Concept Drift-Tolerant Transfer Learning in Dynamic Environments

Cuie Yang, Yiu-ming Cheung, *Fellow, IEEE*, Jinliang Ding, *Senior Member, IEEE*, and Kay Chen Tan, *Fellow, IEEE*

***Abstract*— Existing transfer learning methods that focus on problems in stationary environments are not usually applicable to dynamic environments, where concept drift may occur. To the best of our knowledge, the concept drift-tolerant transfer learning (CDTL), whose major challenge is the need to adapt the target model and knowledge of source domains to the changing environments, has yet to be well explored in the literature. This article, therefore, proposes a hybrid ensemble approach to deal with the CDTL problem provided that data in the target domain are generated in a streaming chunk-by-chunk manner from nonstationary environments. At each time step, a class-wise weighted ensemble is presented to adapt the model of target domains to new environments. It assigns a weight vector for each classifier generated from the previous data chunks to allow each class of the current data leveraging historical knowledge independently. Then, a domain-wise weighted ensemble is introduced to combine the source and target models to select useful knowledge of each domain. The source models are updated with the source instances performed by the proposed adaptive weighted CORrelation ALignment (AW-CORAL). AW-CORAL iteratively minimizes domain discrepancy meanwhile decreases the effect of unrelated source instances. In this way, positive knowledge of source domains can be potentially promoted while negative knowledge is reduced. Empirical studies on synthetic and real benchmark data sets demonstrate the effectiveness of the proposed algorithm.**

***Index Terms*— Concept drift, ensemble learning, negative transfer, positive transfer, transfer learning.**

Manuscript received July 14, 2020; revised November 17, 2020; accepted January 21, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61672444 and Grant 61988101, in part by the Hong Kong Baptist University (HKBU) under Grant RC-FNRA-IG/18-19/SCI/03 and Grant RC-IRCMs/18-19/SCI/01, in part by the Innovation and Technology Fund of Innovation and Technology Commission of the Government of the Hong Kong SAR under Project ITS/339/18, in part by the National Key Research and Development Program of China under Grant 2018YFB1701104, in part by the Xingliao Plan of Liaoning Province under Grant XLYC1808001, in part by the Science and Technology Program of Liaoning Province under Grant 2020JH2/10500001 and Grant 2020JH1/10100008, and in part by the Research Grants Council of the Hong Kong SAR under Grant CityU11202418 and Grant CityU11209219. *(Corresponding authors: Yiu-ming Cheung; Jinliang Ding.)*

Cuie Yang is with the Department of Computer Science, Hong Kong Baptist University, Hong Kong, and also with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China (e-mail: cuieyang@comp.hkbu.edu.hk).

Yiu-ming Cheung is with the Department of Computer Science, Hong Kong Baptist University, Hong Kong (e-mail: ymc@comp.hkbu.edu.hk).

Jinliang Ding is with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China (e-mail: jlding@mail.neu.edu.cn).

Kay Chen Tan is with the Department of Computing, Hong Kong Polytechnic University, Hong Kong (e-mail: kaychen.tan@polyu.edu.hk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3054665.

Digital Object Identifier 10.1109/TNNLS.2021.3054665

I. Introduction

TRANSFER learning aims to enhance the learning performance of a model in a target domain by transferring knowledge from source domains [1], [2]. It has been extensively studied in addressing target tasks that have sparse or no labeled data. In the past decades, efforts have been paid to improve the performance of transfer learning along two aspects: 1) boosting positive knowledge transfer; and 2) avoiding negative knowledge transfer [3]. Because of discrepancies between the source and target domains, approaches to improve useful knowledge transfer focus on minimizing the domain discrepancy across the source and target domains, such as instance re-weighting [4], [5] and feature matching [6], [7]. On the other hand, techniques for reducing negative knowledge transfer often try to weaken the effect of irrelevant source data via decreasing their weights [3].

Most existing studies on transfer learning focus on stationary environments, assuming that the data in each domain are generated from the same distribution. However, this assumption may not be true from the practical point of view, e.g., financial data inferences, energy demand predictions, and climate data analysis, where the environments are dynamic. In case the environment of a domain changes, meaning the distributions of the generated data evolving over time, this domain is a concept drift problem [8], [9]. Compared with the transfer learning in a static environment, the one in dynamic environments require the learner to continuously adapt to the concept drift as well as adaptively leverage useful knowledge from source domains at each environment. The new difficulties of transfer learning in dynamic environments challenge traditional transfer learning approaches because they are not designed to react to concept drift and track the evolving data.

This article concentrates on studying multisource concept drift-tolerant transfer learning (CDTL), particularly for the case that only the environment of the target domain changes over time. Specifically, the source data are given in advance while the target data are continuously collected from different distributions and a chunk of data is available at each distribution. We aim to design a CDTL model that can well adapt to concept drift and extracts knowledge from source domains to improve its learning performance. In the literature, ensemble learning is a popular approach to concept drift problems [10], [11]. The ensemble learning for concept drift incorporates a set of classifiers that are learned from previous time steps, whereby adapting to new concepts as

well as preserving the knowledge of historical classifiers. The most used techniques in such approaches include adding new individual classifiers, removing old classifiers, and adjusting weights of individual classifiers [10], [11]. Inspired by the effectiveness of the ensemble model in addressing concept, the proposed CDTL model uses the ensemble strategy as the learning algorithms to deal with the concept drift of the target domain. In other words, the CDTL model on each time step leverages the knowledge from both source domains and historical environments of the target domain. It should be noted that in the concept drift problems, the data of historical environments are not stored but the limited learned knowledge is accessible [12], e.g., the learned classifiers. A similar idea to CDTL has been reported in [13], named Melanie. It considers an online problem in which the data in source and target domains are generated from non-stationary environments. Melanie applies an online ensemble to learn models from each of the domain and combines these models via a weighted-sum approach. It incrementally trains the models and dynamically adjusts their weights to handle the concept drift. Generally, Melanie could be extended to address CDTL by replacing the online learning ensemble with an ensemble for chunk-based concept drift.

However, such an extension has the following two limitations: 1) when reacting to a new concept, the existing ensembles for concept drift often adjust the weights of historical individual classifiers to preserve historical knowledge. These weights often represent the contributions of historical classifiers to the current learning. Weight of a historical classifier is usually assigned according to its performance on the current data chunk [14], [15]. Existing approaches generally evaluate a global performance and assign a single weight to a classifier. However, they ignore the performance variation of a classifier at different locations of a data chunk and the different contributions to learning the different parts of the current data. Suppose a shift across a historical concept and a new one occurs only in a few classes, or in different degrees, the classifier trained on the historical concept may perform diversely on classes of the new concept. An example is illustrated in Fig. 1, in which Ct 1 and Ct 2 are the two historical concepts, and CurCt is the current concept. In the example, the drift between Ct 1 and the current concept CurCt only occurs on Class 1 and 2. In this case, the classifier of Ct 1 would perform well on Class 3 and 4, while worse on Class 1 and 2. An extreme situation is across Ct 2 and CurCt, where the concept drift only causes the labels of Class 3 and 4 swapping. In this scenario, an ideal classifier of Ct 2 would totally misclassify the instances in Class 3 and 4, yet work perfectly on Class 1 and 2. Therefore, it is not appropriate to set a shared global weight for a historical classifier when preserving its knowledge for future learning and 2) to leverage the knowledge of source domains, a straightforward way is to combine the learned classifiers of these domains with the target classifier. However, combining the classifiers directly learned from source data may hinder effective knowledge transfer due to domain discrepancies.

This article, therefore, presents a Hybrid Ensemble approach to concept drift-tolerate transfer learning (HE-CDTL), which

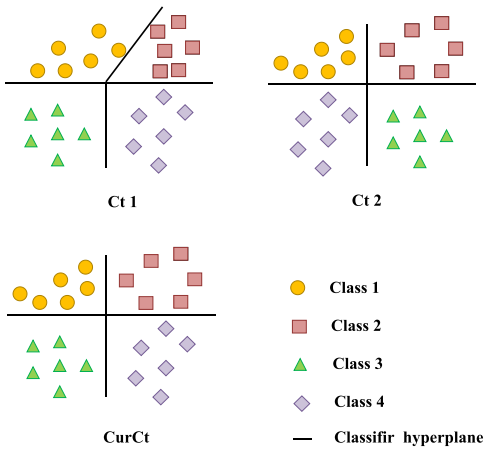


Fig. 1. Illustration of different shifts among classes. Ct 1 and Ct 2 are the two historical concepts, and CurCt is a concept at the current time step. The drift between Ct 1 and CurCt exists in Class 1 and Class 2. The drift across the Ct 2 and CurCt only causes the labels of Class 3 and Class 4 swapping.

consists of three main stages: 1) a class-wise weighted ensemble is introduced in HE-CDTL for tackling the concept drift of the target domain. In particular, at each arrived data chunk, a new classifier is created and assigned a weight vector. Each element in the weight vector is associated with a class and initialized to 1. Then, historical classifiers are individually evaluated on each class of the newest data chunk. Accordingly, a weight vector of each classifier is obtained according to the evaluated performances on the classes of the newest data chunk. After that, the ensemble model of target domains is formulated by a weighted combination of these individual classifiers based on the weight vectors. In this way, the previous knowledge can be flexibly leveraged when a classifier predicts the different classes of the current data; 2) a domain-wise weighted ensemble model, which aims to select useful knowledge from source and target domains, is created by integrating the source and target models; and 3) an adaptive weighted CORrelation alignment (AW-CORAL) is proposed to promote the effective knowledge transfer from source domains. Its purpose is to reduce the distribution discrepancy across the source and target domains, as well as to avoid transferring negative knowledge of source domains. AW-CORAL initializes a weight vector for each source domain and projects source domains associated with the weight vectors via the proposed weighted CORrelation alignment (CORAL) [7]. In AW-CORAL, the weight vectors of source domains are iteratively adjusted in order to decrease the effects of irrelevant source instances. Consequently, the source models are updated over the iterations.

The main contributions of this article are summarized as follows.

1) A hybrid ensemble approach is proposed to tackle CDTL, where data in target domains are generated from nonstationary environments. The purpose of HE-CDTL is to extract the historical knowledge of the target domain and the knowledge of source domains to achieve good performance in future time steps.

2) A class-wise weighted ensemble is introduced to preserve the useful historical knowledge in target domains.
3) An AW-CORAL is presented to promote positive knowledge while reducing negative knowledge transfer from source domains.

The rest of this article is organized as follows. Related works consisting of transfer learning and ensemble methods for concept drift problems are introduced in Section II. The proposed algorithm is described in detail in Section III. Experiments and discussions are presented in Section IV. Finally, the conclusion is drawn in Section V.

II. Related Work

This section makes an overview of the related research topics, i.e., transfer learning and ensemble learning for concept drift problems.

A. Transfer Learning

Transfer learning has been attracting increasing interest in recent years [1], [2], [16]. In transfer learning, instances of source and target domains are often generated from different distributions. Thus, the minimization of the distribution discrepancy across domains plays a crucial role in achieving good performance in transfer learning. A variety of approaches have been developed to reduce domain discrepancy, which can be generally classified into two categories, i.e., instance re-weighting and feature matching [17]. Instance re-weighting methods decrease domain discrepancy via adjusting the weights of source instances, so as to reuse the source instances that are similar to the target domain. The key feature of instance re-weighting methods is on how the weights of instances are estimated. For example, Huang *et al.* [18] proposed a kernel mean matching (KMM) approach, which estimates the weights via minimizing the mean across the source and target domain instances in a reproducing kernel Hilbert space (RKHS). Sugiyama *et al.* [19] presented a Kullback–Leibler importance estimation procedure (KLIEP) which applies Kullback–Leibler distance as a domain distribution measurement. During the training process, KLIEP not only minimizes the Kullback–Leibler distance but also performs a model selection procedure. Based on the existing instance re-weighting techniques, Sun *et al.* [20] proposed a 2-Stage weighting framework for multisource domain adaption (2SW-MDA) to address multisource transfer learning problems. In 2SW-MDA, the source domains and their instances are re-weighted simultaneously to reduce marginal distribution and conditional distribution discrepancies, respectively. The technique of weighting instances is similar to KMM and the realization of domain weighting is based on the smoothness assumption.

TrAdaBoost [21] is an approach based on the framework of AdaBoost [22], which iteratively updates the weights of training data. At each iteration, TrAdaBoost trains a classifier on the combination of source and target data, then uses this classifier to predict the training data. For a source instance, its weight is reduced if it is wrongly predicted. Consequently, its impact on the classifier is decreased. On the contrary, the weights of misclassified target instances are increased to have a larger impact. Multisource TrAdaBoost (MsTrAdaBoost) [23] is an extension of TrAdaBoost to solve multisource transfer learning problems. MsTrAdaBoost combines each source and target data, and trains a classifier for each of them. Then, MsTrAdaBoost selects the classifier that has the minimal error on the target data to update the weights of instances.

Feature matching aims to discover a common feature representation space between the source and target domains. The common space can be obtained by either a symmetric transformation or an asymmetric transformation. The symmetric transformation maps both source and target domains to a new space while the asymmetric transformation projects one to another domain. The transfer component analysis (TCA) approach proposed by Pan *et al.* [6] is a typical symmetric transformation. It employs maximum mean discrepancy (MMD) [24]–[26] as a measurement to learn an RKHS, where the marginal distribution discrepancy across the source and target domains is minimized. Based on TCA, the joint distribution adaptation (JDA) [27] has been proposed to reduce both marginal and conditional distribution discrepancies. Considering the different importance of marginal and conditional distribution discrepancies across problems, Wang *et al.* [28], [29] presented a Balanced Distribution Adaptation (BDA) approach, which introduces a balance factor between the marginal and conditional distribution discrepancies to adaptively leverage the importance of them. Subspace alignment (SA) [30] is an approach to aligning domain distributions in a lower-dimensional subspace. It first selects a number of important eigenvectors of source and target domains using principal component analysis [31]. Then, it learns a linear transformation matrix to minimize the eigenvectors of the two domains. As reported in [32], SA ignores the different distributions of domains in the subspace. For this remedy, an approach, namely distribution alignment between two subspaces (SDA-TS) [32], has been proposed to align the bases as well as the distributions. Correlation alignment (CORAL) [7], [33] is a well-known asymmetric transformation approach, which also aims to align the subspace bases. Unlike SA and SDA-TS that apply the first-order statistics to find the sub-spaces of domains, CORAL aligns the second-order statistics. Besides, CORAL adopts the learned transformation matrix to project source instances to the target domain.

Although many feature matching approaches have been developed in transfer learning, it is important to avoid a negative transfer. The negative transfer refers to a phenomenon that the transferred knowledge degenerates the performance of target tasks. As pointed in [3], transferring unrelated or harmful source samples to the target is a reason for negative transfer. In the case where the source and target domains are dissimilar, the unrelated source samples can still exist after feature matching. For this reason, the negative transfer could be overcome by reducing the impact of unrelated source instances via weighting instances. For example, transfer joint matching (TJM) [17] introduces a sparsity regularizer on the feature transformation matrix to jointly match the features and re-weight the instances. Zhong *et al.* [34] developed

an adaptive mapping approach that matches features across the source and target domains, and updates the weights of the training instances iteratively. To address partial transfer learning problems [35]–[37], in which the source domain has more classes than the target one, [35]–[37] introduce an instance-level re-weighting and class-level re-weighting mechanisms to reduce the outlier classes of source domains. Also, [3] employs an adversarial neural network to align domain distribution, where lower weights are assigned to the source samples that are far away from the discriminator since these instances are considered to be less related to the target domain.

B. Ensemble Learning for Concept Drift

This subsection reviews the ensemble approaches to chunk-based concept drift problems. In general, these methods maintain a number of classifiers learned from historical data chunks in an archive. They react to concept drift through managing the individual classifiers and their combinations [10], [11], [38], [39]. The streaming ensemble algorithm (SEA) proposed by Street and Kim [14] is the earliest approach to concept drift. It creates a classifier for each newly arrived data chunk and assigns it a weight. If the maximum size of the archive is reached, one existing classifier that performs the poorest on the current data chunk is removed. SEA applies the mean square error as a performance measurement to preserve the most accurate classifiers of the ensemble. For the selected classifiers, SEA assigns a weight for each of them according to their performances and uses the weighted voting to combine the achieved and the new classifiers. Accuracy weighted ensemble (AWE) [40] is an approach similar to SEA, which continuously creates classifiers on the incoming data chunks. However, AWE discards the historical classifiers whose prediction errors are higher than the one of a randomly selected classifier. Accuracy updated ensemble (AUE) introduced by Brzezinski and Stefanowski [15], [41] aims to improve the adaption ability of an ensemble. AUE not only trains a new classifier on the arrived data chunk but also incrementally updates the existing classifiers with the new data chunk. The ensemble prediction of AUE is made according to the weighted combination, where the weights of individual classifiers are the evaluated performance on the newest data chunk. Following the way of adapting classifiers in AUE, Sun *et al.* [42] presented an approach, namely diversity and transfer based ensemble learning approach (DTEL), which employs Yules Q-statistic [43] as a quality measurement to select individual classifiers in order to maintain the diversified ensemble components, thereby avoiding the problem of overfitting.

Learn++ for nonstationary environments (called Learn++.NSE) is an approach based on boosting framework [12]. Learn++.NSE first weights each sample of the new coming data chunk. Specifically, Learn++.NSE initializes the weights of all samples to 1, and reduces the weight of a sample if it is misclassified by the ensemble. Then, it creates a new classifier on the weighted samples and formulates the ensemble with the weighted combination. The weights of individual classifiers are calculated based on their prediction error on the weighted new data chunk.

It can be found from the above-mentioned methods that most existing approaches determine the weights of classifiers based on performance evaluations, whereby deciding which historical classifiers can be reused and how much knowledge from them can be preserved. However, these works simply evaluate a classifier on the whole data chunk without considering the different performances of a classifier among classes.

III. Proposed Method

In this section, we first introduce the definitions of CDTL and the main framework of the proposed HE-CDTL. Then, we detail the two components of HE-CDTL, i.e., class-wise weighted ensemble and AW-CORAL.

A. Definitions and Overall Framework

In CDTL, the training data of N source domains are given in advance and denoted as $S_1 = (\mathbf{X}_{s,1}, \mathbf{y}_{s,1})$, $S_2 = (\mathbf{X}_{s,2}, \mathbf{y}_{s,2}), \ldots, S_N = (\mathbf{X}_{s,N}, \mathbf{y}_{s,N})$, where $\mathbf{X}_{s,n}$, $\mathbf{y}_{s,n}$, $n = 1, 2, \ldots, N$ are the feature matrix and the label vector of the nth domain, respectively. The training instances of the target domain arrive in a streaming chunk-by-chunk manner. In particular, a chunk of target data $D_t = (\mathbf{X}_t, \mathbf{y}_t)$, where $\mathbf{X}_t$, $\mathbf{y}_t$ are the corresponding feature matrix and label vector, generated from an unknown distribution $p_t(\mathbf{x}, y)$ is available at each time step t, $t = 1, 2, \ldots, T$. Here, T is the total number of time steps in a learning process. Due to concept drift, the distribution p may change over time, i.e., $p_t(\mathbf{x}, y) \neq p_{t+1}(\mathbf{x}, y)$. The goal of CDTL is to learn a good prediction function on the sequential data chunk D_t, $t = 1, 2, \ldots, T$ of the target domain. The challenge of CDTL is on how to effectively borrow knowledge from source domains and historical knowledge for a learner on D_t, $t = 1, 2, \ldots, T$, when there are very few data in each D_t.

The generic diagram of the proposed HE-CDTL is shown in Fig. 2. HE-CDTL is a hybrid ensemble which contains a class-wise weighted ensemble and a domain-wise weighted ensemble, where the class-wise weighted ensemble leverages knowledge from target historical time steps, the domain-wise weighted ensemble extracts knowledge of all source domains and the historical knowledge of the target domain. In the class-wise weighted ensemble, at the t-th time step, K classifiers $f_1^t, f_2^t, \ldots, f_K^t$ learned from previous time steps are stored in an achieve B_t. On the arrived data chunk D_t, HE-CDTL first creates a new classifier f_t. Then, f_t and the selected individual classifiers from B_t are combined as F_t via the proposed class-wise ensemble strategy (detailed in Section III-B), which is considered to be the model of the target domain at the t-th time step. Regarding the source domains $S_1, S_2, \ldots, S_N$, AW-CORAL is presented to improve the positive knowledge transfer meanwhile reducing the negative transfer. In particular, AW-CORAL assigns a weight vector $\mathbf{w}_n$ to each source domain S_n, $n = 1, 2, \ldots, N$, where each element in $\mathbf{w}_n$ represents the importance of an instance. Then, the source domains along with the weight vectors are transformed to $S_{1,p}^t, S_{2,p}^t, \ldots, S_{N,p}^t$ via transformation

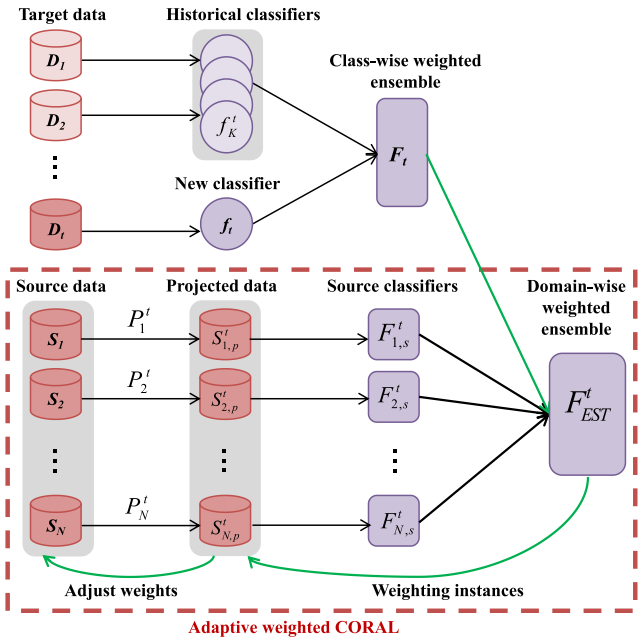


Fig. 2. Framework of the proposed hybrid ensemble approach to a concept drift-tolerate transfer learning problem at the t-th time step.

Algorithm 1 General Framework of HE-CDTL

1: **Input:** T: The total time steps in a learning system; K: The number of historical classifier in the achieve; N: The number of source domains; $B_t = \{f_1^t, f_2^t, \ldots, f_K^t\}$: A set of preserved historical classifiers at the t-th time step; $S_1, S_2, \ldots, S_N$: All the source domains;
2: **Output:** F_{EST}^t: The generated ensemble model at the t-th time step;
3: **while** $t \leq T$ **do**
4: Receive a data chunk D_t;
5: Create the classifier F_t via the class-wise weighted ensemble;
6: Obtain F_{EST}^t via domain-wise weighted ensemble and AW-CORAL;
7: **end while**

matrixes $P_1^t, P_2^t, \ldots, P_N^t$ to reduce the domain discrepancy. Subsequently, source models $F_{1,s}^t, F_{2,s}^t, \ldots, F_{N,s}^t$ are created on the projected source data $S_{1,p}^t, S_{2,p}^t, \ldots, S_{N,p}^t$. Each source model $F_{n,s}^t$,is evaluated on the current data chunk D_t and assigned a weight dw_n^t, $n = 1, 2, \ldots, N$. After that, all source models, the model of the target domain, and their corresponding weights are integrated as the domain-wise weighted ensemble, F_{EST}^t. To reduce the negative transfer, AW-CORAL applies F_{EST}^t to adjust the weights of source instances. Accordingly, the transformation matrixes, the projected source data, the source classifiers and the domain-wise weighted ensemble are updated. In AW-CORAL, the above procedures are iterated until a termination criterion, defined as the maximum iteration $Iter_{max}$ in this article, is reached. The pseudocode of the HE-CDTL is presented in *Algorithm 1*.

B. Class-Wise Weighted Ensemble for Concept Drift

The main feature of the proposed class-wise weighted ensemble technique is that it evaluates the performance of a classifier on each class instead of the whole data. Generally, the class-wise weighted ensemble technique can be implemented in any ensemble approach to concept drift, such as SEA [14], AUE2 [15], and Lear++.NSE [12]. This article incorporates it in SEA and denotes it as class-wise weighted SEA (C-SEA). Similar to SEA, C-SEA includes three components to handle new concepts, i.e., train a new classifier on the most current data chunk, remove the worst classifier when the maximum size is reached, and update the weights of classifiers. The pseudocode of C-SEA at a certain time step t is shown in *Algorithm 2*.

Algorithm 2 Training Process of C-SEA at Time Step t

1: **Input:** K: The number of archived classifiers; $B_t = \{f_1^t, f_2^t, \ldots f_K^t\}$: The existing classifiers at the t-th time step; D_t: The training data chunk at the t-th time step; C: The number of classes of D_t; K_{max}: The maximum size of the archive;
2: **Output:** F_t: The created ensemble model at time step t;
3: Create a new classifier f_t on D_t;
4: Set a weight vector $\mathbf{cw}_t$ for f_t, $cw_{t,c} = 1$, $c = 1, 2, \ldots, C$;
5: **for** k = 1: K **do**
6: Calculate the accuracy of f_k^t on each class of D_t according to Eq. (1);
7: Calculated the weight vectors $\mathbf{cw}_1^t, \mathbf{cw}_2^t, \ldots, \mathbf{cw}_K^t$ for historical classifiers $f_1^t, f_2^t, \ldots f_K^t$ according to Eq. (2);
8: **end for**
9: **if** $K >= K_{max}$ **then**
10: Remove the classifier with the smallest $cw_{k,c}^t$, $c = 1, 2, .., C$;
11: **end if**
12: Normalize the weights of individual classifiers according to Eq. (3) and Eq. (4);
13: Construct the ensemble F_t according to Eq. (5);
14: Add the new classifier f_t to B_t and denoted as B_{t+1};

Given the data chunk D_t, C-SEA learns a new classifier from D_t, which is denoted as f_t. Then, C-SEA assigns a weight vector $\mathbf{cw}_t = (cw_{t,1}, cw_{t,2}, \ldots, cw_{t,C})$ for f_t, C is the number of classes of D_t. Each element in $\mathbf{cw}_t$ is associated with a class and is set to 1. Meanwhile, C-SEA evaluates the archived K classifiers $f_1^t, f_2^t, \ldots, f_K^t$ on each class of D_t. We use the prediction accuracy as the performance measurement, and the accuracy of a classifier f_k, $k = 1, 2, \ldots, K$ is calculated according to the following:

$$\mathrm{Auc}_{k,c}^t = \frac{\sum_{i \in U_c} \hat{y}_k^i == c}{|U_c|}, \quad c = 1, 2, \ldots, C \tag{1}$$

where $\hat{y}_k^i$ is the predicted label of f_k on the ith instance, U_c contains the indices of instances whose labels are c, $|U_c|$ is the cardinality of U_c, and C is the number of classes of D_t.

Once the performances of the existing classifiers are obtained, C-SEA assigns each classifier f_k^t a weight vector $\mathbf{cw}_k^t$ according to the evaluated performance. In particular,

the weight vector of a classifier on each class is set to the estimated accuracy

$$cw_{k,c}^{t} = \text{Auc}_{k,c}^{t}, \quad k = 1, 2, \ldots, K, \ c = 1, 2, \ldots, C. \quad (2)$$

If K is smaller than the maximum size of the archive $K_{\max}$, the new trained classifier f_t is simply added in the archive. Otherwise, one archived classifier with the smallest weight element will be discarded and removed from the archive

After the individual classifiers of the ensemble are selected, the weight vectors of these classifiers are normalized before they are combined. Specially, the weight vectors of the old classifiers and the new created classifier are normalized according to the following:

$$Ncw_{k,c}^{t} = \frac{cw_{k,c}^{t}}{(\sum_{k=1}^{K} cw_{k,c}^{t} + cw_{t,c})}, \quad k = 1, 2, \ldots, K, \ c = 1, 2, \ldots, C. \quad (3)$$

$$Ncw_{t,c} = \frac{cw_{t,c}}{(\sum_{k=1}^{K} cw_{k,c}^{t} + cw_{t,c})}, \quad k = 1, 2, \ldots, K, \ c = 1, 2, \ldots, C. \quad (4)$$

Finally, the class-wise weighted ensemble F_t at time step t can be formulated according to the following:

$$F_t = \sum_{c=1}^{C} \sum_{k=1}^{K} Ncw_{k,c}^{t} \times f_k^t + Ncw_{t,c} \times f_t. \quad (5)$$

It can be seen that C-SEA assigns each class a specific weight instead of a single weight. In the situation where all elements in the weight vector of a classifier are equal, C-SEA is the same as SEA. Consequently, SEA can be regarded as a special case of C-SEA. Compared with SEA, C-SEA allows each class to flexibly select positive knowledge from a classifier, whereby improving the prediction accuracy of the ensemble.

C. AW-CORAL

As described in Section III-A, a data chunk can be achieved at each time step of the target domain in CDTL. Thus, CDTL can be regarded as a traditional multisource transfer learning problem at a certain time step. A common way of leveraging knowledge from source domains is to integrate all source models as an ensemble [44], [45]. Inspired by the multisource transfer learning, this article adopts an ensemble to preserve the knowledge of source domain. In particular, we use a domain-wise weighted ensemble to combine each source models and the class-wise weighted ensemble model. In this way, the useful knowledge of source domains and the historical knowledge of the target domain can be selected and reused. Suppose the weights for N source classifiers $F_{1,s}^{t}, F_{2,s}^{t}, \ldots, F_{N,s}^{t}$ and the class-wise weighted ensemble F_t are $dw_{1,s}^{t}, dw_{2,s}^{t}, \ldots, dw_{N,s}^{t}$ and dw_t, the domain-wise weighted ensemble can be denoted as

$$F_{\text{EST}}^{t} = \sum_{n=1}^{N} dw_{n,s}^{t} \times F_{n,s}^{t} + dw_t \times F_t. \quad (6)$$

In F_{EST}^{t}, the weights of source classifiers are set to their prediction accuracy on the latest data chunk D_t. The weight for F_t is defined in the following:

$$dw_t = \sum_{k=1}^{K} \frac{\sum_{c=1}^{C} cw_{k,c}^{t}}{C} + \frac{\sum_{c=1}^{C} cw_{t,c}}{C} \quad (7)$$

which represents the sum weights of each individual classifier.

To promote the positive knowledge transfer, a feature matching approach is often performed to transform domains to new spaces, where the distributions of source and target domains are close. Since there is a small number of labeled data and no unlabeled data in each D_t, we use feature matching techniques to minimize the marginal distributions across the source and target domains. CORAL [7] is a simple yet efficient feature matching method, which attempts to project the data of source domains to the target space. Given a source domain $D_S = (\mathbf{X}_S, \mathbf{y}_S)$ and a target domain $D_T = (\mathbf{X}_T, \mathbf{y}_T)$, where $\mathbf{X}_S$ and $\mathbf{X}_T$ are the feature matrixes, and $\mathbf{y}_S$ and $\mathbf{y}_T$ are the label vectors. CORAL first extracts the second-order statistics, i.e., the covariance of the source and target features according to the following:

$$\mathbf{C}_S = \text{cov}(\mathbf{X}_S) + \text{eye}(\text{size}(\mathbf{X}_S, 2)) \quad (8)$$

$$\mathbf{C}_T = \text{cov}(\mathbf{X}_T) + \text{eye}(\text{size}(\mathbf{X}_T, 2)). \quad (9)$$

In $\mathbf{C}_S$ and $\mathbf{C}_T$, the first component is the covariance matrix of $\mathbf{X}_S$ and $\mathbf{X}_T$, and the second component is a unit matrix that has the same size as $\mathbf{X}_S$ and $\mathbf{X}_T$, respectively.

Then, CORAL applies a linear transformation matrix $\mathbf{A}$ to align the covariances of source and target domains. The transformation matrix can be obtained via solving

$$\text{CORAL}(D_S, D_T) = \min_{\mathbf{A}} \left\| \mathbf{A}'\mathbf{C}_S\mathbf{A} - \mathbf{C}_T \right\|_F^2 \quad (10)$$

where $\mathbf{A}'$ is the transposed matrix of $\mathbf{A}$, and $\|\cdot\|_F^2$ is the matrix Frobenius norm. Subsequently, the original source features are projected by $\mathbf{A}$.

Similar to other discrepancy methods, standard CORAL may fail to discover source instances that are not relevant to the target in case the domain shifts are very large. As analyzed in [3] and [34], transferring these irrelevant source data may reduce the learning performance, thus causing negative transfer. To reduce the negative transfer, the proposed AW-CORAL aims to decrease the weights of unrelated source data, thus weakening the impact of those potentially harmful instances.

AW-CORAL adopts the scheme of TrAdaBoost [21] to iteratively re-weight the instances. In particular, AW-CORAL initializes a weight vector $\mathbf{w}_n$ of each source domain S_n, $n = 1, 2, \ldots, N$. The element in each weight vector is set to 1. At each iteration, AW-CORAL transforms source domains along with weight vectors to the target domain, denoted as $S_{n,p}^{t}$, $n = 1, 2, \ldots, N$, via a modified weighted CORAL. Suppose a source domain is $D_S = (\mathbf{X}_S, \mathbf{y}_S)$, the corresponding instance weight vector is $\mathbf{w}_S$, and a target domain is $D_T = (\mathbf{X}_T, \mathbf{y}_T)$. Compared with TrAdaBoost, AW-CORAL employs the weighted CORAL alignment strategy to reduce the distance between source and target, which will facilitate knowledge transfer [6]. The complete process of the modified weighted CORAL is presented in *Algorithm 3*.

Algorithm 3 Process of Modified Weighted CORAL

Input: $D_S = (\mathbf{X}_S, \mathbf{y}_S)$: The source domain; $\mathbf{w}_S$: The instance weight vector of source domain; $D_T = (\mathbf{X}_T, \mathbf{y}_T)$: The target domain;
Output: $D_{S,p}$: The transformed source data;
$\mathbf{C}_S = \text{cov}(\mathbf{X}_S) + \text{eye}(\text{size}(\mathbf{X}_S, 2))$;
$\mathbf{C}_T = \text{cov}(\mathbf{X}_T) + \text{eye}(\text{size}(\mathbf{X}_T, 2))$;
$\mathbf{X}_S = \mathbf{w}_S \mathbf{X}_S \mathbf{C}_S^{-\frac{1}{2}}$;
$\mathbf{X}_S = \mathbf{X}_S \mathbf{C}_T^{\frac{1}{2}}$;
$D_{S,p} = (\mathbf{X}_S, \mathbf{y}_S)$

Based on the transformed source domains, a source model $F^t_{n,s}$ is learned from each projected source domain $S^t_{n,p}$, $n = 1, 2, \ldots, N$. Accordingly, the weight of each source model is re-calculated and updated. After that, the domain-wise ensemble F^t_{EST} is generated via the weighted combination of the source models $F^t_{1,s}, F^t_{2,s}, \ldots, F^t_{N,s}$ and the class-wise weighted ensemble model F_t by (6).

Once F^t_{EST} is obtained, it is adopted to estimate the weights of source instances because it is more reliable on D_t. Similar to TrAdaBoost, the weights of incorrect predicted source instances are reduced because they are considered to be dissimilar to the target data. In other words, the instance weight vector $\mathbf{w}_n$ of a source domain S_n, $n = 1, 2, \ldots, N$ is updated according to the following:

$$w^i_n = w^i_n \cdot e^{-\beta_n |\hat{y}_i == y_i|} \tag{11}$$

where $\hat{y}_i$ is the predicted label of the i-th instance in the source domain, β_n is a parameter defined as

$$0.5 \ln\left(1 + \sqrt{2 \ln \frac{L_n}{\text{Iter}_{\max}}}\right)$$

L_n is the number of instances of S_n, and $\text{Iter}_{\max}$ is the maximum iterations for termination defined in AW-CORAL.

In this way, the weights of the misclassified source instances are decreased and effects of these instances on the model F^t_{EST} become weaker. The detailed description of AW-CORAL is presented in *Algorithm 4*.

Algorithm 4 Process of AW-CORAL

Input: $S_n = (\mathbf{X}^n_s, \mathbf{y}^n_s)$, $n = 1, 2, \ldots, N$: The N Source domains; K: The number of existing classifiers; L_n, $n = 1, 2, \ldots, N$: The number of instances of each source domain; $Iter_{max}$: The maximum number of iterations; $D_t = (\mathbf{X}_t, \mathbf{y}_t)$: The training data chunk at time step t; L_t: The number of instances in D_t; F_t: The class-wise weighted ensemble classifier of target domain;
Output: F^t_{EST}: The created domain-wise ensemble model at time step t;
Set $\beta_n = 0.5 \ln(1 + \sqrt{2 \ln \frac{L_n}{Iter_{\max}}})$, $n = 1, 2, \ldots, N$;
Initialize the weight vector w_n, $n = 1, 2, \ldots, N$, each element is set to 1;
for $iter = 1 : Iter_{\max}$ **do**
  for $n = 1 : N$ **do**
    Obtain the projected data of each source domain $S^t_{n,p}, n = 1, 2, \ldots, N$ according to **Algorithm 3**;
    Create a classifier $F^t_{n,s}$ on $S^t_{n,p}$, $n = 1, 2, \ldots, N$;
    Evaluate the prediction accuracy of F^s_n on D_t and set its weight to the evaluated accuracy;
  end for
  Generate domain-wise ensemble F^t_{EST} according to Eq. (6);
  for $n = 1 : N$ **do**
    Use F^t_{EST} to predict the instance in $S^t_{n,p}$, $n = 1, 2, \ldots, N$;
    Adjust the weight vector of each source domain $\mathbf{w}_n$, $n = 1, 2, \ldots, N$ using Eq. (11);
  end for
end for

IV. Experiments and Analysis

To evaluate the performance of the proposed HE-CDTL, we conduct extensive experiments to compare HE-CDTL with the existing methods on several synthetic and real-world benchmark data sets. We also investigate the effectiveness of the two components of HE-CDTL, i.e., class-wise weighted ensemble and AW-CORAL. Finally, we discuss the influence of the achieved size.

A. Experimental Settings

We compare the proposed HE-CDTL with three state-of-the-art baselines, i.e., SEA [14], CORAL [7], and Melanie [13]. In particular, we employ performance-based weight strategy for SEA [14] and implement the class-wise weighted strategy in the framework of SEA. We also compare HE-CDTL with CORAL [7], which is a feature mapping approach to transfer learning in order to access the performance of AW-CORAL strategy. We implement CORAL in CDTL by splitting each time step in the target domain as a series of traditional transfer learning problems. Melanie [13], an algorithm for transfer learning in nonstationary environments, is also selected for comparison.

To make the comparisons consistent, we use the Support Vector Machine (SVM) [46] with a cubic polynomial kernel as the base learner for the compared algorithms on CIR and SIN data sets, and the SVM with linear kernel for other data sets. We use the toolboxes [47] to train the SVM with a cubic polynomial kernel. For the SVM with linear kernel, the hinge loss is applied as a loss function and the learning rate is set to 0.001. The maximum iteration for training SVM is set to 100. The maximum size of the archive that stores historical classifiers is set to $K_{\max} = 15$ for SEA and HE-CDTL. The maximum iteration for AW-CORAL is set to $\text{iter}_{\max} = 10$. The parameters of SEA and Melanie are the same to HE-CDTL. CORAL is a parameter-free algorithms. To access performances of the compared algorithms, the widely used prediction accuracy [17], [27], [42] is applied as the performance metric. The prediction accuracy at the tth time step is

TABLE I
PARAMETERS OF ROT

	C	μ_x	μ_y	σ_x	σ_y
$0 < \theta \leq 0.2$	C1	2	5	1	2+5θ
	C2	5-5θ	8	3-10 θ	1
	C3	5-5θ	2	3-10θ	1
	C4	N/A	N/A	N/A	N/A
$0.2 < \theta \leq 0.4$	C1	2	5	1+5θ	5-5θ
	C2	4+2θ	8	5	5
	C3	4+2θ	2	5	5
	C4	N/A	N/A	N/A	N/A
$0.4 < \theta \leq 0.6$	C1	2	5-1.5θ	4-5θ	4-5θ
	C2	8	8-2θ	1	1+5θ
	C3	8-θ	2	1+10θ	1
	C4	5	5+1.5θ	1	1
$0.6 < \theta \leq 0.8$	C1	N/A	N/A	N/A	N/A
	C2	8	4+θ	1+2.5θ	2-2.5θ
	C3	6-2θ	2+3θ	3-1.5θ	1+1.5θ
	C4	5+1.5θ	8-3θ	1+0.5θ	1+1.5θ
$0.8 < \theta \leq 1.0$	C1	N/A	N/A	N/A	N/A
	C2	8	8-3θ	1.5	1.5
	C3	2+3θ	5	1.5	1.5
	C4	8-3θ	2	1.5	1.5

calculated via the following:

$$\text{Prediction Accuracy} = \frac{\sum_{i \in D_t} \widehat{y_i} == y_i}{|D_t|} \tag{12}$$

where D_t is the test data at the tth time step, $\widehat{y_i}$ is the predicted label of the ith test data, y_i is the true label of the i-th test data, and $|D_t|$ is the cardinality of D_t.

B. Benchmark Data Sets

SEA moving hyperplane concepts (SEA) [14], [42], Rotating concepts (ROT) [12], [15], Circle concepts (CIR) [42], [48], and Sine concepts (SIN) [42], [48] are the widely used synthetic benchmark generators for concept drift problems. The real benchmark data sets are generated from Office-31 [49], Caltech-256 [50] and PIE [27], which are popular real benchmark data sets for transfer learning.

1) Synthetic Benchmark Data Sets:

1) *SEA*: SEA generates three features for each data point (x_1, x_2, x_3), where the value of each feature is randomly sampled between 0 and 10. It is a two class classification problem and only the first two features x_1 and x_2 are relevant to the classifier hyperplane. The labels of the two classes $C1$ and $C2$ are defined as

$$\begin{cases} C1 : x_1 + x_2 \leq \theta \\ C2 : x_1 + x_2 > \theta \end{cases} \tag{13}$$

where the value of θ changes during the learning process to simulate concept drift. Similar to [42], the value of θ changes among $10, 7, 3, 10, 13, 16, 13$. To extend SEA as a CDTL, we generate two source data sets at two randomly time steps with each containing 500 samples. We denote the generated data set as T-SEA. Based on T-SEA, we propose a multiclass SEA (M-SEA) and define the class labels in the following:

$$\begin{cases} C1 : \ x_1 + x_2 \leq \theta_1 \\ C2 : \ \theta_1 < x_1 + x_2 \leq \theta_2 \\ C3 : \ x_1 + x_2 > \theta_2 \& x_1 - x_2 > 0 \\ C4 : \ x_1 + x_2 > \theta_2 \& x_1 - x_2 < 0. \end{cases} \tag{14}$$

We sample each feature of the data between 0 and 15. Similar to T-SEA, the values of θ_1 and θ_2 vary with time, where $\theta_1 \in \{5, 5, 7, 7, 10, 10, 13, 13, 16, 16\}$, and $\theta_2 \in \{7, 7, 10, 10, 13, 13, 16, 16, 20, 20\}$. We also swap the label of class 3 and class 4 between each contiguous two time steps. The two source data sets are generated in a similar way to T-SEA. For both T-SEA and M-SEA, 60 chunks with 10% noise introduced are generated for each target domain.

2) *ROT*: ROT generates a more difficult data set with class removing or addition in the learning process. It generates the samples of each class using a two-dimensional Gaussian Function. The distribution of each class changes independently according to the class means $(\mu_x, \ \mu_y)$ and variances $(\sigma_x, \ \sigma_y)$, which vary with a parameter θ given in Table I. A total number of 100 time steps are evenly sampled during $\theta = [0, 1]$, and each data chunk is induced to 10% noise. Two source domains are generated at $0.4 < \theta \leq 0.6$ and $0.8 < \theta \leq 1.0$, respectively. Each source domain consists of 500 samples.

3) *CIR*: CIR generates two features, x_1, x_2, for each the data, where the value of each feature is sampled between -5 and 5. It is a two class problem and employs a circle as the decision boundary. The labels of the two classes C1 and C2 are defined as

$$\begin{cases} C1 : x_1^2 + x_2^2 \leq \theta \\ C2 : x_1^2 + x_2^2 > \theta \end{cases} \tag{15}$$

where $\theta \in \{3, 2, 1, 2, 3, 4, 5, 4\}$ to simulate the concept drift problems. The data of the two source domains are generated with $\theta = 2$ and $\theta = 4$ separately.

4) *SIN*: SIN is also a two-dimensional problem, where the data are located in $[-5, 5]$. It contains two classes and defines the label using a sine curve

$$\begin{cases} C1 : \sin(x_1 + t\theta) \leq x_2 \\ C2 : \sin(x_1 + t\theta) > x_2 \end{cases} \tag{16}$$

where $\theta = \pi/3$. We set $x_1 + t\theta = \pi/6$ and $x_1 + t\theta = 2\pi/3$ to generate the data of the two source domains.

2) Real Benchmark Data Sets:

1) *Office-10+Caltech-10*: Office-10+Caltech-10 is composed of the overlapping ten classes of two data sets, Office-31 [49], [51] and Caltech-256 [50]. Office-31 consists of three domains: Amazon (A), Webcam (W) and DSLR (D). It has 4652 object images classified into 31 categories. Caltech-256 (C) contains 958, 295, 157 images with 256 categories. This experiment uses the shallow features (SURF) of each domain, which are encoded and normalized to 800-bin histograms with bag-of-words. To extend it as a CDTL

TABLE II

STATISTICAL AVERAGED PREDICTION ACCURACY (%)(± THE STANDARD DEVIATION) OF THE COMPARED ALGORITHMS OVER 20 RUNS ON SYNTHETIC DATA SETS, WHERE THE BEST RESULT ON EACH DATA SET IS HIGHLIGHTED

Dataset	Melanie	CORAL	SEA	HE-CDTL
T-SEA	95.842 ± 0.533 =	94.788 ± 0.412 −	95.984 ± 0.631 =	95.857 ± 0.483
M-SEA	69.226 ± 0.594 −	70.041 ± 0.457 −	68.787 ± 0.662 −	**71.054 ± 0.456**
ROT	74.124 ± 0.656 −	71.178 ± 0.651 −	74.537 ± 0.691 −	**76.422 ± 0.576**
CIR	82.676 ± 1.031 −	91.928 ± 0.404 =	90.425 ± 1.113 −	91.787 ± 0.981
SIN	91.722 ± 1.10 −	92.684 ± 0.711 −	92.635 ± 0.952 −	**93.543 ± 0.39**

problem, we select two domains among the four in Office-10+Caltech-10 to construct the target domain and the remaining two as two source domains. In this way, six cross domain tasks can be generated (W, A → (C, D), W, C → (D, A), W, D → (C, A), C, A → (W, D), D, A → (C, W), C, D → (W, A)), where → refers to transfer knowledge from the former domains (i.e. the source tasks) to the later domains (i.e. the target task). To make the target domain as a concept drift problem, we separate the data in each domain into data chunks, and introduce two methods to manipulate these data chunks. The first method randomly disturbs the order of generated data chunks. Consequently, two concepts, sampled from the two domains of the target task, are randomly appeared along with the time steps. In the second method, we randomly select two classes in each data chunk and swap their labels. Data sets generated in this way have a number of various concepts. We denote the two data sets generated from the above methods as ROff+Cal and SOff+Cal, respectively.

2) **PIE**: PIE is short for "Pose, Illumination, Expression," is a data set contains 41 368 face images of 68 individuals. We choose five subsets, (C05, left pose), PIE2 (C07, upward pose), PIE3 (C09, downward pose), PIE4 (C27, frontal pose), PIE5 (C29, right pose) and the first ten classes of each subset for experiments. The face images in each subset differ in lighting, illumination, and expression conditions. The dimension of SURF features of each image is 1024. We randomly select two subsets as two source domains, and the remaining three subsets to construct the target domain. Accordingly, ten cross domain tasks are formulated, i.e., C05, C07 → (C09, C27, C29), C09, C07 → (C05, C27, C29). Two versions of PIE, RPIE and SPIE, are generated, where the target domain is manipulated in the same way as in Office-10+Caltech-10.

In all the data sets, each class in a data chunk contains ten instances for training data, and five data for testing data.

C. Comparison Results on Benchmark Data Sets

1) Results on Synthetic Benchmark Data Sets: To assess the performance of the compared algorithms, the results of averaged prediction accuracy of all data chunks over 20 independent runs are presented in Table II. The Wilcoxon's rank sum test at 0.05 significance level is conducted to check whether the differences among SEA, CORAL, Melanie, and HE-CDTL are statistically significant, whose results are shown in Table II as well. In the table, the best metric values are in bold. Symbols

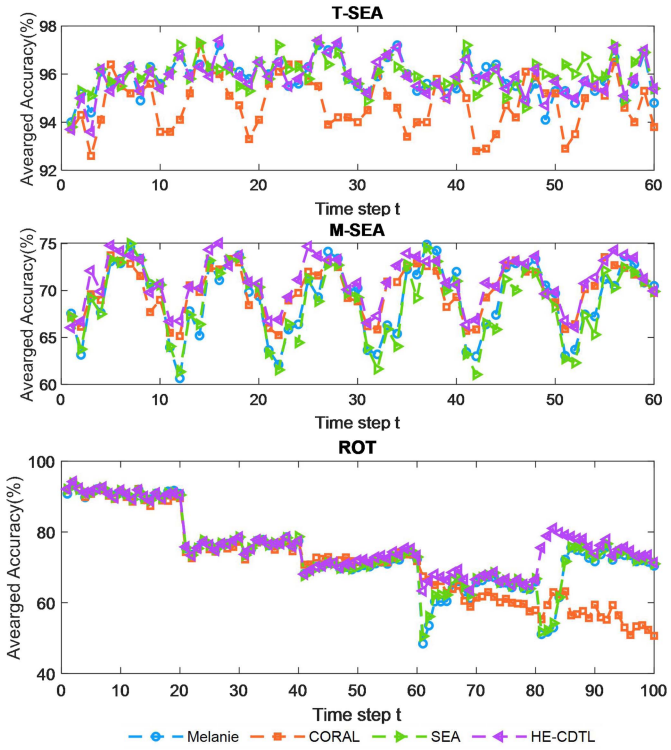


Fig. 3. Averaged prediction accuracy over 20 independent runs of Melanie, CORAL, SEA, and HE-CDTL on three synthetic data sets.

"+", "−", and " =" indicate the corresponding algorithm performs significantly better than, worse than, and similar to the proposed HE-CDTL, respectively.

It is clear from Table II that the five synthetic benchmark data sets in the experiments are in favor of the proposed HE-CDTL. In particular, the experimental results show that HE-CDTL achieves similar performance with Melanie and SEA, and they all outperform CORAL on T-SEA data set. On the CIR data set, CORAL and HE-CDTL perform best compared with other two algorithms. HE-CDTL obtains the best performance over the baselines on M-SEA, ROT, and SIN data sets. The reasons for the superior performance of HE-CDTL are not difficult to understand. On the one hand, the class-wise weighted ensemble in HE-CDTL enhances the historical knowledge of the target domain transfer. On the other hand, the use of adaptive weighted CORAL leads HE-CDTL to increase positive knowledge of source domain transfer as well as avoid harmful knowledge transfer.

To visualize the quality of compared algorithms on each time step, we also list averaged prediction accuracy of T-SEA, M-SEA, and ROT along with the time steps in Fig. 3. It can be seen that: 1) the prediction accuracy of SEA is steadily

TABLE III

STATISTICAL AVERAGED PREDICTION ACCURACY(%) (± THE STANDARD DEVIATION) OF THE COMPARED ALGORITHMS OVER 20 RUNS ON REAL BENCHMARK DATA SETS, WHERE THE BEST RESULT ON EACH DATA SET IS HIGHLIGHTED

	Dataset (Index)	Melanie	CORAL	SEA	HE-CDTL
SOff+Cal	A, D → (W, C) (1)	47.210 ± 0.201 −	47.550 ± 0.221 −	46.815 ± 0.274 −	**50.480 ± 0.258**
	W, D → (A, C) (2)	45.877 ± 0.261 −	43.031 ± 0.157 −	46.354 ± 0.114 −	**48.738 ± 0.160**
	C, D → (A, W) (3)	55.011 ± 0.157 −	57.011 ± 0.210 −	54.057 ± 0.244 −	**60.029 ± 0.207**
	W, A → (C, D) (4)	47.225 ± 0.109 −	45.610 ± 0.251 −	46.395 ± 0.239 −	**49.005 ± 0.220**
	C, A → (D, W) (5)	71.340 ± 0.327 −	**74.620 ± 0.394** +	71.000 ± 0.283 −	73.880 ± 0.270
	C, W → (A, D) (6)	52.451 ± 0.176 −	51.954 ± 0.255 −	51.177 ± 0.235 −	**55.034 ± 0.201**
ROff+Cal	A, D → (W, C) (7)	47.210 ± 0.204 −	44.950 ± 0.339 −	46.370 ± 0.290 −	**48.060 ± 0.279**
	W, D → (A, C) (8)	51.298 ± 0.150 −	41.126 ± 0.159 −	53.175 ± 0.252 −	**53.658 ± 0.067**
	C, D → (A, W) (9)	55.520 ± 0.200 −	48.891 ± 0.160 −	56.429 ± 0.240 −	**57.663 ± 0.273**
	W, A → (C, D) (10)	46.890 ± 0.108 =	40.520 ± 0.244 −	**47.420 ± 0.234** +	46.750 ± 0.170
	C, A → (D, W) (11)	70.600 ± 0.316 −	70.320 ± 0.482 −	69.400 ± 0.327 −	**71.720 ± 0.303**
	C, W → (A, D) (12)	58.057 ± 0.181 −	50.206 ± 0.354 −	**59.623 ± 0.495** +	58.509 ± 0.208
SPIE	C05, C09 → (C07, C27, C29) (1)	39.472 ± 0.101 −	40.384 ± 0.143 −	37.904 ± 0.131 −	**43.140 ± 0.137**
	C05, C07 → (C09, C27, C29) (2)	37.872 ± 0.161 −	40.320 ± 0.150 −	37.712 ± 0.201 −	**42.060 ± 0.165**
	C05, C27 → (C07, C09, C29) (3)	29.280 ± 0.157 −	30.320 ± 0.152 −	27.440 ± 0.174 −	**31.100 ± 0.351**
	C05, C29 → (C07, C27, C09) (4)	31.008 ± 0.108 −	**34.064 ± 0.067** +	31.312 ± 0.072 −	31.840 ± 0.146
	C07, C09 → (C05, C27, C29) (5)	33.794 ± 0.118 −	32.080 ± 0.143 −	33.017 ± 0.226 −	**34.471 ± 0.055**
	C07, C27 → (C05, C09, C29) (6)	35.336 ± 0.151 −	36.144 ± 0.067 −	34.096 ± 0.104 −	**37.500 ± 0.137**
	C09, C27 → (C05, C07, C29) (7)	28.664 ± 0.093 −	**30.672 ± 0.121** +	27.712 ± 0.091 −	29.720 ± 0.240
	C09, C29 → (C05, C09, C27) (8)	33.291 ± 0.155 −	32.697 ± 0.102 −	31.646 ± 0.322 −	**38.271 ± 0.072**
	C27, C29 → (C05, C07, C09) (9)	27.456 ± 0.150 −	29.648 ± 0.134 −	27.648 ± 0.107 −	**33.780 ± 0.077**
	C07, C29 → (C05, C09, C27) (10)	36.920 ± 0.122 −	35.634 ± 0.192 −	36.971 ± 0.265 −	**38.714 ± 0.411**
RPIE	C05, C09 → (C07, C27, C29) (11)	41.897 ± 0.096 −	40.171 ± 0.134 −	43.863 ± 0.065 −	**44.040 ± 0.191**
	C05, C07 → (C09, C27, C29) (12)	35.776 ± 0.135 −	35.280 ± 0.150 −	36.832 ± 0.121 −	**42.408 ± 0.116**
	C05, C27 → (C07, C09, C29) (13)	34.387 ± 0.172 −	31.547 ± 0.223 −	32.560 ± 0.219 −	**34.867 ± 0.126**
	C05, C29 → (C07, C27, C09) (14)	41.184 ± 0.098 +	35.984 ± 0.104 −	**42.032 ± 0.156** +	40.576 ± 0.112
	C07, C09 → (C05, C27, C29) (15)	37.120 ± 0.082 −	33.074 ± 0.102 −	36.057 ± 0.167 −	**41.091 ± 0.125**
	C07, C27 → (C05, C09, C29) (16)	34.592 ± 0.094 −	32.656 ± 0.131 −	35.728 ± 0.091 −	**42.560 ± 0.160**
	C09, C27 → (C05, C07, C29) (17)	39.680 ± 0.065 =	34.752 ± 0.072 −	39.632 ± 0.107 =	39.584 ± 0.309
	C09, C29 → (C05, C09, C27) (18)	37.366 ± 0.102 −	32.606 ± 0.065 −	38.423 ± 0.051 −	**39.874 ± 0.080**
	C27, C29 → (C05, C07, C09) (19)	39.488 ± 0.077 =	37.216 ± 0.036 −	38.304 ± 0.182 −	39.624 ± 0.222
	C07, C29 → (C05, C09, C27) (20)	39.937 ± 0.063 −	32.206 ± 0.096 −	41.714 ± 0.114 −	**42.429 ± 0.268**
	Average	42.600 ± 0.147	40.726 ± 0.175	42.463 ± 0.193	45.037 ± 0.191

higher than CORAL on each time step on T-SEA, showing that the knowledge of the previous time step in the target domain contributes more to the learning performance than the source domains. On the contrary, for M-SEA, the source domain knowledge is more helpful than historical knowledge of the target domain since CORAL outperforms SEA. However, the performance of HE-CDTL is better than CORAL and SEA on both of these two data sets. This observation confirms that for the given knowledge, HE-CDTL is able to automatically select helpful knowledge for learning; 2) the four compared algorithms show a very different behavior on ROT with the one on T-SEA and M-SEA. This is because the concepts in ROT data set are very complex, which is unlike T-SEA and M-SEA with rotated drifts. Such characters of ROT cause the performance to degrade at time steps of removing classes, adding new classes, or overlapping classes, i.e., $t = 21$, $t = 41$, $t = 61$, $t = 81$. An interesting observation can also be found in the figure, i.e., the behavior of HE-CDTL is very distinct and the accuracy dramatically increases at $t = 61$ and $t = 81$. The underlying reason may be attributed to the class-wise weighted ensemble technique; and 3) HE-CDTL achieves superior performance than Melanie at most time steps of the three data sets. It shows that the benefit knowledge of source domains can be increased after the adaptive weighted CORAL has been performed.

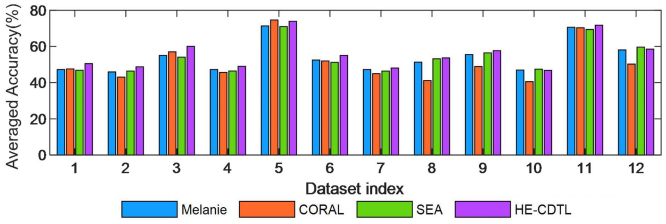


Fig. 4. Averaged prediction accuracy(%) over 20 independent runs of Melanie, CORAL, SEA, and HE-CDTL on ROff+Cal, SOff+Cal data sets.

2) Results on Real Benchmark Data Sets: The statistical classification accuracies over 20 independent runs of the four compared algorithms on 32 cross-domain (ROff+Cal, SOff+Cal, RPIE, SPIE) real benchmark data sets are listed in Table III. The results of ROff+Cal, SOff+Cal, RPIE, and SPIE are plotted in Figs. 4 and 5 for better visualization. It can be observed that HE-CDTL achieves the highest accuracy as compared to Melanie, CORAL, and SEA on most cross-domain tasks (24 out of 32). The averaged prediction accuracy of HE-CDTL over 32 cross-domain data sets is 45.037%, gaining a significant performance improvement of 2.437% compared to Melanie. This result indicates that, without feature matching and controlling of negative transfer,

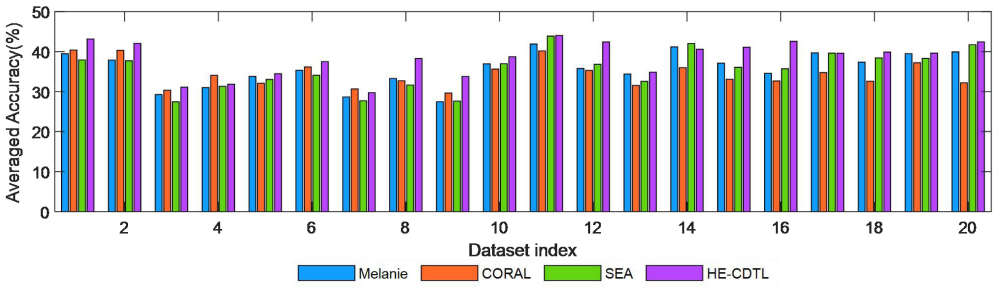


Fig. 5. Averaged prediction accuracy(%) over 20 independent runs of Melanie, CORAL, SEA, and HE-CDTL on RPIE, SPIE data sets.

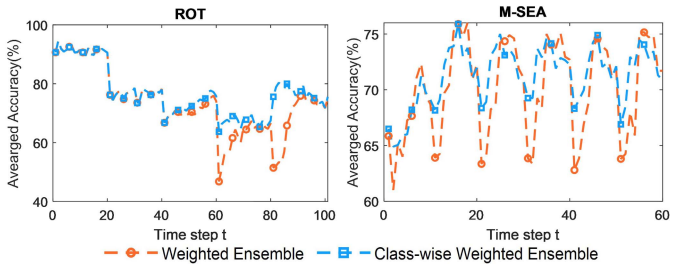


Fig. 6. Averaged prediction accuracy over 20 independent runs of Class-wise weighted ensemble and weighted ensemble on M-SEA and ROT data sets.

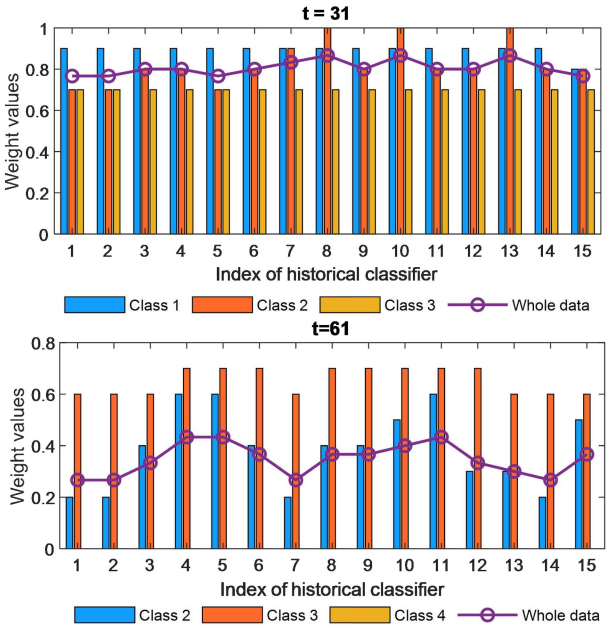


Fig. 7. Weights of historical classifiers on each class and the whole data chunk when $t = 31$ and $t = 61$.

Melanie is not efficient in transferring knowledge in comparison with HE-CDTL.

Also, it can be observed that COARL performs better than SEA on SOff+Cal and SPIE data sets that are generated by switching class labels, while performs poorly on ROff+Cal and RPIE that are generated by disturbing the order of data chunks. A plausible reason is that the concepts on SOff+Cal, SPIE data sets are randomly drift, leading to less knowledge of historical time steps to be used. In this case, SEA can leverage little knowledge, while COARL is able to preserve a stable knowledge from source domains via domain discrepancy. On the contrary, in the target domain of ROff+Cal and RPIE data sets, a large number of previous data chunks are similar to the one in the current time step because only two concepts are rotated. The similar data chunks provide more helpful knowledge than source domains, thus leading SEA to achieve better performance.

D. Discussions and Analysis

1) Effectiveness of Class-Wise Weighted Ensemble: To access the effectiveness of the proposed class-wise weighted ensemble strategy, we apply the class-wise weighted ensemble to handle the concept drift problems and compare it with the corresponding weighted ensemble approach. We conduct the experiments on the two synthetic data sets: M-SEA and ROT, without considering the source domains. The averaged prediction accuracies of the two algorithms on each data chunk are plotted in Fig. 6. It can be observed that the class-wise weighted ensemble performs much better than the weighted ensemble on M-SEA. This is because the distributions of two classes in M-SEA are rotated at two continuous time steps, while the other two classes gradually change with a parameter θ. A classifier learned from a historical data chunk may perform well or poorly on the two rotated classes, and performs averagely on the other two classes. In such a situation, it is difficult for the weighted ensemble to balance the transferred historical knowledge among the four classes with a single global weight. However, the class-wise weighted ensemble allows each class to obtain historical knowledge independently via different weights, hence achieving superior performance.

On the ROT data set, the class-wise weighted ensemble and the weighted ensemble show a similar performance before $t = 60$. However, the accuracy of class-wise weighted ensemble is significantly higher than that of weighted ensemble after $t = 60$, especially when $61 \leq t \leq 70$, $81 \leq t \leq 90$. To analyze the behavior of weighted ensemble on ROT, we list the values of weight vectors of the class-wise weighted ensembles and the global weights of weighted ensembles of each archived classifier at $t = 31$ and $t = 61$ in Fig. 7. We can see that, given a historical classifier, its global weight (the weight is equal to the prediction accuracy) is generally equal to the

TABLE IV
STATISTICAL PREDICTION ACCURACY (%) OF AW-CORAL AND TRADABOOT ON THREE SYNTHETIC DATA SETS AND 32 REAL-WORLD DATA SETS

Dataset	T-SEA	M-SEA	ROT		win	tie	loss
Tradaboot	91.9	90.1	65.0		3	0	0
AW-CORAL	93.7	92.1	66.0				
Dataset	A, D → W	A, D → C	W, D → A	W, D → C	win	tie	loss
Tradaboot	73.31	34.82	33.12	29.48	3	1	0
AW-CORAL	77.54	40.60	33.38	31.70			
Dataset	C, D → A	C, D → W	A, W → C	A, W → D	win	tie	loss
Tradaboot	44.07	75.00	76.98	34.82	3	1	0
AW-CORAL	45.89	75.00	78.57	42.60			
Dataset	A, C → D	A, C → W	C, W → A	C, W → D	win	tie	loss
Tradaboot	42.86	31.78	44.07	74.60	1	1	2
AW-CORAL	40.48	29.66	44.18	78.57			
Dataset	(C07, C27, C29)→C05	(C07, C27, C29)→C09	(C09, C27, C29)→C05	(C09, C27, C29)→C07	win	tie	loss
Tradaboot	29.34	34.37	29.59	34.90	1	3	0
AW-CORAL	29.59	37.50	29.83	34.85			
Dataset	(C07, C27, C09)→C05	(C07, C27, C09)→C29	(C07, C27, C09) → C05	(C07, C27, C09) → C29	win	tie	loss
Tradaboot	29.34	34.69	29.59	34.90	3	1	0
AW-CORAL	30.10	34.69	32.40	35.94			
Dataset	(C05, C27, C29)→C07	(C05, C27, C29)→C09	(C05, C09, C29)→C07	(C05, C09, C29)→C27	win	tie	loss
Tradaboot	42.19	39.06	35.94	32.40	3	1	0
AW-CORAL	45.83	45.83	39.06	32.65			
Dataset	(C05, C07, C29)→C09	(C05, C07, C29)→C27	(C05, C09, C27)→C09	(C05, C09, C27)→C29	win	tie	loss
Tradaboot	38.02	38.78	40.63	33.33	4	0	0
AW-CORAL	46.88	44.64	45.31	36.98			
Dataset	(C05, C07, C09)→C27	(C05, C07, C09)→C29	(C05, C09, C27)→C07	(C05, C09, C27)→C29	win	tie	loss
Tradaboot	38.02	35.20	38.54	42.71	3	0	1
AW-CORAL	37.33	43.37	40.10	50.52			
Total					24	8	3

averaged value of the weight vector. When $t = 31$, the values in a weight vector are relatively similar and they are around the global weight. This means that each previous classifier contributes about equally to the three classes of the current time. However, when $t = 61$, the elements in a weight vector are very different. In particular, the element associated with class 4 is zero because it is a new added class, while the one with class 3 is relatively higher. In such a situation, in the weighted ensemble, the knowledge of an archived classifier cannot sufficiently be explored for class 3, while it may induce negative knowledge for class 4.

We also complementary compare the proposed C-SEA with Learn++.NSE [12] and DTEL [42] on the synthetic concept drift benchmark data sets to demonstrate the superiority of C-SEA. In this experiment, we use the gradual and sudden drift of T-SEA, CIR, and SIN as the test benchmark data sets, denoted as T-SEAG, T-SEAA, M-SEAG, M-SEAA, ROTG, ROTA, CIRG, CIRA, SING, and SINA. The settings of the gradual and sudden drift of T-SEA, CIR, and SIN are the same as SEA, CIR, and SIN in [42]. M-SEA and ROT are also applied as test data sets, the settings of which are the same as in Section IV. Similar to [12], [42], we set two different sizes of data chunks, 80 and 200, for each data set. We employ the Iterative Dichotomiser 3 (ID3) [52] from MATLAB 2019a as the base learner for the three compared algorithm. The compared results in Table V show that C-SEA wins 8 to the other two algorithms among 13 data sets.

TABLE V
STATISTICAL AVERAGED PREDICTION ACCURACY (%)(± THE STANDARD DEVIATION) OF THE COMPARED ALGORITHMS OVER 20 RUNS ON SYNTHETIC DATA SETS, WHERE THE BEST RESULT ON EACH DATA SET IS HIGHLIGHTED

Dataset	SEA	Learn++.NSE	DTEL	C-SEA
T-SEAA80	75.82 ± 1.02	79.39 ± 1.07	78.36 ± 1.31	**81.77 ± 1.32**
T-SEAA200	76.20 ± 0.81	80.87 ± 0.59	78.82 ± 0.95	**82.91 ± 0.76**
T-SEAG80	81.49 ± 0.95	83.98 ± 0.68	78.36 ± 1.31	**85.40 ± 0.95**
M-SEA80	55.11 ± 0.94	74.13 ± 0.82	75.50 ± 1.03	**79.90 ± 0.67**
M-SEA200	57.40 ± 0.80	79.32 ± 0.52	80.95 ± 1.46	**86.74 ± 0.68**
ROT80	75.52 ± 0.40	76.14 ± 0.38	77.45 ± 1.02	**78.54 ± 0.50**
ROT200	75.74 ± 0.34	76.98 ± 0.30	76.98 ± 0.53	**78.71 ± 0.43**
CIRA80	94.76 ± 0.76	94.80 ± 0.67	94.54 ± 0.71	**95.10 ± 0.74**
CIRA200	95.17 ± 0.35	95.53 ± 0.28	95.73 ± 0.27	95.56 ± 0.35
CIRG80	95.11 ± 0.69	95.07 ± 0.56	94.54 ± 0.71	95.41 ± 0.65
SINA80	93.41 ± 0.42	93.38 ± 0.39	93.47 ± 0.38	93.43 ± 0.42
SINA200	93.54 ± 0.29	93.72 ± 0.34	93.54 ± 0.32	93.61 ± 0.30
SING80	93.51 ± 0.44	93.47 ± 0.41	93.47 ± 0.38	93.51 ± 0.44
win-tie-loss		8-5-0	8-5-0	8-5-0

2) Effectiveness of AW-CORAL: We first investigate the effectiveness of AW-CORAL in comparison with TrAdaBoost on both Synthetic and real data sets. The compared results in Table IV show that AW-CORAL wins 24 times, ties 8 times, and losses three times to Tradaboot among 35 data sets.

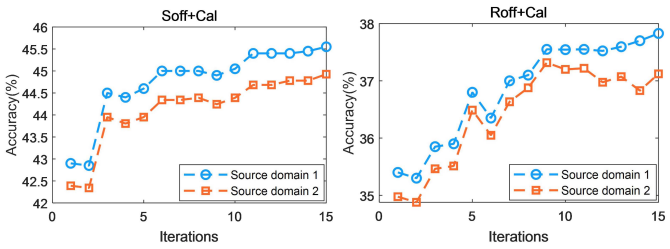


Fig. 8. Prediction accuracy(%) along with the iteration of AW-CORAL.

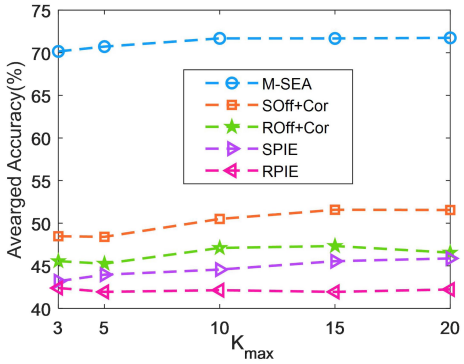


Fig. 9. Averaged prediction accuracy over 20 independent runs of HE-CDTL with different $K_{\max}$ on five cross-domain tasks.

We further evaluate the proposed HE-CDTL on SOFF+Cor (A, D $\rightarrow$ (W, C)), ROFF+Cor (A, D $\rightarrow$ (W, C)) to investigate the effectiveness of the AW-CORAL. We set the maximum iteration of AW-CORAL to 15 and record the averaged accuracy for each source domain at every iteration. The accuracy for each source domain is evaluated by the weighted combination of source and target models. The prediction accuracies along with the iterations are plotted in Fig. 8. It can be observed that the accuracies of the two source domains generally improve with the increasing of iterations. In other words, after the third iteration, the accuracy at each iteration is higher than the first iteration. This observation justifies that AW-CORAL can gradually weaken the importance of irrelevant source instances via adaptive instance re-weighting. Besides, it can be seen that the maximum adaptive iteration is recommended to be larger than 10 in order to obtain a robust and good performance.

3) Influence of the Archive Size: The parameter $K_{\max}$ in HE-CDTL determines the number of historical classifiers, thus influencing the stored historical knowledge and the performance of HE-CDTL. Here, we study the influence of $K_{\max}$ on five data sets, M-SEA, SOFF+Cor (A, D $\rightarrow$ (W, C)), ROFF+Cor(A, D $\rightarrow$ (W, C)), SPIE (C05, C09 $\rightarrow$ (C07, C27, C29)), RPIE (C05, C09 $\rightarrow$ (C07, C27, C29)). The prediction accuracies of HE-CDTL with $K_{\max} = 3, 5, 10, 15, 20$ are presented in Fig. 9. It shows that when $K_{\max}$ is smaller than 10, the averaged accuracies of the five data sets generally improve as $K_{\max}$ is increased. However, the accuracies on M-SEA, ROFF+Cor, and RPIE do not much change after $K_{\max} = 10$. A plausible reason is that these three data sets contain only very few concepts, thus a small number of historical classifiers can afford enough historical knowledge. However, on the SOFF+Cor and SPIE data sets, HE-CDTL achieves the highest accuracy around $K_{\max} = 15$. Hence, by a rule of thumb, we recommend the maximum size of achieve as 15.

V. Conclusion

This article has proposed the HE-CDTL approach to CDTL. It aims to leverage knowledge from both source domains and historical time steps in the target domain to enhance the learning performance in the learning process. The advantages of HE-CDTL lie in two aspects, i.e. class-wise weighted ensemble for leveraging historical knowledge, and AW-CORAL for extracting knowledge from source domains. The class-wise weighted ensemble enables each class in the current learning to select historical knowledge independently. The proposed AW-CORAL can minimize domain discrepancy across source and target domains, meanwhile reducing the negative knowledge transfer. Extensive experimental results have shown that the proposed HE-CDTL significantly outperforms the baselines for handling transfer learning under concept drift problems.

In the future, we would like to conduct a theoretical analysis of HE-CDTL. Furthermore, the proposed class-wise weighted ensemble strategy is one way of locally preserving historical knowledge. Other methods for leveraging previous knowledge in a local manner can be explored as well.

References

[1] S. Jialin Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[2] F. Zhuang *et al.*, "A comprehensive survey on transfer learning," 2019, *arXiv:1911.02685*. [Online]. Available: http://arxiv.org/abs/1911.02685
[3] Z. Wang, Z. Dai, B. Poczos, and J. Carbonell, "Characterizing and avoiding negative transfer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11293–11302.
[4] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*, 2004, p. 114.
[5] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, "Sample selection bias correction theory," in *Proc. Int. Conf. Algorithmic Learn. Theory*. Berlin, Germany: Springer, 2008, pp. 38–53.
[6] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.
[7] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 2058–2065.
[8] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary environments: A survey," *IEEE Comput. Intell. Mag.*, vol. 10, no. 4, pp. 12–25, Nov. 2015.
[9] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 12, pp. 2346–2363, Dec. 2019.
[10] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Woźniak, "Ensemble learning for data stream analysis: A survey," *Inf. Fusion*, vol. 37, pp. 132–156, Sep. 2017.
[11] H. M. Gomes, J. P. Barddal, F. Enembreck, and A. Bifet, "A survey on ensemble learning for data stream classification," *ACM Comput. Surv.*, vol. 50, no. 2, p. 23, Jun. 2017.
[12] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Trans. Neural Netw.*, vol. 22, no. 10, pp. 1517–1531, Oct. 2011.
[13] H. Du, L. L. Minku, and H. Zhou, "Multi-source transfer learning for non-stationary environments," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.
[14] W. N. Street and Y. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2001, pp. 377–382.

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

[16] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, no. 1, p. 9, 2016.

[17] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer joint matching for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1410–1417.

[18] J. Huang, A. Gretton, K. Borgwardt, B. Schölkopf, and A. J. Smola, "Correcting sample selection bias by unlabeled data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 601–608.

[19] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1433–1440.

[20] Q. Sun, R. Chattopadhyay, S. Panchanathan, and J. Ye, "A two-stage weighting framework for multi-source domain adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 505–513.

[21] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, 2007, pp. 193–200.

[22] Y. Freund *et al.*, "Experiments with a new boosting algorithm," in *Proc. Int. Conf. Mach. Learn.*, vol. 96, 1996, pp. 148–156.

[23] Y. Yao and G. Doretto, "Boosting for transfer learning with multiple sources," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1855–1862.

[24] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Scholkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, Jul. 2006.

[25] Y. Zhu *et al.*, "Deep subdomain adaptation network for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 13, 2020, doi: 10.1109/TNNLS.2020.2988928.

[26] M. Long, Y. Cao, Z. Cao, J. Wang, and M. I. Jordan, "Transferable representation learning with deep adaptation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 12, pp. 3071–3085, Dec. 2019.

[27] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2200–2207.

[28] J. Wang, Y. Chen, S. Hao, W. Feng, and Z. Shen, "Balanced distribution adaptation for transfer learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2017, pp. 1129–1134.

[29] J. Wang, Y. Chen, W. Feng, H. Yu, M. Huang, and Q. Yang, "Transfer learning with dynamic distribution adaptation," *ACM Trans. Intell. Syst. Technol.*, vol. 11, no. 1, pp. 1–25, Feb. 2020.

[30] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2960–2967.

[31] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," *London, Edinburgh, Dublin Phil. Mag. J. Sci.*, vol. 2, no. 11, pp. 559–572, Nov. 1901.

[32] B. Sun and K. Saenko, "Subspace distribution alignment for unsupervised domain adaptation," in *Proc. Brit. Mach. Vis. Conf.*, 2015, pp. 1–24.

[33] M. M. Rahman, C. Fookes, M. Baktashmotlagh, and S. Sridharan, "Correlation-aware adversarial domain adaptation and generalization," *Pattern Recognit.*, vol. 100, Apr. 2020, Art. no. 107124.

[34] E. Zhong *et al.*, "Cross domain distribution adaptation via kernel mapping," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2009, pp. 1027–1036.

[35] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2724–2732.

[36] Z. Cao, K. You, M. Long, J. Wang, and Q. Yang, "Learning to transfer examples for partial domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2985–2994.

[37] L. Li, Z. Wan, and H. He, "Dual alignment for partial domain adaptation," *IEEE Trans. Cybern.*, early access, Apr. 29, 2020, doi: 10.1109/TCYB.2020.2983337.

[38] A. Liu, J. Lu, and G. Zhang, "Diverse instance-weighting ensemble based on region drift disagreement for concept drift adaptation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 293–307, Jan. 2021.

[39] Y. Lu, Y.-M. Cheung, and Y. Yan Tang, "Adaptive chunk-based dynamic weighted majority for imbalanced data streams with concept drift," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 8, pp. 2764–2778, Aug. 2020.

[40] H. Wang, W. Fan, P. S. Yu, and J. Han, "Mining concept-drifting data streams using ensemble classifiers," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2003, pp. 226–235.

[41] D. Brzeziński and J. Stefanowski, "Accuracy updated ensemble for data streams with concept drift," in *Proc. Int. Conf. Hybrid Artif. Intell. Syst.* Berlin, Germany: Springer, 2011, pp. 155–163.

[42] Y. Sun, K. Tang, Z. Zhu, and X. Yao, "Concept drift adaptation by exploiting historical knowledge," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4822–4832, Oct. 2018.

[43] Y. George Udny, "VII. On the association of attributes in statistics: With illustrations from the material of the childhood society, &c," *Philos. Trans. Roy. Soc. London. A*, vol. 194, nos. 252–261, pp. 257–319, 1900.

[44] J. Hoffman, M. Mohri, and N. Zhang, "Algorithms and theory for multiple-source adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 8246–8256.

[45] Y. Zhu, F. Zhuang, and D. Wang, "Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 5989–5996.

[46] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[47] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.

[48] L. L. Minku, A. P. White, and X. Yao, "The impact of diversity on online ensemble learning in the presence of concept drift," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 5, pp. 730–742, May 2010.

[49] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2010, pp. 213–226.

[50] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category data set," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. 7694, 2007.

[51] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2066–2073.

[52] M. Slocum, "Decision making using ID3 algorithm," *InSight: Rivier Academic J.*, vol. 8, no. 2, pp. 1–12, 2012.

Cuie Yang received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 2019.

She is currently a Post-doctoral Research Fellow with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. Her research interests include transfer learning, concept drift learning, multitasking evolutionary optimization, and data-driven evolutionary optimization.

Yiu-ming Cheung (Fellow, IEEE) received the Ph.D. degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

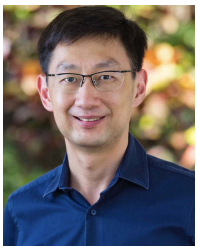
He is currently a Full Professor with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. His research interests include machine learning, computer vision, pattern recognition, data mining, multiobjective optimization, and information hiding.

Dr. Cheung is a fellow of IET, BCS, RSA, and IETI Distinguished Fellow. He serves as an Associate Editor for the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE, the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS from 2014 to 2020, *Pattern Recognition*, and *Neurocomputing*.

Jinliang Ding (Senior Member, IEEE) received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 2012.

He is currently a Professor with the State Key Laboratory of Synthetical Automation for Process Industry, Northeastern University. He has authored or coauthored over 100 refereed journal articles and refereed articles at international conferences. He is also the inventor or co-inventor of 17 patents. His current research interests include modeling, plant-wide control and optimization for the complex industrial systems, stochastic distribution control, and multiobjective evolutionary algorithms and its application.

Dr. Ding was a recipient of the Young Scholars Science and Technology Award of China in 2016, the National Science Fund for Distinguished Young Scholars in 2015, the National Technological Invention Award in 2013, two First-Prize of Science and Technology Award of the Ministry of Education in 2006 and 2012, respectively, and the IFAC Control Engineering Practice 2011–2013 Paper Prize.



Kay Chen Tan (Fellow, IEEE) received the B.Eng. degree (Hons.) and the Ph.D. degree from the University of Glasgow, Glasgow, U.K., in 1994 and 1997, respectively.

He was the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, and currently serves on the Editorial Board member for 10+ journals. He is currently a Chair Professor with the Department of Computing, The Hong Kong Polytechnic University. He is currently the Vice-President (Publications) of the IEEE Computational Intelligence Society, an IEEE Distinguished Lecturer Program (DLP) speaker, an Honorary Professor at the University of Nottingham, U.K., and the Chief Coeditor of Springer Book Series on *Machine Learning: Foundations, Methodologies, and Applications*.