datasets: Usenet-1, Usenet-2, Luxembourg, Spam, Email, Weather, GasSensor, Powersupply, Electricity, and Covertype. The number of data items varies from 1500 to 581,012, and the class number varies from 2 to 6. Detailed descriptions are provided in "Appendix 1". We conduct all the experiments for five trails and report overall mean and standard deviation of *predictive accuracy* in Table 2, which is the ratio between the number of iterations with correct predictions and the time horizon T. The measure reflects the average performance of the algorithm over the whole data stream. Apart from all the real-world datasets, synthetic datasets are also included.

Table 2 shows that Condor outperforms other contenders. It achieves the best on 16 over 22 datasets in total and ranks the second on four other datasets. Especially, Condor behaves significantly better than other approaches in all ten real-world datasets. The reason that Condor behaves poor on two synthetic datasets (CIR500G and SIN500G) is that these two datasets are highly nonlinear (generated by a circle and sine function respectively). This problem can be solved by using the non-linear mapping. For example, we adopt the random feature technique to linearize these two datasets, and the results show the accuracy is improved from 68.41 ± 0.87 to 85.14 ± 0.08 for CIR500G, and from 65.68 ± 0.12 to 73.59 ± 0.43 for SIN500G. These show the superiority of our proposed approach.

From the win/tie/loss summarized in the last row of Table 2, we can see that DWM is the most competitive contender. We now discuss its complexity. In each epoch, its space complexity is $O(d(K + p))$ and the time complexity is $O(dp^2 + dpK)$. We remark that there is typically a limited number of data in each epoch, so the complexity of DWM is basically comparable with that of Condor.

6.2 Effectiveness of model reuse mechanism

In the theoretical analysis presented in the last sections, we demonstrate that the WeightUpdate mechanism leads to the phenomenon of weight concentration, highly useful for adaptively reusing the previous model according to their reusability, and thus provides a good combination of previous models. Theorem 1 and Theorem 4 further show that ModelReuse mechanism is capable of utilizing such an initialization, in both binary and multi-class classification. We now validate the effectiveness via empirical studies.

Figure 3 shows the performance comparison between Condor and Condor without model reuse (that is, directly training a new model without reusing previous models). We can observe that the model reuse mechanism does help, especially on these "difficult" datasets where Condor without model reuse only achieves around 60% accuracy, whereas the model reuse mechanism can bring at least ten percents accuracy improvement.

As will be shown later, the default epoch size p will be set as 50. That is, in each epoch, there will be at most 50 data items for training. Therefore, it would be rather undesired if one train a new model with such a limited number of training data directly. Theorem 1 and

Table 2 Performance comparisons on synthetic and real-world datasets

Dataset	Ensemble category				Transfer category		Ours
	Learn^{++}.NSE	DWM	AddExp	HybridForest	DTEL	TIX	Condor
SEA200A	84.48 ± 0.19 •	86.07 ± 0.30 •	84.35 ± 0.86 •	80.80 ± 0.54 •	80.50 ± 0.58 •	82.79 ± 0.27 •	**86.42 ± 0.17**
SEA200G	85.48 ± 0.33 •	86.92 ± 0.13 •	85.54 ± 0.69 •	82.37 ± 0.35 •	80.73 ± 0.19 •	82.95 ± 0.12 •	**86.97 ± 0.16**
SEA500G	86.03 ± 0.19 •	87.63 ± 0.06 •	87.14 ± 0.12 •	83.79 ± 0.25 •	80.42 ± 0.24 •	83.26 ± 0.07 •	**87.70 ± 0.05**
CIR500G	84.77 ± 0.56 ○	77.09 ± 0.71 ○	76.48 ± 0.81 ○	84.02 ± 0.19 ○	79.03 ± 0.34 ○	66.38 ± 0.85 •	68.41 ± 0.87
SIN500G	79.41 ± 0.07 ○	66.99 ± 0.10 ○	66.81 ± 0.12 ○	78.78 ± 0.29 ○	74.93 ± 0.34 ○	62.73 ± 0.14 •	65.68 ± 0.12
STA500G	83.97 ± 0.13 •	87.43 ± 0.18 •	86.89 ± 0.27 •	71.72 ± 0.43 •	88.26 ± 0.18 •	85.95 ± 0.07 •	**88.60 ± 0.07**
1CDT	99.77 ± 0.14 •	99.92 ± 0.10	99.92 ± 0.10	99.64 ± 0.10 •	99.69 ± 0.11 •	99.56 ± 0.08 •	**99.95 ± 0.04**
1CHT	99.69 ± 0.20	99.71 ± 0.28	99.56 ± 0.46	99.51 ± 0.19 •	92.08 ± 0.22 •	99.41 ± 0.22 •	**99.86 ± 0.13**
UG-2C-2D	94.42 ± 0.12 •	95.60 ± 0.12 ○	94.36 ± 0.78 •	92.67 ± 0.63 •	93.98 ± 0.13 •	94.69 ± 0.13 •	95.27 ± 0.09
UG-2C-3D	93.82 ± 0.60 •	95.23 ± 0.59	94.61 ± 0.73 •	92.59 ± 0.84 •	92.94 ± 0.72 •	94.31 ± 0.69 •	94.84 ± 0.59
UG-2C-5D	90.30 ± 0.30 •	92.85 ± 0.23 ○	92.20 ± 0.23 ○	87.73 ± 0.54 •	88.21 ± 0.35 •	89.84 ± 0.38 •	**91.83 ± 0.24**
GEARS-2C-2D	95.82 ± 0.02 •	95.83 ± 0.02 •	95.83 ± 0.02 •	95.97 ± 0.10	94.96 ± 0.03 •	95.03 ± 0.02 •	**95.91 ± 0.01**
Usenet-1	63.76 ± 2.01 •	67.26 ± 3.11 •	62.11 ± 2.67 •	60.19 ± 1.94 •	68.02 ± 1.19 •	65.03 ± 1.70 •	**73.13 ± 1.12**
Usenet-2	72.42 ± 1.14 •	68.41 ± 1.17 •	70.55 ± 2.41 •	66.57 ± 0.29 •	72.02 ± 0.87 •	70.56 ± 1.15 •	**75.13 ± 1.06**
Luxembourg	98.64 ± 0.00 •	90.42 ± 0.55 •	90.77 ± 0.52 •	97.89 ± 0.38 •	100.0 ± 0.00	90.99 ± 0.97 •	99.98 ± 0.03
Spam	90.79 ± 0.85 •	92.18 ± 0.34 •	91.78 ± 0.33 •	88.27 ± 0.76 •	85.53 ± 1.22 •	87.10 ± 1.45 •	**95.22 ± 0.48**
Email	74.21 ± 4.61 •	72.58 ± 4.10 •	60.78 ± 6.12 •	96.52 ± 2.31 ○	83.36 ± 1.87 •	79.83 ± 3.73 •	**91.60 ± 1.86**
Weather	75.99 ± 0.36 •	70.83 ± 0.49 •	70.07 ± 0.34 •	74.71 ± 0.77 •	68.92 ± 0.27 •	70.21 ± 0.33 •	**79.37 ± 0.26**
GasSensor	42.36 ± 3.72 •	76.61 ± 0.36 •	76.61 ± 0.36 •	86.10 ± 1.50 ○	63.82 ± 3.64 •	43.40 ± 2.88 •	**81.57 ± 3.77**
Powersupply	74.06 ± 0.28 ○	72.09 ± 0.29 •	72.13 ± 0.23 •	72.37 ± 0.19 •	69.90 ± 0.38 •	68.34 ± 0.16 •	72.82 ± 0.29
Electricity	78.97 ± 0.18 •	78.03 ± 0.17 •	75.62 ± 0.42 •	78.51 ± 0.49 •	81.05 ± 0.35 •	58.44 ± 0.71 •	**84.73 ± 0.33**
Covertype	79.08 ± 1.30 •	74.17 ± 0.87 •	73.13 ± 1.53 •	84.74 ± 1.07 •	69.43 ± 1.30 •	64.60 ± 0.89 •	**89.58 ± 0.14**
Condor w/t/l	18/ 1/ 3	14/ 3/ 4	17/ 2/ 3	17/ 1/ 4	19/ 1/ 2	22/ 0/ 0	Rank first 16/ 22

On each dataset, five test runs were conducted and the average performance as well as standard deviation are presented. Besides, the best accuracy on each data set is bolded, and • (○) indicates our approach Condor is significantly better (worse) than compared approaches (paired t-tests at 95% significance level)

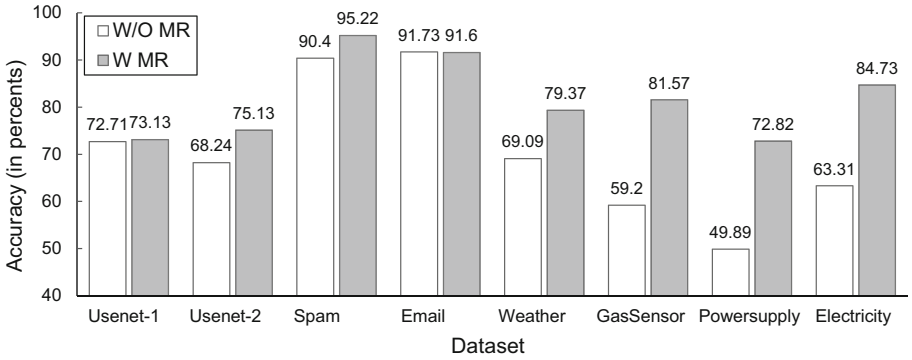


Fig. 3 Performance comparisons (in predictive accuracy) of Condor with/without model reuse mechanism

Table 3 *Emailing list* dataset: There are in total three different topics, "+" indicates the users are interested in it, while "−" indicates not interested in it

Topic	1–300	301–600	601–900	901–1200	1201–1500
Medicine	+	−	+	−	+
Space	−	+	−	+	−
Baseball	−	+	−	+	−

Say an example, in the first period (time stamp 0–300), users' are only interested in messages of the topic medicine

Theorem 4 reveal that the model reuse mechanism can reduce the sample complexity when provided with a proper initialization. The results accord to the theoretical insight.

6.3 Effectiveness of weight concentration mechanism

We examine the weight concentration phenomenon on a real-world dataset *Emailing list*, whose detailed information of concept drift is shown in Table 3. We can see that the concept drift happens for every 300 rounds, in a recurring manner.

Let us consider the epoch in 1200–1500 (epoch 5), and focus on the weights of previous models h_1, h_2, h_3, h_4, namely, $\beta_1, \beta_2, \beta_3, \beta_4$. Clearly, the concept of epoch 5 has emerged previously, i.e., it is the same as the concepts of epoch 1 and epoch 3. Thus, we expect that the weight distribution should concentrate on β_1 and β_4. As we can see in Table 4, the empirical results show the weights β_1 and β_4 indeed dominate while β_2 and β_3 are almost zero. The result validates that the returned weight distribution largely concentrates on those previous models who better fit current data epoch. Additionally, the results, to some extent, justify why our approach succeeds in recurring concept drift scenarios. A more detailed elaboration will be presented in the next paragraph.

6.4 Recurring concept drift

In this paragraph, we conduct the performance comparison on the *recurring concept drift* scenario, a specific sub-type of concept drift, in which previous concepts may disappear and then re-appear in the future. Therefore, previous models may be beneficial for future learning.

Table 4 Demonstration of weights concentration phenomenon on Emailing list dataset

Iteration	1	2	3	4	5	...	296	297	298	299	300
β_1	0.25	0.299	0.345	0.384	0.416		0.5	0.5	0.5	0.5	0.5
β_2	0.25	0.201	0.155	0.116	0.084		1.9E−52	1.3E−52	8.5E−53	5.7E−53	3.8E−53
β_3	0.25	0.299	0.345	0.384	0.416		0.5	0.5	0.5	0.5	0.5
β_4	0.25	0.201	0.155	0.116	0.084		1.9E−52	1.3E−52	8.5E−53	5.7E−53	3.8E−53

Consider the epoch in 1201–1500, and following reports weights associated with previous models h_1, h_2, h_3, h_4

Table 5 Performance comparisons on two recurring concept drift datasets: Email list and Spam filtering

Category	Approach	Email list			Spam filtering		
		Accuracy	Precision	Recall	Accuracy	Precision	Recall
Window	SVM-fix	71.4	73.7	72.1	88.1	82.0	68.5
	NB-sw	74.7	77.9	73.2	91.9	90.2	77.0
Ensemble	Learn^{++}.NSE	70.0	76.5	76.5	90.4	84.5	79.6
	DWM	78.2	75.1	81.4	91.9	89.1	84.0
	AddExp	70.4	68.2	71.4	91.3	90.0	80.7
Model-reuse	TIX	86.2	88.2	88.2	88.5	82.3	69.3
	DTEL	86.2	88.2	88.2	86.3	73.4	71.4
Recurring	CCP	77.5	79.7	77.6	92.3	85.7	83.9
	DACC	76.2	73.8	75.9	94.7	95.1	**97.8**
	ADACC	77.5	75.2	77.2	94.9	**95.6**	97.6
Ours	Condor	**95.6**	**93.2**	**99.8**	**95.4**	91.1	90.8

The best accuracy on each data set is bolded

Previous studies show that one needs to consider the recurring structure specifically, otherwise the performance will dramatically drop, even for approaches dealing with gradually evolving concept drift.

Datasets We adopt two popular real-world datasets with recurring concept drift, i.e., *Email list* and *Spam filtering* datasets (Katakis et al. 2008, 2010; Jaber et al. 2013). Both datasets are extracted from email corpus, and concept is decided by users' personal interests, which changes in a recurring manner.

Comparisons We include contenders are from the following four categories: (a) Sliding window based approaches, including SVM-fix (batch implementation by SVM) and NB-sw (update only use data in the sliding window based on incremental Naive Bayes). (b) Ensemble based approaches, including Learn^{++}.NSE, DWM and AddExp. (c) Transfer learning based approaches, including TIX and DTEL. (d) Recurring approaches, which are specifically designed for recurring concept drift scenarios, including Conceptual Clustering and Prediction (CCP) approach, an ensemble method that handle recurring concept drift via similarity clustering (Katakis et al. 2010). Also, we compare with Dynamic Adaptation to Concept Change (DACC) and its adaptive variant ADACC, detecting recurring concept drift based on a new second-order online learning mechanism (Jaber et al. 2013). Since codes of CCP, DACC and ACACC are not available, we directly use the results reported in their papers, as we use the whole dataset without any random splitting according to their settings.

Results Table 5 reports experimental results. We can see that Condor exhibits an encouraging performance on both datasets over three different performance measures. It performs significantly better than general concept drift approaches and is comparable or even better than those approaches designed explicitly for recurring concept drift scenario.

The effectiveness of Condor in recurring datasets lies in the effect of *weight concentration*, since our approach guarantees the weight concentrates on the best-fit previous models (See Observation 1).

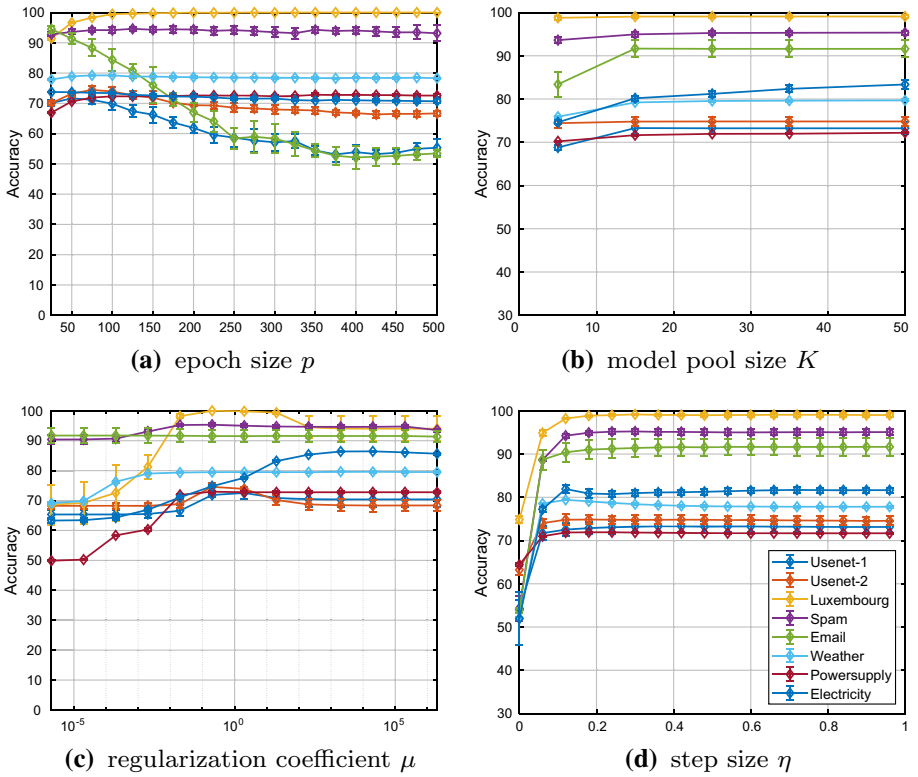


Fig. 4 Parameter study on different real-world datasets

6.5 Parameter study

In this part, we study the parameter influence. There are four parameters: epoch size p, model pool size K, regularization coefficient μ and step size η. We conduct the experiments for five times and plot the mean and standard variance of predictive accuracy with respect to different parameter values in Fig. 4.

Epoch Size We set the value of epoch size (i.e., maximum update period) p from 25 to 500. Figure 4a shows that the overall performance is relatively stable in terms of different epoch sizes. However, for some highly non-stationary datasets (particularly, email and electricity), the accuracy curve drops significantly with a larger epoch size, because the epoch size is so large that data within are not iid and thus the model trained from scratch is not desired. On the other hand, although a small epoch size leads to a more timely update, the model trained from each epoch will suffer from the insufficient data. Actually, the epoch size is a data-dependent parameter, reflecting the inherent extent of fluctuation. Meanwhile, the setting of epoch size might also subject to requirements from real-world applications. In this paper, we set the default value of epoch size p as 50.

Model Pool Size We set the value of model pool size K from 5 to 50. Figure 4b shows that the predictive accuracy rises as the model pool size increases, and will not benefit from an even larger model pool. Although an even larger K might be benign for performance improvement,

the memory cost will also significantly increase with a larger K. We set the default value of model pool size K as 25.

Regularization Coefficient We set the value of regularization coefficient μ from 2×10^{-6} to 2×10^{6}. Figure 4c shows that when we set a relatively large μ value, all datasets basically achieve the best performance, and are not sensitive to the μ value. This agrees with the intuition since μ value represents a trade-off between empirical loss and biased regularization, a larger value addresses more importance on reusing models. In other words, when μ is large, it tends to exploit more information from previous models to build the new model. Hence, the result implies the effectiveness of model reuse. We set the default value of the regularization coefficient μ as 200.

Step Size We set the value of step size η from 0 to 1. From Fig. 4d, we can see that when step size is relatively large, say larger than 0.5, the performance is satisfying and stable. This phenomenon matches the theoretical suggestion value in Theorem 2, calculated by $\eta_{\texttt{theory}} = \sqrt{8\ln(K)/p} = \sqrt{8\ln(25)/50} \approx 0.718$, where the model size K is 25 and epoch size p is 50 by default. We set the default value of step size η as 0.75.

7 Conclusion

In this paper, a novel and effective approach Condor is proposed for handling concept drift via model reuse, which consists of two key modules, `ModelUpdate` and `WeightUpdate`. Specifically, ModelUpdate mechanism reuses previous models in a weighted manner to train a new model along with current data. Meanwhile, WeightUpdate mechanism adaptively adjusts weights of previous models according to their performance. By the generalization analysis, we prove that the model reuse mechanism helps if we properly reuse previous models. Through regret analysis, we show that the weights finally concentrate on those better-fit models and thus provides a good weighted combination of previous models as the initialization for ModelUpdate mechanism. Moreover, our approach enjoys an $O(T^{2/3}V_T^{1/3})$ dynamic regret, where T is the length, and V_T is the function variation, representing the non-stationarity of the data stream. Empirical results demonstrate the superiority of our approach to other compared methods, on both synthetic and real-world datasets.

In the future, it would be interesting to incorporate more techniques from model reuse learning into handling concept drift problems. Interesting future issue is to incorporate a condor-like approach into the recently proposed *abductive learning* (Zhou 2019), a new paradigm which leverages both machine learning and logical reasoning, to enable it handle changing concepts and predicates.

Acknowledgements This research was supported by the National Key R&D Program of China (2018YFB1004300), NSFC (61921006), and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

Appendix A: Prerequisite knowledge and technical lemmas

In this section, we introduce prerequisite knowledge and technical lemmas in order to prove the main results. Specifically, we will utilize Rademacher complexity (Bartlett and Mendelson 2002) in proving generalization bounds. Besides, we will also exploit the function properties when bounding Rademacher complexity and proving the regret bounds.

Appendix A.1: Rademacher complexity

To simplify the presentation, first, we introduce some notations. Let $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ be a sample of m points drawn i.i.d. according to the underlying distribution $\mathcal{D}$, then the risk and empirical risk of hypothesis h are defined by

$$R(h) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\ell(h(\mathbf{x}), y)\right], \quad \hat{R}_S(h) = \frac{1}{m}\sum_{i=1}^{m}\ell(h(\mathbf{x}_i), y_i).$$

In the following, we will utilize the notion of Rademacher complexity (Bartlett and Mendelson 2002) to measure the hypothesis complexity and use it to bound the generalization error.

Definition 2 (*Rademacher Complexity* Bartlett and Mendelson 2002) Let $\mathcal{G}$ be a family of functions and a fixed sample of size m as $S = (\mathbf{z}_1, \cdots, \mathbf{z}_m)$. Then, the *empirical Rademacher complexity* of $\mathcal{G}$ with respect to the sample S is defined as:

$$\hat{\mathfrak{R}}_S(\mathcal{G}) = \mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{g\in\mathcal{G}}\frac{1}{m}\sum_{i=1}^{m}\sigma_i\, g(\mathbf{z}_i)\right].$$

Besides, the *Rademacher complexity* of $\mathcal{G}$ is the expectation of the empirical Rademacher complexity over all samples of size m drawn according to $\mathcal{D}$:

$$\mathfrak{R}_m(\mathcal{G}) = \mathbb{E}_{S\sim\mathcal{D}^m}[\hat{\mathfrak{R}}_S(\mathcal{G})]. \tag{11}$$

Appendix A.2: Function properties

In this paragraph, we introduce several common and useful function properties.

Definition 3 (*Lipschitz Continuity*) A function $f : \mathcal{K} \mapsto \mathbb{R}$ is L-Lipschitz continuous w.r.t. a norm $\|\cdot\|$ over domain $\mathcal{K}$ if for all $\mathbf{x}, \mathbf{y} \in \mathcal{K}$, we have

$$|f(\mathbf{y}) - f(\mathbf{x})| \leq L\|\mathbf{y} - \mathbf{x}\|.$$

Definition 4 (*Strong Convexity*) A function $f : \mathcal{K} \mapsto \mathbb{R}$ is λ-strongly convex w.r.t. a norm $\|\cdot\|$ if for all $\mathbf{x}, \mathbf{y} \in \mathcal{K}$ and for any $\alpha \in [0, 1]$, we have

$$f((1-\alpha)\mathbf{x} + \alpha\mathbf{y}) \leq (1-\alpha)f(\mathbf{x}) + \alpha f(\mathbf{y}) - \frac{\lambda}{2}\alpha(1-\alpha)\|\mathbf{x} - \mathbf{y}\|^2.$$

A common and equivalent form for the differentiable case is,

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^{\mathrm{T}}(\mathbf{y} - \mathbf{x}) + \frac{\lambda}{2}\|\mathbf{y} - \mathbf{x}\|^2. \tag{12}$$

Definition 5 (*Smoothness*) A function $f : \mathcal{K} \mapsto \mathbb{R}$ is σ-smooth w.r.t. a norm $\|\cdot\|$ if f is everywhere differentiable and for all $\mathbf{x}, \mathbf{y} \in \mathcal{K}$, we have

$$f(\mathbf{y}) \leq f(\mathbf{x}) + \nabla f(\mathbf{x})^{\mathrm{T}}(\mathbf{y} - \mathbf{x}) + \frac{\sigma}{2}\|\mathbf{y} - \mathbf{x}\|^2.$$

The above condition is equivalent to a Lipschitz condition over the gradients,

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq \sigma\|\mathbf{x} - \mathbf{y}\|.$$

Appendix A.3: Technical lemmas

To obtain a fast generalization rate, essentially, we need a Bernstein-type concentration inequality. And we adopt the functional generalization of Bennett's inequality due to Bousquet (2002), for self-containedness, we state the conclusion in Lemma 1 as follow.

Lemma 1 (Theorem 2.11 of Bousquet 2002) *Assume the random variables $X_1, \ldots, X_n$ are identically distributed according to P. Let $\mathcal{F}$ be a countable set of functions from $\mathcal{X}$ to $\mathbb{R}$ and assume that all functions f in $\mathcal{F}$ are P-measurable, square-integrable and satisfy $\mathbb{E}[f] = 0$. If $\sup_{f\in\mathcal{F}} \operatorname{ess\,sup} f \leq 1$ then we denote*

$$Z = \sup_{f\in\mathcal{F}} \sum_{i=1}^{n} f(X_i),$$

and if $\sup_{f\in\mathcal{F}} \|f\|_\infty \leq 1$, Z can be defined as above or as

$$Z = \sup_{f\in\mathcal{F}} \left| \sum_{i=1}^{n} f(X_i) \right|.$$

Let σ be a positive real number such that $\sigma^2 \geq \sup_{f\in\mathcal{F}} \operatorname{Var}[f(X_1)]$ almost surely, then for all $t \geq 0$, we have

$$\Pr\left[Z \geq \mathbb{E}[Z] + t\right] \leq \exp\left\{-vg\left(\frac{t}{v}\right)\right\},$$

with $v = n\sigma^2 + 2\mathbb{E}[Z]$ and $g(y) = (1+y)\log(1+y) - y$, also

$$\Pr\left[Z \geq \mathbb{E}[Z] + \sqrt{2tv} + \frac{t}{3}\right] \leq e^{-t}.$$

Besides, for a strongly convex regularizer, we have following property, which will be useful in proving Theorem 1.

Lemma 2 (Corollary 4 of Kakade et al. 2012) *If $f : \mathcal{K} \mapsto \mathbb{R}$ is λ-strongly convex w.r.t. a norm $\|\cdot\|$ and $f^\star(\boldsymbol{0}) = 0$, then, denoting the partial sum $\sum_{j\leq i} \boldsymbol{v}_j$ by $\boldsymbol{v}_{1:i}$, we have for any feasible sequence $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ and for any $\boldsymbol{u} \in \mathcal{K}$,*

$$\sum_{i=1}^{n} \langle \boldsymbol{v}_i, \boldsymbol{u}\rangle - f(\boldsymbol{u}) \leq f^\star(\boldsymbol{v}_{1:n}) \leq \sum_{i=1}^{n} \langle \nabla f^\star(\boldsymbol{v}_{1:i-1}), \boldsymbol{v}_i\rangle + \frac{1}{2\lambda}\sum_{i=1}^{n} \|\boldsymbol{v}_i\|_\star^2,$$

where $f^\star$ is the Fenchel conjugate of f, namely, $f^\star(\boldsymbol{x}) = \sup_{\boldsymbol{y}\in\mathcal{K}}\{\langle \boldsymbol{y}, \boldsymbol{x}\rangle) - f(\boldsymbol{y})\}$.

Lemma 3 (Lemma 8.1 of Mohri et al. 2012) *Let $\mathcal{F}_1, \ldots, \mathcal{F}_l$ be l hypothesis sets in $\mathbb{R}^{\mathcal{X}}$, $l \geq 1$, and let $\mathcal{F} = \{\max\{h_1, \ldots, h_l\} : h_j \in \mathcal{F}_j, j \in [1, l]\}$. Then, for any sample S of size m, the empirical Rademacher complexity of $\mathcal{F}$ can be upper bounded as follows:*

$$\hat{\mathfrak{R}}_S(\mathcal{F}) \leq \sum_{j=1}^{l} \hat{\mathfrak{R}}_S(\mathcal{F}_j).$$

Appendix B: Proof of Theorem 1

We prove the statement in Theorem 1 in the following two steps,

(1) First, in Lemma 4, we establish fast generalization error bound in terms of the Rademacher complexity of loss function family, i.e., $\mathfrak{R}_m(\mathcal{L})$.
(2) Next, in Lemma 5, we upper bound the Rademacher complexity of Lipschitz loss function family by terms regarding to R_p, which is defined as the risk of the combination of previous models h_p on current distribution $\mathcal{D}_k$.

Appendix B.1: Fast rate generalization error bound

Lemma 4 *Assume that the non-negative loss function $\ell : \mathbb{R} \times \mathcal{Y} \mapsto \mathbb{R}_+$ is bounded by $M \geq 0$. Let $\mathcal{H}$ denote the hypothesis set and S be a sample of m points drawn i.i.d. according to distribution $\mathcal{D}$. For any constant $r \geq 0$, we define the truncated hypothesis set as $\mathcal{H}_r = \{h : h \in \mathcal{H} \wedge R(h) \leq r\}$, whose associated loss function family $\mathcal{L}$ is*

$$\mathcal{L} = \{(\boldsymbol{x}, y) \mapsto \ell(h(\boldsymbol{x}, y) : h \in \mathcal{H}_r\}.$$

Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $h \in \mathcal{H}_r$ with ,

$$R(h) - \hat{R}_S(h) \leq 2\mathfrak{R}_m(\mathcal{L}) + \frac{3M \log(1/\delta)}{2m} + 3\sqrt{\frac{(8\mathfrak{R}_m(\mathcal{L}) + r)M \log(1/\delta)}{4m}}. \tag{13}$$

Proof The proof is based on the application of functional generalization of Bennett's inequality in Lemma 1.

For any sample $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ and any hypothesis $h \in \mathcal{H}_r$, we turn to consider the uniform upper bound of $R(h) - \hat{R}_S(h)$. The argument consists of applying functional generalization of Bennett's inequality, namely, Lemma 1, to function Φ_S defined by

$$\Phi_S = \frac{m}{2M} \sup_{h \in \mathcal{H}_r} \{R(h) - \hat{R}_S(h)\},$$

which satisfies the condition in Lemma 1. We now first verify this claim. Following the notations in Lemma 1, let $f(X_i) = \frac{1}{2M}\{\mathbb{E}[\ell(h(\mathbf{x}), y)] - \ell(h(\mathbf{x}_i), y_i)\}$. Evidently, on one hand, $\mathbb{E}[f] = 0$. One the other hand, $\sup_{f \in \mathcal{F}} \operatorname{ess\,sup} f = \sup_{h \in \mathcal{H}_r} \operatorname{ess\,sup} \frac{1}{2M}\{\mathbb{E}[\ell(h(\mathbf{x}), y)] - \ell(h(\mathbf{x}_i), y_i)\} \leq 1$, due to the boundedness of loss function. Therefore, the quantity Φ_S essentially represents Z in Lemma 1, which can be verified as follows,

$$\begin{aligned} Z &= \sup_{f \in \mathcal{F}} \sum_{i=1}^{m} f(X_i) = \sup_{h \in H_r} \sum_{i=1}^{m} \frac{1}{2M}\{\mathbb{E}[\ell(h(\mathbf{x}), y)] - \ell(h(\mathbf{x}_i), y_i)\} \\ &= \frac{m}{2M} \sup_{h \in H_r} R(h) - \hat{R}_S(h) = \Phi_S. \end{aligned}$$

Therefore, we can now apply Lemma 1, for all $t \geq 0$, the following holds

$$\Pr[\Phi_S \geq \mathbb{E}_S[\Phi_S] + t] \leq \exp\left\{-vg\left(\frac{t}{v}\right)\right\},$$

where $g(y) = (1 + y)\log(1 + y) - y$ and

$$v = m\sigma^2 + 2\mathbb{E}_S[\Phi_S], \quad \sigma^2 = \sup_{h \in \mathcal{H}_r} \operatorname{Var}\left[\frac{1}{2M}\mathbb{E}_{(\mathbf{x}', y) \sim \mathcal{D}}[\ell(h(\mathbf{x}'), y')] - \ell(h(\mathbf{x}), y)\right].$$

And we can reverse the above inequality and obtain that for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $h \in \mathcal{H}_r$,

$$\Phi_S \leq \mathbb{E}_S[\Phi_S] + \frac{3\log(1/\delta)}{4} + \frac{3}{2}\sqrt{v\log(1/\delta)}. \tag{14}$$

Thus, we proceed to bound term σ^2 and $\mathbb{E}_S[\Phi_S]$. First, we bound the term σ^2.

$$\begin{aligned}
\sigma^2 &= \sup_{h\in\mathcal{H}_r} \mathrm{Var}\left[\frac{1}{2M}\mathbb{E}_{(\mathbf{x}',y)\sim\mathcal{D}}[\ell(h(\mathbf{x}'),y')] - \ell(h(\mathbf{x}),y)\right] \\
&= \sup_{h\in\mathcal{H}_r} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\frac{1}{4M^2}\left(\ell(h(\mathbf{x}),y) - \mathbb{E}_{(\mathbf{x}',y')\sim\mathcal{D}}[\ell(h(\mathbf{x}'),y')]\right)^2\right] \\
&\leq \sup_{h\in\mathcal{H}_r} \frac{1}{4M^2}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\ell(h(\mathbf{x}),y)^2] \\
&\leq \sup_{h\in\mathcal{H}_r} \frac{1}{4M}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\ell(h(\mathbf{x}),y)] \qquad (15) \\
&= \sup_{h\in\mathcal{H}_r} \frac{1}{4M}R(h) = \frac{r}{4M}. \qquad (16)
\end{aligned}$$

Equation (15) holds due to the boundedness of loss function, and Eq. (16) holds because of the definition of H_r, as we know that $R(h) \leq r$ holds for all $h \in \mathcal{H}_r$.

Next, we turn to prove the bound on $\mathbb{E}_S[\Phi_S]$ by utilizing the standard *symmetrization* technique (Koltchinskii 2011; Mohri et al. 2012).

$$\begin{aligned}
\mathbb{E}_S[\Phi_S] &= \mathbb{E}_S\left[\frac{m}{2M}\sup_{h\in\mathcal{H}_r}\{R(h) - \hat{R}_S(h)\}\right] \\
&= \mathbb{E}_S\left[\frac{m}{2M}\sup_{h\in\mathcal{H}_r}\left\{\mathbb{E}_{S'}[\hat{R}_{S'}(h) - \hat{R}_S(h)]\right\}\right] \\
&\leq \frac{m}{2M}\mathbb{E}_{S,S'}\left[\sup_{h\in\mathcal{H}_r}\left\{\hat{R}_{S'}(h) - \hat{R}_S(h)\right\}\right] \qquad (17) \\
&= \frac{m}{2M}\mathbb{E}_{S,S'}\left[\sup_{h\in\mathcal{H}_r}\frac{1}{m}\sum_{i=1}^{m}\left(\ell(h(\mathbf{x}_i'),y_i') - \ell(h(\mathbf{x}_i),y_i)\right)\right] \\
&= \frac{m}{2M}\mathbb{E}_{\boldsymbol{\sigma},S,S'}\left[\sup_{h\in\mathcal{H}_r}\frac{1}{m}\sum_{i=1}^{m}\sigma_i\left(\ell(h(\mathbf{x}_i'),y_i') - \ell(h(\mathbf{x}_i),y_i)\right)\right] \qquad (18) \\
&\leq \frac{m}{M}\mathbb{E}_{\boldsymbol{\sigma},S}\left[\sup_{h\in\mathcal{H}_r}\frac{1}{m}\sum_{i=1}^{m}\sigma_i\left(\ell(h(\mathbf{x}_i),y_i)\right)\right] = \frac{m}{M}\mathfrak{R}_m(\mathcal{L}). \qquad (19)
\end{aligned}$$

Equation (17) holds due to the convexity of supremum function, and thus we can apply Jensen's inequality. In Eq. (18), we introduce Rademacher random variables σ_is, that are uniformly distributed independent random variables taking values in $\{-1, +1\}$, and thus this does not change the expectation. Equation (19) holds due to the sub-additivity of supremum function.

Thus, we can obtain the bound on v,

$$v = m\sigma^2 + 2\mathbb{E}_S[\Phi_S] \leq \frac{rm}{4M} + \frac{2m}{M}\mathfrak{R}_m(\mathcal{L}).$$

Combining the bounds on v and $\mathbb{E}_S[\Phi_S]$, we have

$$\Phi_S = \frac{m}{2M} \sup_{h\in\mathcal{H}_r} \{R(h) - \hat{R}_S(h)\} \le \mathbb{E}_S[\Phi_S] + \frac{3\log(1/\delta)}{4} + \frac{3}{2}\sqrt{v\log(1/\delta)}$$
$$\le \frac{m}{M}\mathfrak{R}_m(\mathcal{L}) + \frac{3\log(1/\delta)}{4} + \frac{3}{2}\sqrt{\left(\frac{rm}{4M} + \frac{2m}{M}\mathfrak{R}_m(\mathcal{L})\right)\log(1/\delta)}.$$

Hence, we conclude that for any hypothesis $h \in \mathcal{H}_r$,

$$R(h) - \hat{R}_S(h) \le \sup_{h\in\mathcal{H}_r} \{R(h) - \hat{R}_S(h)\} = (2M/m)\Phi_S$$
$$\le 2\mathfrak{R}_m(\mathcal{L}) + \frac{3M\log(1/\delta)}{2m} + 3\sqrt{\frac{(r + 8\mathfrak{R}_m(\mathcal{L}))M\log(1/\delta)}{4m}}.$$

□

Appendix B.2: Rademacher complexity of Lipschitz loss function family

Lemma 5 *Assume that the non-negative loss function $\ell : \mathbb{R}\times\mathcal{Y} \mapsto \mathbb{R}_+$ is bounded by $M \ge 0$. Meanwhile, for all $y \in \mathcal{Y}$, $\ell(\cdot, y)$ is L-Lipschitz continuous. Meanwhile, assume that the regularizer $\Omega : \mathcal{H}\times\mathcal{H} \mapsto \mathbb{R}_+$ is λ-strongly convex in its first argument w.r.t. the norm $\|\cdot\|$. Given the previous model h_p (aka, $\boldsymbol{w}_p$), which be a linear weighted combination of previous models. Define the truncated hypothesis set $\mathcal{W}_r = \{\boldsymbol{w} : \Omega(\boldsymbol{w}, \boldsymbol{w}_p) \le \alpha\hat{R}_S(h_p)\}$, and let the loss function family $\mathcal{L}$ be as,*

$$\mathcal{L} = \{(\boldsymbol{x}, y) \mapsto \ell(\langle \boldsymbol{w}, \boldsymbol{x}\rangle, y) : \boldsymbol{w} \in \mathcal{W}_r\},$$

and let S be a sample of m points drawn i.i.d. according to distribution $\mathcal{D}$, then we have

$$\mathfrak{R}_m(\mathcal{L}) \le L\sqrt{\frac{2B^2\alpha R_p}{\lambda m}}. \tag{20}$$

where $B = \sup_{\boldsymbol{x}\in\mathcal{X}}\|\boldsymbol{x}\|_\star$, and $R_p = R(h_p) = \mathbb{E}_{(\boldsymbol{x},y)\sim S_k}[\ell(h_p(\boldsymbol{x}), y)]$, representing the risk of reusing model on current distribution.

Proof The empirical Rademacher complexity of $\mathcal{L}$, by definition, is,

$$\hat{\mathfrak{R}}_S(\mathcal{L}) = \frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{\mathbf{w}\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i\ell\left(\langle\mathbf{w}, \mathbf{x}_i\rangle, y_i\right)\right] \le \frac{L}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{\mathbf{w}\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i\langle\mathbf{w}, \mathbf{x}_i\rangle\right], \tag{21}$$

where the last inequality holds due to the fact that for all $y \in \mathcal{Y}$, the loss function $\ell(\cdot, y)$ is L-Lipschitz continuous, and thus we can apply the *Talagrand's Comparison Inequality* (Ledoux and Talagrand 2013) to relate the Rademacher complexity of loss function family with that of hypothesis set.

Now, we turn to bound the empirical Rademacher complexity term in the r.h.s. of Eq. (21). Similar to the technique developed in the work of Kakade et al. (2012), we utilize the primal-dual technique by exploiting the strongly-convex property of the regularizer Ω. Since the Ω does not satisfy the condition of $\Omega(\mathbf{0}, \mathbf{w}_p) = 0$, we introduce the shifted regularizer $\tilde{\Omega}(\mathbf{w}) = \Omega(\mathbf{w} + \mathbf{w}_p, \mathbf{w}_p)$, which is also λ-strongly convex in $\mathbf{w}$ with respect to the norm $\|\cdot\|$ and satisfies $\tilde{\Omega}(\mathbf{0}) = 0$ at the same time. Besides, we also introduce the non-negative variable t in order to obtain a tighter bound by tunning the variable.

For the the r.h.s. of Eq. (21), we apply Lemma 2 with $\mathbf{u} = \mathbf{w}$ and $\mathbf{v}_i = t\sigma_i \mathbf{x}_i$ ($t > 0$),

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{\mathbf{w}\in\mathcal{W}} \frac{1}{m}\sum_{i=1}^{m}\langle \mathbf{w}, t\sigma_i\mathbf{x}_i\rangle\right] \\
&\leq \frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\frac{t^2}{2\lambda}\sum_{i=1}^{m}\|\sigma_i\mathbf{x}_i\|_\star^2 + \sup_{\mathbf{w}\in\mathcal{W}}\tilde{\Omega}(\mathbf{w}) + \sum_{i=1}^{m} t\langle \nabla\tilde{\Omega}^\star(\mathbf{v}_{1:i-1}), \sigma_i\mathbf{x}_i\rangle\right] \\
&\leq \frac{B^2t^2}{2\lambda} + \frac{\alpha\hat{R}_S(h_p)}{m},
\end{aligned}$$

where the last inequality holds due to the fact that $\sup_{\mathbf{w}\in\mathcal{W}_r}\Omega(\mathbf{w}, \mathbf{w}_p) \leq \alpha\hat{R}_S(h_p)$ and $\sup_{\mathbf{x}\in\mathcal{X}}\|\mathbf{x}\|_\star \leq B$. Meanwhile, the last term vanishes due to the expectation over the Rademacher variables $\boldsymbol{\sigma}$. Furthermore, by dividing the non-negative t on both sides and noticing that the above upper bound holds for any $t > 0$, we choose a particular t_0 making the upper bound tight,

$$\frac{L}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{\mathbf{w}\in\mathcal{W}}\sum_{i=1}^{m}\sigma_i\langle \mathbf{w}, \mathbf{x}_i\rangle\right] \leq \inf_{t_1>0} L\left(\frac{B^2t_1}{2\lambda} + \frac{\alpha\hat{R}_S(h_p)}{mt_1}\right) = L\sqrt{\frac{2B^2\alpha\hat{R}_S(h_p)}{\lambda m}}. \tag{22}$$

Therefore, we obtain the bound for $\hat{\mathfrak{R}}_S(\mathcal{L})$. Notice that the square-root function is concave, by applying the Jensen's inequality w.r.t. the both side, we have the following result for the expected Rademacher complexity term,

$$\mathfrak{R}_m(\mathcal{L}) \leq L\mathbb{E}_S\left[\sqrt{2B^2\alpha\hat{R}_S(h_p)/\lambda m}\right] \leq L\sqrt{2B^2\alpha\mathbb{E}_S\left[\hat{R}_S(h_p)\right]/\lambda m} = L\sqrt{\frac{2B^2\alpha R_p}{\lambda m}},$$

where R_p is short for $R(h_p)$. Hence, we complete the proof of the statement. □

Appendix B.3: Proof of Theorem 1

Proof To prove the generalization bound for the learned model $\hat{\mathbf{w}}$ stated in Theorem 1, we turn to show that the statement holds uniformly over the following truncated hypothesis set

$$\mathcal{H}_r = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \ \Omega(\mathbf{w}, \mathbf{w}_p) \leq \frac{1}{\mu}\hat{R}(h_p) \ \wedge \ \hat{R}_S(\mathbf{w}) \leq \hat{R}_S(h_p)\}.$$

First, we demonstrate that the final model $\hat{w}$ returned by the ModelUpdate procedure belongs to the set $\mathcal{H}_r$. Actually, since $\hat{\mathbf{w}}$ is the empirical minimizer of the biased regularization objective function, thus, it is evidently better than the choice $\hat{\mathbf{w}}_p$,

$$\frac{1}{m}\sum_{i=1}^{m}\ell(\hat{\mathbf{w}}^{\mathrm{T}}\mathbf{x}_i, y_i) + \mu\Omega(\hat{\mathbf{w}}, \mathbf{w}_p) \leq \frac{1}{m}\sum_{i=1}^{m}\ell(\mathbf{w}_p^{\mathrm{T}}\mathbf{x}_i, y_i) + \mu\Omega(\mathbf{w}_p, \mathbf{w}_p) = \hat{R}_S(h_p),$$

combining the non-negative property of loss and regularizer, we know that $\hat{\mathbf{w}}$ satisfies $\hat{R}_S(\hat{\mathbf{w}}) \leq \hat{R}_S(h_p)$ and $\Omega(\hat{\mathbf{w}}, \mathbf{w}_p) \leq \frac{1}{\mu}\hat{R}_S(h_p)$. Therefore, $\hat{\mathbf{w}}$ belongs to the hypothesis set $\mathcal{H}_r$.

Besides, we can upper bound the Rademacher complexity of any hypothesis in the truncated hypothesis set $\mathcal{H}_r$ as,

$$r = \sup_{h\in\mathcal{H}_r} R(h) = \sup_{h\in\mathcal{H}_r}\mathbb{E}_S[\hat{R}_S(h)] \leq \mathbb{E}_S[\sup_{h\in\mathcal{H}_r}\hat{R}_S(h)] \leq \mathbb{E}_S[\hat{R}_S(h_p)] = R_p. \tag{23}$$

Thus, we can apply Lemma 4 by setting $r = R_p$, obtaining that for any $\delta > 0$, with probability at least $1 - \delta$, the following uniform convergence bound holds for all $h \in \mathcal{H}_r$,

$$R(h) - \hat{R}_S(h) \leq 2\mathfrak{R}_m(\mathcal{L}) + \frac{3M\log(1/\delta)}{4m} + 3\sqrt{\frac{(8\mathfrak{R}_m(\mathcal{L}) + R_p)M\log(1/\delta)}{4m}}. \tag{24}$$

Particularly, as aforementioned, the final model h_k (that is, $\hat{\mathbf{w}}$) returned by the ModelUpdate procedure belongs to the set $\mathcal{H}_r$. So $R(h_k) - \hat{R}_S(h_k)$ also enjoys the upper bound in Eq. (24). Then, we apply Lemma 5 by setting $\alpha = 1/\mu$ to bound the Rademacher complexity term, and obtain that $\mathfrak{R}_m(\mathcal{L}) \leq L\sqrt{\frac{2B^2R_p}{\lambda\mu m}}$. This in conjunction with Eq. (24) yields the desired result of (1) presented in Theorem 1. □

Corollary 1 *If we further suppose an H-smooth condition on the loss function, then under the same conditions (except for the L-Lipschitz continuity) stated in Theorem 1, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds,*

$$R(h_k) - \hat{R}(h_k) = O\left(\frac{\hat{\epsilon}_1}{\sqrt{m}} + \frac{\hat{\epsilon}_2}{m}\right),$$

where $\hat{\epsilon}_1 = \left(\sqrt{R_p} + \sqrt{\frac{\sqrt{H}R_p}{\lambda^2}} + \sqrt[4]{\frac{\sqrt{H}R_p}{\lambda^2 m}}\right)$, *and* $\hat{\epsilon}_2 = \left(\sqrt{\frac{1}{\lambda}} + \sqrt[4]{\frac{1}{\lambda}}\right)$.

Proof From Lemma B.1 of Srebro et al. (2010), we know that for any H-smooth non-negative function $f : \mathbb{R} \mapsto \mathbb{R}_+$ and any $x, y \in \mathbb{R}$, $|f(x) - f(y)| \leq \sqrt{6H(f(x) + f(y))}|x - y|$.

Suppose f is bounded by $M > 0$, we can easily conclude that f is also L-Lipschitz continuous with $L = \sqrt{12HM}$. Thus, we can apply Theorem 1 to obtain a generalization bound. It turns out above bound can be tighter than that obtained by directly analyzing the smoothness condition in Theorem 1 (Kuzborskij and Orabona 2017) under certain conditions. □

Appendix C: Proof of Theorems 2 and 3

Our `WeightUpdate` strategy is essentially exponentially weighted average forecaster (Cesa-Bianchi and Lugosi 2006), and thus the local regret guarantee can be achieved naturally.

Proof *(Proof of Theorem 2)* The argument is based on a simple reduction from our scenario to standard *exponentially weighted average forecaster*. For epoch S_k, let previous models pool $\{h_1, h_2, \ldots, h_{k-1}\}$ be as the expert pool. Then, plugging the expert number $N = k - 1$ and number of instances $n = m_k$ into Cesa-Bianchi and Lugosi (2006), Theorem 2.2 and Cesa-Bianchi and Lugosi (2006), Corollary 2.4, we thus obtain the statement.

Besides, the proof of exponentially weighted average forecaster is standard, which utilizes *potential function* (Cesa-Bianchi and Lugosi 2006; Mohri et al. 2012). For proofs we refer the reader to Mohri et al. (2012), Chapter 7, pp. 157–159. □

Now, we proceed to prove Theorem 3.

Proof *(Proof of Theorem 3)* Our proof relies on the application of local static regret analysis in each epoch.

Let $\mathcal{K}$ denote the number of all epochs, clearly, $\mathcal{K} \leq T/p$. Since $S_1, \ldots, S_{\mathcal{K}}$ is a partition of the whole period T, we thus decompose the dynamic regret as follows,

$$\text{D-Regret}_T = \sum_{k=1}^{\mathcal{K}} \left(L_{S_k} - \sum_{i \in S_k} \ell(h_i^*(\mathbf{x}_i), y_i) \right) \tag{25}$$

$$= \underbrace{\sum_{k=1}^{\mathcal{K}} \left(L_{S_k} - L_{j_k^*} \right)}_{\texttt{term(a)}} + \underbrace{\sum_{k=1}^{\mathcal{K}} \left(L_{j_k^*} - \sum_{i \in S_k} \ell(h_i^*(\mathbf{x}_i), y_i) \right)}_{\texttt{term(b)}} \tag{26}$$

$$\leq \mathcal{K}\sqrt{p \ln(K)/2} + p \sum_{k=1}^{\mathcal{K}} V_k \tag{27}$$

$$\leq \sqrt{\frac{\ln K}{2}} \frac{T}{\sqrt{p}} + p V_T. \tag{28}$$

Equation (25) holds due to the decomposition of regret over $\mathcal{K}$ epochs. We insert the fixed comparator in each epoch that appears in the static regret and thus obtain Eq. (26). Then term (a) can be upper bounded by applying Theorem 2 in each epoch locally, where K is number of the previous models. Besides, term (b) is bounded due to Besbes et al. (2015), Theorem 1, where V_k is the variation within the epoch k. We thus combine them to get Eqs. (27) and (28).

Therefore, we can set $p = \lceil \sqrt[3]{\frac{\ln K}{2}} (T/2V_T)^{2/3} \rceil$ and obtain the dynamic regret in the order of $O\left(V_T^{1/3} T^{2/3}\right)$. □

Appendix D: Proof of Theorem 4

Similar to the arguments in Appendix B, we prove Theorem 4 by first applying Lemma 4. We thus obtain the fast generalization error bound in terms of the Rademacher complexity of loss function family. Then, in Lemma 6, we provide the upper bound of Rademacher complexity of Lipschitz loss function family.

Appendix D.1: Rademacher complexity of Lipschitz loss function family in multi-class case

Lemma 6 *Let $H \subseteq \mathbb{R}^{\mathcal{X} \times \mathcal{Y}}$ be a hypothesis set with $\mathcal{Y} = \{1, 2, \ldots, c\}$. Assume that the non-negative loss function $\ell : \mathbb{R} \mapsto \mathbb{R}_+$ is L-regular and bounded by $M > 0$. Given the previous model h_p (aka, $\boldsymbol{w}_p$), which be a linear weighted combination of previous models. Define the truncated hypothesis set $\mathcal{W}_r = \{W : \Omega(W) \leq \alpha \hat{R}_S(h_p)\}$, whose associated loss function family is then defined as*

$$\mathcal{L} = \left\{ (\boldsymbol{x}, y) \mapsto \ell\left(\rho_{h_W}(\boldsymbol{x}, y)\right) : \ W \in \mathcal{W}_r \right\},$$

and let S be a sample of m points drawn i.i.d. according to distribution $\mathcal{D}$, then we have

$$\mathfrak{R}_m(\mathcal{L}) \leq L c^2 \sqrt{\frac{B^2 \alpha R_p}{m}}. \tag{29}$$

where $R_p = R(h_p)$, $B = \sup_{\boldsymbol{x} \in \mathcal{X}} \|\boldsymbol{x}\|_2$.

Proof To make the presentation clear, we first define the truncated hypothesis class $\mathcal{H}_r = \{h : h \in H \wedge R(h) \leq r\}$. And the empirical Rademacher complexity of $\mathcal{L}$ can be calculated as,

$$\begin{aligned}
\hat{\mathfrak{R}}_S(\mathcal{L}) &= \frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i\ell\big(\rho_{h_W}(\mathbf{x}_i, y_i)\big)\right] \\
&\leq \frac{L}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i\rho_{h_W}(\mathbf{x}_i, y_i)\right] && (30)\\
&\leq \frac{L}{m}\sum_{y\in\mathcal{Y}}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i\rho_{h_W}(\mathbf{x}_i, y)\right] && (31)\\
&= \frac{L}{m}\sum_{y\in\mathcal{Y}}\left\{\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i(h_W(\mathbf{x}_i, y) - \max_{y'\neq y}h_W(\mathbf{x}_i, y'))\right]\right\} && (32)\\
&\leq \frac{L}{m}\sum_{y\in\mathcal{Y}}\left\{\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i h_W(\mathbf{x}_i, y)\right] + \mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i\max_{y'\in\mathcal{Y}\setminus y}h_W(\mathbf{x}_i, y')\right]\right\} && (33)\\
&\leq \frac{L}{m}\sum_{y\in\mathcal{Y}}\left\{\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i h_W(\mathbf{x}_i, y)\right] + \sum_{y'\in\mathcal{Y}\setminus y}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i h_W(\mathbf{x}_i, y')\right]\right\} && (34)\\
&= \frac{Lc^2}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i\langle\mathbf{w}_y, \mathbf{x}_i\rangle\right] && (35)
\end{aligned}$$

where $\mathbf{w}_y \in \mathbb{R}^d = W[:, y]$ is the yth column of W. Meanwhile, Eq. (30) holds due to the fact that loss function $\ell(\cdot)$ is L-Lipschitz, and thus we can apply Talagrand's Comparison Inequality (Ledoux and Talagrand 2013; Mohri et al. 2012) to relate the Rademacher complexity of loss family to margin function family. Equations (31) and (33) hold due to the sub-additivity of supremum function, Eq. (32) is an application of margin definition, Eq. (34) holds due to Lemma 3.

In the following, we proceed to bound the last term. The basic idea is essentially the same as the argument in bounding Eq. (22), a slight trick here is that we need to introduce an auxiliary vector regularizer, the Euclidean norm $\Omega_{\text{aux}}(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2$, to establish the connection between Frobenius norm of matrix and Lemma 2, since Lemma 2 is designed for vector norm.

Similarly, we introduce the variable t_1 and t_2 into two Rademacher complexity term, i.e., term (a) and term (b), respectively. Note that the Euclidean norm $\Omega_{\text{aux}}(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_2^2$ is 1-strongly convex with respect to the ℓ_2-norm $\|\cdot\|_2$, whose dual norm is itself.

$$\begin{aligned}
&\frac{Lc^2}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\langle\mathbf{w}_y, t_1\sigma_i\mathbf{x}_i\rangle\right] \\
&\leq \frac{Lc^2}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\frac{t_1^2}{2}\sum_{i=1}^{m}\frac{1}{2}\|\sigma_i\mathbf{x}_i\|_2^2 + \sup_{W\in\mathcal{W}_r}\frac{1}{2}\|\mathbf{w}_y\|_2^2 + \sum_{i=1}^{m}\frac{1}{2}\langle\nabla\|\mathbf{v}_{1,i-1}\|_2^2, \sigma_i\mathbf{x}_i\rangle\right] && (36)
\end{aligned}$$

$$\leq Lc^2\left(\frac{B^2t_1^2}{4}+\frac{\alpha\hat{R}_S(h_p)}{2m}\right) \tag{37}$$

Inequality Eq. (36) is obtained by applying Lemma 2 with $f = \Omega_{\text{aux}}$, and $\mathbf{u} = \mathbf{w}_y$ and $\mathbf{v}_i = t\mathbf{x}_i$. And the last step in Eq. (37) holds due to the fact that $\Omega_{\text{aux}}(\mathbf{w}_y) \leq \frac{1}{2}\sup_{W\in\mathcal{W}_r}\Omega(W, W_p) \leq \frac{1}{2}\alpha\hat{R}_S(h_p)$ and $\sup_{\mathbf{x}\in\mathcal{X}}\|\mathbf{x}\|_2 \leq B$. By dividing t on both sides, and notice that the above upper bound holds for any $t > 0$, we choose a particular t making the upper bound tight,

$$\frac{Lc^2}{m}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{W\in\mathcal{W}_r}\sum_{i=1}^{m}\sigma_i\langle\mathbf{w}, \mathbf{x}_i\rangle\right] \leq \inf_{t>0} Lc^2\left(\frac{B^2t}{4}+\frac{\alpha\hat{R}_S(h_p)}{2mt}\right) = Lc^2\sqrt{\frac{B^2\alpha\hat{R}_S(h_p)}{2m}}. \tag{38}$$

Therefore, we obtain obtain the upper bound for $\hat{\mathfrak{R}}_S(\mathcal{L})$. Notice that the square-root function is concave, by applying the Jensen's inequality w.r.t. the both side, we have the following upper bound for Rademacher complexity of loss function family,

$$\begin{aligned}\mathfrak{R}_m(\mathcal{L}) &\leq Lc^2\mathbb{E}_S\left[\sqrt{B^2\alpha\hat{R}_S(h_p)/2m}\right] \leq Lc^2\sqrt{B^2\alpha\mathbb{E}_S\left[\hat{R}_S(h_p)\right]/2m} \\ &= Lc^2\sqrt{B^2\alpha R(h_p)/2m}.\end{aligned}$$

Hence, we complete the proof. □

Remark 9 Our analysis follows the framework of Mohri et al. (2012), Chapter 8 and thus shows a quadratic dependency on the number of classes c. In fact, it can be further improved into linear or radical dependency by utilizing Gaussian complexity (Lei et al. 2015) or vector-contraction inequality for Rademacher complexities (Maurer 2016).

Appendix D.2: Proof of main theorem

Proof To prove the generalization bound for the learned model $\hat{W}$ stated in Theorem 4, we turn to show that the statement holds uniformly over the following truncated hypothesis set

$$\mathcal{H}_r = \{\mathbf{x}\mapsto W^{\mathrm{T}}\mathbf{x}:\ \Omega(W, W_p) \leq \frac{1}{\mu}\hat{R}_S(h_p)\ \wedge\ \hat{R}_S(W) \leq \hat{R}_S(h_p)\}.$$

First, we demonstrate that the final model $\hat{W}$ returned by Eq. (9) belongs to the set $\mathcal{H}_r$. Actually, since the $\hat{W}$ is the empirical minimizer of the biased regularization objective function, thus, it is evidently better than the choice $\hat{W}_p$,

$$\frac{1}{m}\sum_{i=1}^{m}\ell(\rho_{\hat{W}}(\mathbf{x}_i, y_i)) + \mu\Omega(\hat{W}, W_p) \leq \frac{1}{m}\sum_{i=1}^{m}\ell(\rho_{W_p}(\mathbf{x}_i, y_i)) + \mu\Omega(W_p, W_p) = \hat{R}_S(h_p),$$

combining the non-negative property of loss and regularizer, we know that $\hat{W}$ satisfies $\hat{R}_S(\hat{W}) \leq \hat{R}_S(h_p)$ and $\Omega(\hat{W}, W_p) \leq \frac{1}{\mu}\hat{R}_S(h_p)$. Therefore, $\hat{W}$ belongs to the hypothesis set $\mathcal{H}_r$.

Besides, similar to the argument in Eq. (23), we know that the upper bound of the risk can be chosen as $r = R_p$, the risk of previous models on target distribution.

Thus, we can apply Lemma 4, obtaining that for $h_{\hat{W}}$,

$$\begin{aligned} R(h_{\hat{W}}) - \hat{R}_S(h_{\hat{W}}) &\leq R_\ell(h_{\hat{W}}) - \hat{R}_S(h_{\hat{W}}) \\ &\leq 2\mathfrak{R}_m(\mathcal{L}) + \frac{3M\log(1/\delta)}{4m} + 3\sqrt{\frac{(8\mathfrak{R}_m(\mathcal{L}) + R_p)M\log(1/\delta)}{4m}}. \end{aligned} \tag{39}$$

Therefore, we can apply Lemma 6 by setting $\alpha = 1/\mu$ and obtain that $\mathfrak{R}_m(\mathcal{L}) \leq Lc^2\sqrt{\frac{B^2R_p}{2\mu m}}$. Plugging this into Eq. (39),

$$\begin{aligned} R(h_{\hat{W}}) - \hat{R}_S(h_{\hat{W}}) \leq Lc^2\sqrt{\frac{B^2R_p}{2\mu m}} + \frac{3M\log(1/\delta)}{4m} \\ + 3\sqrt{\left(8Lc^2\sqrt{\frac{B^2R_p}{2\mu m}} + R_p\right)\frac{M\log(1/\delta)}{4m}}. \end{aligned}$$

By standard derivations and transforms, we conclude the desired conclusion. □

Appendix E: Dataset description

We provide detailed descriptions of datasets used in experiments as follows.

Appendix E.1: Descriptions of synthetic datasets

In the experiments, we adopt four commonly used synthetic datasets and their variants: SEA (SEA200A, SEA200G and SEA500G), CIR500G, SIN500G and STA500G.

The first family is SEA dataset (Street and Kim 2001), which consists of three attributes x_1, x_2, x_3, and $0 \leq x_i \leq 10.0$. The target concept is determined by $x_1 + x_2 \leq b$. For the three variants, there are 24, 000 instances. The drift period of SEA200A and SEA200G is 200, and for SEA500G, the drift period is 500. 'A' indicates $b \in \mathcal{A}$ and 'G' indicates $b \in \mathcal{G}$, where $\mathcal{A} = \{10, 5, 10, 15\}$ and $\mathcal{G} = \{10, 8, 10, 12\}$.

The information of other three synthetic datasets are listed as follows.

- CIR500G is a variant of CIRCLE datasets (Elwell and Polikar 2011), which applies a circle as the decision boundary in a 2-D feature space and simulates concept drift by changing the radius of the circle. The target label is $x_1 + x_2^2 \leq r$ with $r = \{3, 2.5, 2, 2.5, 3, 3.5, 4, 3.5\}$. The drift period is 500.
- SIN500G is a variant of SINE datasets (Elwell and Polikar 2011), which applies a sine curve as the decision boundary in a 2-D feature space and simulates concept drift by changing the angle. The target label is $\sin(x_1 + \theta) \leq x_2$ with $\theta_0 = 0$ and $\Delta\theta = \pi/60$. The drift period is 500.
- STA500G is a variant of STAGGER Boolean Concepts (Schlimmer and Granger 1986), which generate the data with categorical features using a set of rules to determine the class label. Details is included in the paper of Sun et al. (2018). The drift period is 500.

The other six synthetic datasets are 1CDT, 1CHT, UG-2C-2D, UG-2C-3D, UG-2C-5D and GEARS-2C-2D. Their basic information are reported in Table 1. For more details we refer the reader to de Souza et al. (2015).

Appendix E.2: Descriptions of real-world datasets

In the experiments, we adopt nine real-world datasets: Usenet-1, Usenet-2, Luxembourg, Spam, Email, Weather, GasSensor, Powersupply, Electricity and Covertype. The number of data items varies from 1500 to 581,012.

- *Usenet* (Katakis et al. 2008) is split into *Usenet-1* and *Usenet-2* which both consist of 1500 instances with 100 attributes based on 20 newsgroups collection. They simulate a stream of messages from different newsgroups that are sequentially presented to a user, who then labels them according to his/her personal interests.
- *Luxembourg* (Zliobaite 2011) is constructed by using European Social Survey data. There are 1,900 instances with 32 attributes in total, and each instance is an individual and attributes are formed from answers to the survey questionnaire. The label indicates high or low internet usage.
- *Spam* (Katakis et al. 2009) is a real-world textual dataset that uses email messages from the Spam Assassin Collection, and boolean bag-of-words approach is adopted to represent emails. It consists of 9,324 instances with 500 attributes, and label indicates spam or legitimate.
- *Email* (Katakis et al. 2009) is a stream of 1500 examples and 913 attributes which are words that appeared at least 10 times in the corpus (boolean bag-of-words representation), which are collected from 20 Newsgroup collection. The users' personal interests are changing in a recurring manner.
- *Weather* (Elwell and Polikar 2011) dataset is originally collected from the Offutt Air Force Base in Bellevue, Nebraska. 18,159 instances are presented with an extensive range of 50 years (1949 – 1999) and diverse weather patterns. Eight features are selected based on their availability, eliminating those with a missing feature rate above 15%. The remaining missing values are imputed by the mean of features in the preceding and following instances. Class labels are based on the binary indicator(s) provided for each daily reading of rain with 18,159 daily readings: 5698 (31%) positive (rain) and 12,461 (69%) negative (no rain).
- *GasSensor* (Vergara et al. 2012) is a dataset contains 4,450 measurements from 16 chemical sensors utilized in simulations for drift compensation in a discrimination task of six gases (six classes) at various levels of concentrations.
- *Powersupply* (Chen et al. 2015) contains three year power supply records including 29,928 instances with 2 attributes from 1995 to 1998, and our learning task is to predict which hour the current power supply belongs to. We relabel into binary classification according to p.m. or a.m.
- *Electricity* (Harries and Wales 1999) is wildly adopted and collected from the Australian New South Wales Electricity Market where prices are affected by demand and supply of the market. The dataset contains 45,312 instances with 8 features. The class label identifies the change of the price relative to a moving average of the last 24 hours.
- *Covertype* (Gama et al. 2003; Sun et al. 2018) is a real-world data set for describing the observation of a forest area with 51 cartographic variables obtained from US Forest Service (USFS) Region 2 Resource Information System (RIS). Binary class labels are involved to represent the corresponding forest cover type.

References

Bartlett, P. L., Bousquet, O., Mendelson, S., et al. (2005). Local rademacher complexities. *The Annals of Statistics*, *33*(4), 1497–1537.

Bartlett, P. L., & Mendelson, S. (2002). Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, *3*, 463–482.

Besbes, O., Gur, Y., & Zeevi, A. J. (2015). Non-stationary stochastic optimization. *Operations Research*, *63*(5), 1227–1244.

Beygelzimer, A., Kale, S., & Luo, H. (2015). Optimal and adaptive algorithms for online boosting. In *Proceedings of the 32nd international conference on machine learning (ICML)*, pp. 2323–2331.

Bifet, A., & Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. In *Proceedings of the 7th SIAM international conference on data mining (SDM)*, pp. 443–448 .

Bousquet, O. (2002). Concentration inequalities and empirical processes theory applied to the analysis of learning algorithms. PhD thesis, Ecole Polytechnique.

Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D. P., Schapire, R. E., & Warmuth, M. K. (1997). How to use expert advice. *Journal of the ACM*, *44*(3), 427–485.

Cesa-Bianchi, N., & Lugosi, G. (2006). *Prediction, learning, and games*. Cambridge: Cambridge University Press.

Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., & Batista, G. (2015). The ucr time series classification archive. Retrieved September 8, 2018, from http://www.cs.ucr.edu/~eamonn/time_series_data.

Crammer, K., Mansour, Y., Even-Dar, E.,& Vaughan, J. W. (2010). Regret minimization with concept drift. In *Proceedings of the 23rd annual conference computational learning theory (COLT)*, pp. 168–180.

Dietterich, T. G. (2017). Steps toward robust artificial intelligence. AI Magazine, 38(3), 3–24.

Dietterich, T. G. (2019). Robust artificial intelligence and robust human organizations. *Frontiers of Computer Science, 13*(1), 1–3.

Du, S. S., Koushik, J., Singh, A., & Póczos, B. (2017). Hypothesis transfer learning via transformation functions. In *Advances in neural information processing systems 30 (NIPS)*, pp. 574–584.

Duan, L., Tsang, I. W., Xu, D., & Chua, T. (2009). Domain adaptation from multiple sources via auxiliary classifiers. In *Proceedings of the 26th international conference on machine learning (ICML)*, pp. 289–296.

Elwell, R., & Polikar, R. (2011). Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, *22*(10), 1517–1531.

Forman, G. (2006). Tackling concept drift by temporal inductive transfer. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR)*, pp. 252–259.

Gama, J., & Kosina, P. (2014). Recurrent concepts in data streams classification. *Knowledge and Information Systems*, *40*(3), 489–507.

Gama, J., Rocha, R., & Medas, P. (2003). Accurate decision trees for mining high-speed data streams. In *Proceedings of the 9th ACM SIGKDD international conference on knowledge discovery & data mining (KDD)*, pp. 523–528.

Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys*, *46*(4), 44:1–44:37.

Gomes, H. M., Barddal, J. P., Enembreck, F., & Bifet, A. (2017). A survey on ensemble learning for data stream classification. *ACM Computing Surveys*, *50*(2), 23:1–23:36.

Harel, M., Mannor, S., El-Yaniv, R., & Crammer, K. (2014). Concept drift detection through resampling. In *Proceedings of the 31st international conference on machine learning (ICML)*, pp. 1009–1017.

Harries, M., & Wales, N. S. (1999). Splice-2 comparative evaluation: Electricity pricing. Technical Report of South Wales University.

Hazan, E. (2016). Introduction to online convex optimization. *Foundations and Trends in Optimization*, *2*(3–4), 157–325.

Helmbold, D. P., & Long, P. M. (1994). Tracking drifting concepts by minimizing disagreements. *Machine Learning*, *14*(1), 27–45.

Jaber, G., Cornuéjols, A., & Tarroux, P. (2013). A new on-line learning method for coping with recurring concepts: The ADACC system. In *Proceedings of the 20th international conference on neural information processing (ICONIP)*, pp. 595–604.

Kakade, S. M., Shalev-Shwartz, S., & Tewari, A. (2012). Regularization techniques for learning with matrices. *Journal of Machine Learning Research*, *13*, 1865–1890.

Katakis, I., Tsoumakas, G., Banos, E., Bassiliades, N., & Vlahavas, I. P. (2009). An adaptive personalized news dissemination system. *Journal of Intelligent Information Systems*, *32*(2), 191–212.

Katakis, I., Tsoumakas, G., & Vlahavas, I. P. (2008). An ensemble of classifiers for coping with recurring contexts in data streams. In *Proceedings of the 18th European conference on artificial intelligence (ECAI)*, pp. 763–764.

Katakis, I., Tsoumakas, G., & Vlahavas, I. P. (2010). Tracking recurring contexts using ensemble classifiers: An application to email filtering. *Knowledge and Information Systems*, *22*(3), 371–391.

Klinkenberg, R. (2004). Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, *8*(3), 281–300.

Klinkenberg, R., & Joachims, T. (2000). Detecting concept drift with support vector machines. In *Proceedings of the 17th international conference on machine learning (ICML)*, pp. 487–494.

Koltchinskii, V. (2011). *Oracle inequalities in empirical risk minimization and sparse recovery problems* (Vol. 2033). Berlin: Springer.

Kolter, J. Z., & Maloof, M. A. (2003). Dynamic weighted majority: A new ensemble method for tracking concept drift. In *Proceedings of the 3rd IEEE international conference on data mining (ICDM)*, pp. 123–130.

Kolter, J. Z., & Maloof, M. A. (2005). Using additive expert ensembles to cope with concept drift. In *Proceedings of the 22rd international conference on machine learning (ICML)*, pp. 449–456.

Kolter, J. Z., & Maloof, M. A. (2007). Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, *8*, 2755–2790.

Koolen, W. M., van Erven, T., & Grünwald, P. (2014). Learning the learning rate for prediction with expert advice. In *Advances in neural information processing systems 27 (NIPS)*, pp. 2294–2302.

Koychev, I. (2000). Gradual forgetting for adaptation to concept drift. In *Proceedings of ECAI 2000 workshop on current issues in spatio-temporal reasoning*, pp. 101–106.

Kuncheva, L. I., & Zliobaite, I. (2009). On the window size for classification in changing environments. *Intelligent Data Analysis*, *13*(6), 861–872.

Kuzborskij, I., & Orabona, F. (2013). Stability and hypothesis transfer learning. In *Proceedings of the 30th international conference on machine learning (ICML)*, pp. 942–950.

Kuzborskij, I., & Orabona, F. (2017). Fast rates by transferring from auxiliary hypotheses. *Machine Learning*, *106*(2), 171–195.

Ledoux, M., & Talagrand, M. (2013). *Probability in banach spaces: Isoperimetry and processes*. Berlin: Springer.

Lei, Y., Dogan, Ü., Binder, A., & Kloft, M. (2015). Multi-class svms: From tighter data-dependent generalization bounds to novel algorithms. In *Advances in neural information processing systems 28 (NIPS)*, pp. 2035–2043.

Li, N., Tsang, I. W., & Zhou, Z. H. (2013). Efficient optimization of performance measures by classifier adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(6), 1370–1382.

Maurer, A. (2016). A vector-contraction inequality for rademacher complexities. In *Proceedings of the 27th international conference on algorithmic learning theory (ALT)*, pp. 3–17.

Mohri, M., & Medina, A. M. (2012) New analysis and algorithm for learning with drifting distributions. In *Proceedings of the 23rd international conference on algorithmic learning theory (ALT)*, pp. 124–138.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). *Foundations of machine learning*. Cambridge: MIT press.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). *Foundations of machine learning* (2nd ed.). Cambridge: The MIT Press.

Rad, R. H., & Haeri, M. A. (2019). Hybrid forest: A concept drift aware data stream mining algorithm. CoRR arXiv:1902.03609.

Reddi, S. J., Póczos, B., & Smola, A. J. (2015). Doubly robust covariate shift correction. In *Proceedings of the twenty-ninth AAAI conference on artificial intelligence (AAAI)*, pp. 2949–2955.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*, 197–227.

Schapire, R. E., & Freund, Y. (2012). *Boosting: Foundations and algorithms*. Cambridge: The MIT Press.

Schlimmer, J. C., & Granger, R. H. (1986). Incremental learning from noisy data. *Machine Learning*, *1*(3), 317–354.

Schölkopf, B., Herbrich, R., & Smola, A. J. (2001). A generalized representer theorem. In *Proceedings of the 14th annual conference computational learning theory (COLT)*, pp. 416–426.

Segev, N., Harel, M., Mannor, S., Crammer, K., & El-Yaniv, R. (2017). Learn on source, refine on target: A model transfer learning framework with random forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(9), 1811–1824.

Shewchuk, J. R. (1994). An introduction to the conjugate gradient method without the agonizing pain. Carnegie Mellon University.

de Souza, V. M. A., Silva, D. F., Gama, J., & Batista, G. E. A. P. A. (2015). Data stream classification guided by clustering on nonstationary environments and extreme verification latency. In *Proceedings of the 2015 SIAM international conference on data mining (SDM)*, pp. 873–881.

Srebro, N., Sridharan, K., & Tewari, A. (2010). Smoothness, low noise and fast rates. In *Advances in neural information processing systems 23 (NIPS)*, pp. 2199–2207.

Sridharan, K., Shalev-Shwartz, S., & Srebro, N. (2008). Fast rates for regularized objectives. In *Advances in neural information processing systems 21 (NIPS)*, pp. 1545–1552.

Street, W. N., & Kim, Y. (2001). A streaming ensemble algorithm (SEA) for large-scale classification. In *Proceedings of the 7th ACM SIGKDD international conference on knowledge discovery & data mining (KDD)*, pp. 377–382.

Sun, Y., Tang, K., Zhu, Z., & Yao, X. (2018). Concept drift adaptation by exploiting historical knowledge. *IEEE Transactions on Neural Networks and Learning Systems*, *29*(10), 4822–4832.

Suykens, J. A., Van Gestel, T., & De Brabanter, J. (2002). *Least squares support vector machines*. Singapore: World Scientific.

Tommasi, T., Orabona, F., & Caputo, B. (2010). Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *Proceedings of the 23rd IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 3081–3088.

Tommasi, T., Orabona, F., & Caputo, B. (2014). Learning categories from few examples with multi model knowledge transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(5), 928–941.

Vergara, A., Vembu, S., Ayhan, T., Ryan, M. A., Homer, M. L., & Huerta, R. (2012). Chemical gas sensor drift compensation using classifier ensembles. *Sensors and Actuators B: Chemical*, *166*, 320–329.

Wu, X. Z., Liu, S., & Zhou, Z. H. (2019). Heterogeneous model reuse via optimizing multiparty multiclass margin. In *Proceedings of the 36th international conference on machine learning (ICML)*, pp. 6840–6849.

Yang, T., Li, Y. F., Mahdavi, M., Jin, R., & Zhou, Z. H. (2012). Nyström method vs random fourier features: A theoretical and empirical comparison. In *Advances in neural information processing systems 25 (NIPS)*, pp. 476–484.

Ye, H. J., Zhan, D. C., Jiang, Y., & Zhou, Z. H. (2018). Rectify heterogeneous models with semantic mapping. In *Proceedings of the 35th international conference on machine learning (ICML)*, pp. 1904–1913.

Zhang, L., Lu, S., & Zhou, Z. H. (2018). Adaptive online learning in dynamic environments. In *Advances in neural information processing systems 31 (NeurIPS)*, pp. 1330–1340.

Zhao, P., Wang, X., Xie, S., Guo, L., & Zhou, Z.-H. (2019). Distribution-free one-pass learning. IEEE Transaction on Knowledge and Data Engineering. https://doi.org/10.1109/TKDE.2019.2937078.

Zhou, Z. H. (2012). *Ensemble methods: Foundations and algorithms*. London: Chapman & Hall/CRC Press.

Zhou, Z.-H. (2016). Learnware: On the future of machine learning. *Frontiers of Computer Science, 10*(4), 589–590.

Zhou, Z.-H. (2019). Abductive learning: Towards bridging machine learning and logical reasoning. *Science China Information Sciences, 62*(7), 76101:1–76101:3.

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (ICML)*, pp. 928–936.

Zliobaite, I. (2011). Combining similarity in time and space for training set formation under concept drift. *Intelligent Data Analysis*, *15*(4), 589–611.