Concept Drift Adaptation by Exploiting Historical Knowledge

Yu Sun, *Student Member, IEEE*, Ke Tang, *Senior Member, IEEE*, Zexuan Zhu, and Xin Yao, *Fellow, IEEE*

***Abstract*— Incremental learning with concept drift has often been tackled by ensemble methods, where models built in the past can be retrained to attain new models for the current data. Two design questions need to be addressed in developing ensemble methods for incremental learning with concept drift, i.e., which historical (i.e., previously trained) models should be preserved and how to utilize them. A novel ensemble learning method, namely, Diversity and Transfer-based Ensemble Learning (DTEL), is proposed in this paper. Given newly arrived data, DTEL uses each preserved historical model as an initial model and further trains it with the new data via transfer learning. Furthermore, DTEL preserves a diverse set of historical models, rather than a set of historical models that are merely accurate in terms of classification accuracy. Empirical studies on 15 synthetic data streams and 5 real-world data streams (all with concept drifts) demonstrate that DTEL can handle concept drift more effectively than 4 other state-of-the-art methods.**

***Index Terms*— Concept drift, data stream mining, ensemble learning, incremental learning, transfer learning.**

I. Introduction

MACHINE learning tasks for which training data are available continuously in time have attracted growing attentions due to their wide existence in real-world scenarios, e.g., medical informatics [1], financial data analysis [2], social networks [3], and incremental learning, which updates learning machines (models) when a chunk of new training data arrives, is a major learning paradigm for tackling such tasks. In particular, the learning machines should be updated without access to previous data, such that there is no need to store or reprocess the previous data [4], [5].

Assuming a number of data chunks $D_1 \cdots D_t$ are available sequentially, the incremental learning procedure is composed of t subtasks, each of which can be regarded as a traditional learning task with a distinct data chunk as the training data. Although these subtasks can be tackled independently, i.e., training a model (e.g., a classifier) from scratch for each subtask, it is natural to ask whether the knowledge gained in one subtask can be leveraged to benefit solving future subtasks. Ensemble methods [4], [6] offer a natural approach to incremental learning, as each model obtained during the course of incremental learning could be preserved as a base learner and be utilized for solving future subtasks. It can be observed from the literature [7] that ensemble methods have been used frequently in many advanced incremental learning algorithms and have achieved great successes.

If all the data chunks are generated from the same underlying distribution, ensemble methods for incremental learning are not much different from those for traditional batch learning, i.e., different training data are fed to different base learners. However, the underlying distributions may be nonstationary in real-world applications, since the environment, where data are generated from, may change over time. For example, in the click prediction task of a news website, a breaking news may attract more attention from the visitors and the links of this news are more likely to be clicked. This phenomenon, referred to as *concept drift*, is one of the key challenges that incremental learning approaches [8] [9], including those based on ensembles, need to deal with. Specifically, two research questions need to be answered when designing an ensemble method for incremental learning with concept drift, i.e., which historical (i.e., previously trained) models should be preserved for future use and how to exploit these models to facilitate future learning with concept drift.

To address the above two questions, this paper first reviews the latest progresses on ensemble methods for incremental learning, and then proposes a Diversity and Transfer-based Ensemble Learning approach (DTEL). DTEL employs a decision tree as the base learner and a diversity-based strategy for preserving historical models. When a new data chunk arrives, the preserved models are exploited as "initial points" for searching/training new models. Finally, the newly obtained models are combined to form the new ensemble.

The rest of this paper is organized as follows. Section II introduces the notations and formal definitions of the learning problem considered in this paper and reviews related work. The DTEL approach is presented in Section III. Section IV reports our empirical studies on DTEL and other state-of-the-art methods. Section V concludes this paper with directions for the future work.

Manuscript received August 8, 2016; revised April 23, 2017; accepted November 4, 2017. This work was supported in part by the Ministry of Science and Technology of China under Grant 2017YFC0804002, in part by the National Natural Science Foundation of China under Grant 61672478 and Grant 61329302, in part by the Science and Technology Innovation Committee Foundation of Shenzhen under Grant ZDSYS201703031748284, and in part by the Royal Society Newton Advanced Fellowship under Grant NA150123. *(Corresponding author: Ke Tang.)*

Y. Sun and K. Tang are with the Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China, and also with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: sunyu123@mail.ustc.edu.cn; tangk3@sustc.edu.cn).

Z. Zhu and X. Yao are with the Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: zhuzx@szu.edu.cn; xiny@sustc.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2017.2775225

II. Related Work

A. Basic Concepts and Notations

In incremental learning, at each time step t, a chunk of data $D_t = \{(\mathbf{x}_1^t, y_1^t), (\mathbf{x}_2^t, y_2^t), \ldots, (\mathbf{x}_n^t, y_n^t)\}$, generated from distribution $p_t(\mathbf{x}, y)$, is received, where $\mathbf{x}_i^t$ is a vector of attribute values and y_i^t is a class label. Concept drift can thus be defined as the change of the underlying distribution, e.g., $p_t(\mathbf{x}, y) \neq p_{t-1}(\mathbf{x}, y)$. It should be noted that a special case of a data chunk is a single data example, which is more often referred to as online learning [10].

At each time step t, the learning goal of incremental learning is similar to that of batch learning, i.e., to obtain a good model F_t for $p_t(\mathbf{x}, y)$, which can be stated as

$$F_t = \arg\min_{f \in \mathcal{H}} \mathbb{E}_{(\mathbf{x},y)\in p_t(\mathbf{x},y)}[\ell(f(\mathbf{x}), y)] \tag{1}$$

where $\mathcal{H}$ is the hypothesis set, $\mathbb{E}(\cdot)$ denotes the expected value of a random variable, and $\ell(\cdot,\cdot)$ is the loss function. Considering a sequence of $p_t(\mathbf{x}, y)$, the goal of the whole incremental learning process is given as

$$\min_{F_1, F_2, \ldots, F_t, \ldots} \sum_t \mathbb{E}_{(\mathbf{x},y)\in p_t(\mathbf{x},y)}[\ell(F_t(\mathbf{x}), y)]. \tag{2}$$

From (2), it can be observed that at least t models will be generated during the course of incremental learning, which sets a natural basis for ensemble learning.

B. Concept Drift Handling Techniques

There exist several strategies for handling concept drift in incremental learning, including a sliding window, concept drift detection, and ensemble methods. The sliding window methods [11]–[13], which are mainly applied in the online learning scenario, preserve part of the most recently arrived data and update the current model with both the preserved data and the newly arrived training example. Some other methods [14]–[16] explicitly involve a concept drift detection module in the learning algorithm. If no concept drift is detected, the current model is updated with newly arrived data. Otherwise, the current model, which may be either a single learner [14] or an ensemble [16], is discarded and a new model is built from scratch.

Neither of the above two strategies requires preserving knowledge/models obtained previously. However, one can easily imagine cases, where preserving historical models would be beneficial. For example, if a previously occurred concept appears later [i.e., $p_t(\mathbf{x}, y) \neq p_{t-1}(\mathbf{x}, y)$ and $p_{t+1}(\mathbf{x}, y) = p_{t-1}(\mathbf{x}, y)$], a historical model could be used directly. In a more general case, if two concepts are correlated but do not appear consecutively in time, it might be easier to adapt a historical model than further training the current model or building a new one. For such reasons, ensemble methods, which preserve historical models, are gaining more popularity in recent years [6], [7], [17].

This paper focuses on incremental learning approaches that preserve historical models and exploit them to form ensembles. Typical examples of such approaches include the Streaming Ensemble Algorithm (SEA) [6], the temporal inductive transfer (TIX) approach [18], the dynamic integration of classifiers (DIC) approach [19], the Learn++ algorithm [4] in Nonstationary Environments (Learn^{++}.NSE [17]), and Accuracy Updated Ensemble (AUE2) [7]. Although the idea of ensembles is also adopted in the Diversity for Dealing with Drifts (DDD) method [16], we distinguish it from the above-mentioned ensemble methods as DDD does not preserve historical models and the ensembles used in that context could be regarded as a single model for time step t.

Due to the concept drift, it is clear that historical models may introduce both positive and negative effects to learning the current concept. Hence, a key issue for all the above ensemble methods is how to get benefits from historical models while preventing negative effects. Suppose a new chunk of data has arrived at time step t and a pool of historical models trained in some previous time steps are available. SEA, DIC, AUE2, and Learn^{++}.NSE exploit the historical model in a similar way. That is, the outputs of the historical models are combined with the output of a new model built using D_t to form final decisions on testing examples of the current concept. These methods differ only in the way that the outputs are combined.

In SEA, a simple majority voting is utilized. DIC combines the historical models with the new model D_t through dynamic selection (DS), dynamic voting (DV), or DV with Selection (DVS). For each testing example, its k nearest neighbors in D_t are first identified. Then, the local performance of each individual model is estimated based on these nearest training examples. For each testing example, DS chooses the individual model with the best local performance for classification. DV does not select a single model, but combines the output of all individual models using a weighted voting scheme. DVS is similar to DV, but only takes a half of the individual models that have the best local performance and applies DV to them. AUE2 also employs weighted voting as the combination scheme, where the weights assigned to individual models are determined by mean squared errors of the models. In Learn^{++}.NSE, the weight assigned to an individual model not only depends on its performance on the current training data (as DIC and AUE2), but also depends on its performance on previous data. Specifically, the weight is dynamically adapted as the logarithm of the reciprocal of the model's error on the current and past data chunks, so that the models that performed well on the data that arrive more recently would generally be assigned larger weights when forming an ensemble for the current concept. TIX employs a different method to leverage historical models. That is, given a new chunk of training data D_t, the outputs of the historical models on D_t are used as new features of D_t, and a new model is built with the augmented D_t. If only linear models are built during the learning process, TIX could also be viewed as a special weighted voting scheme, since a linear combination of the original features of D_t can be viewed as the outputs of a linear model on the original D_t.

Preserving historical models induces overhead in terms of both storage and computation, e.g., repeatedly assessing the performance of historical models on new training data. Hence, the number of preserved models should be subject to some constraints, instead of increasing unlimitedly. Specifically, given a predefined largest number of preserved models, a selection scheme is needed to decide which historical models should be preserved. This issue is not explicitly addressed in DIC, TIX, and Learn^{++}.NSE. Although the DS/DVS scheme of DIC and the time-adjust errors scheme of Learn^{++}.NSE could be adapted to select historical models to preserve, the effectiveness of such adaptations have not be assessed. SEA and AUE2, in contrast, explicitly control the number of preserved models under a predefined threshold. To be specific, both SEA and AUE2 preserve all historical models if the size of the historical model set is smaller than the predefined value. Otherwise, when a new model is trained, a rule, which assesses the quality of both the preserved models and the new model, is employed to decide whether the new model should replace a previously preserved model. Both approaches measure the quality of individual models from the accuracy perspective. The difference is that SEA considers the overall accuracy of ensembles (obtained by replacing a preserved model with the new model) on the current training data, while AUE2 evaluates each individual model under consideration on the current training data directly. None of the existing ensemble methods for incremental learning has considered ensemble diversity explicitly, although diversity has been shown to play a crucial role in ensembles [20], [21].

III. Proposed Approach

The existing ensemble methods for incremental learning, as discussed in Section II, share a common design choice. That is, when a new data chunk arrives, all the approaches utilize the preserved historical models without adapting them to the new training data. This manner is effective and easy-to-implement in incremental learning. However, it is not the best way to exploit these models. Due to the problem of concept drift, the knowledge in the historical model is nonidentical to the current one. Assuming a historical model is established from concept C_1, which is quite different from the current concept C_2, this historical model may not perform well on the current data, and a good combination scheme will reduce the effect of the historical model on the final ensemble, e.g., assign a small weight to its output. In the extreme case (a weight of 0), the knowledge contained in this historical model is not exploited at all. However, viewing concepts C_1 and C_2 from the same incremental learning task as the source and target domains of transfer learning [22], it is reasonable to assume that C_1 and C_2 are correlated with each other. Then, the historical model could still be adapted to the current concept via knowledge transfer. More importantly, an appropriate transfer learning method in this case might even benefit learning concept C_2 in terms of accuracy, learning efficiency, or both. Motivated by this consideration, we propose that the historical models are employed as the initial candidate models for building models for new concepts, which is the core idea behind the DTEL approach proposed in this paper.

The framework of DTEL, as given in Algorithm 1, differs from the other ensemble methods for incremental learning in two aspects. First, DTEL does not directly combine the outputs of historical models. Instead, each preserved historical model is first adapted to fit the current data, and then the adapted models and the model constructed from scratch are combined, as illustrated in Fig. 1. Second, historical models are preserved according to a diversity-based criterion, rather than an accuracy-based one. These two key components and concrete steps of DTEL are detailed in the following sections.

Algorithm 1 Framework of DTEL

```
Input: $(D_1, D_2, \cdots, D_t, \cdots)$: the data stream in incremental learning, $S$: a set of preserved historical models
Output: $F_t$, the generalized ensemble model at each time step $t$
1: while data chunk $D_t$ is available do
2:     train a new base model $f_t$ with $D_t$
3:     obtain the transferred models $f_i^t$ by transferring the preserved historical models $f_i \in S$
4:     construct the ensemble model $F_t$ with the transferred models $f_i^t$ and the newly trained model $f_t$
5:     update $S$ with $f_t$ to maximize the diversity and meet the requirement of the archive size
6: end while
```

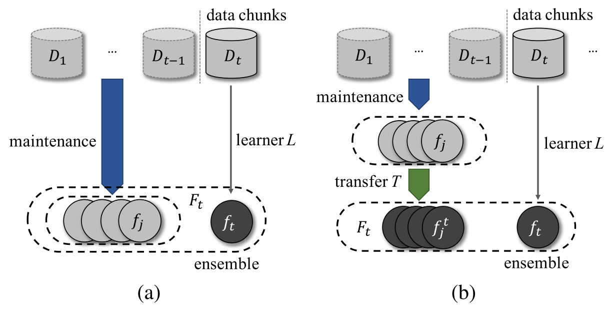

Fig. 1. Illustration of the learning flow. (a) Existing chunk-based ensemble. (b) DTEL.

A. Diversity-Based Model Preservation

Choosing historical models to preserve in the DTEL framework can be stated as a general problem of selecting m historical models out of M, where selected models will be employed as the initial models to be further trained with a new coming data chunk (possibly generated by a drifted concept) and be combined to form an ensemble. In this context, it is the performance of the adapted models after further training that matters, rather than the performance of the original models on the new-coming data chunk. It is well acknowledged in the ensemble learning literature [20], [21] that, with an appropriate combination scheme, diversity among individual models are essential. Diversity between individual models should be encouraged, which could be implemented by diverse training data, diverse initial models, and different learning algorithms [20]. In fact, diversity may play an even more important role in DTEL in case of concept drift. Specifically, without prior knowledge regarding the correlations between different concepts, which are nontrivial to know in practice, a historical model that can be nicely adapted to a new data chunk may not be easily adapted to other concepts that may occur in the future. Since the combination scheme functions as a filter to prevent bad individual models deteriorating the performance of an ensemble, a good final ensemble could still be obtained as long as at least one of the preserved historical models could be adapted to the new concept. For this reason, it is desirable that the preserved historical models have sufficient diversity to deal with different concepts, instead of requiring all of them to adapt to the current concept. Therefore, DTEL employs diversity among historical models, instead of the accuracy of either individual models or the ensemble built for the current concept, as a principle for preserving historical models.

Following the diversity-based principle, DTEL preserves historical models using a two-stage strategy. Let historical models be preserved in an archive of size m. When a data chunk D_t arrives at time step t, the preserved models will be tested on D_t and a new model f_t will be built from scratch using D_t. The newly built model f_t will be directly preserved if the archive is not full. Otherwise, f_t will be temporarily incorporated into the archive. Then, the model whose removal will lead to the largest diversity between the remaining models is removed from the archive. In general, any diversity measure [21] proposed for ensemble learning could be used for this purpose. In this paper, the Yules Q-statistic [23] is employed since it is one of the most popular diversity measures in the literature. Concretely, a model (either a historical or the new one) is removed to maximize

$$\operatorname{div}(S) = 1 - \frac{1}{\sum_{1\le i\neq j\le m} 1} \sum_{1\le i\neq j\le m} Q(f_i, f_j) \tag{3}$$

where $Q(f_i, f_j)$ is the Q-statistic value between f_i and f_j, and is calculated by

$$Q(f_i, f_j) = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}} \tag{4}$$

where N^{ab} is the number of examples for which the classification result is a by f_i and b by f_j, 1 represents a correct classification, and 0 represents a misclassification.

B. Adapting Historical Models Through Knowledge Transfer

Since different learning machines have different learning mechanisms, adapting a historical model to a new data chunk is a model-dependent problem. DTEL employs decision tree as its base learner. A knowledge transfer method has been designed to adapt a previously trained decision tree to a new data chunk. Recall that the process of growing a decision tree incrementally splits the feature space into small regions. Each region corresponds to a leaf node of the tree and is assigned a class label. The structure of the tree inherently contains the knowledge learned from previous subtasks, while the class labels assigned to the obtained regions determine the decision boundary. Hence, the proposed knowledge transfer method aims to preserve the structure of a historical decision tree, while perturbing it to fit the new data chunk. To be specific, this is done in two steps, as detailed below.

Step 1: Place the examples in the new data chunk D_t into the leaf nodes, and reset the class labels of the nodes correspondingly. This step could be viewed as requiring the structure of the adapted decision tree to be as similar to that of the original [24], such that the knowledge contained in the original tree could be maintained.

Step 2: To meet the requirement of correctly classifying the data in chunk D_t, further train a subtree in the leaf node in which the stopping criteria has not been reached.

It should be noted that an adapted decision tree derives the knowledge from the current data and the corresponding historical data. Hence, it represents a hybrid knowledge, which may not exactly belong to a certain data distribution in the incremental learning task. Besides, the adapted decision trees always fit the current data and are less diverse than the preserved historical trees, since the latter were built with different data chunks. For the reasons given above, the adapted trees will not be preserved, i.e., they will be discarded when the next chunk of data arrives.

C. Detailed Steps of DTEL

Given the two key components described in Sections III-A and III-B, the detailed steps of DTEL are presented in Algorithm 2. Suppose t data chunks $D_1 \cdots D_t$ arrive sequentially. The classification and regression tree (CART) [25], is employed as the base learner in DTEL. DTEL first builds a decision tree, denoted as f_1, with the first chunk of data and preserve f_1 in an archive. Then, when a new data chunk, say D_t arrives, the preserved decisions tree(s) is adapted to D_t and a new decision tree f_t is built from scratch based on D_t. The adapted trees and the new tree are combined to form the final ensemble for time step t. Meanwhile, the new tree f_t is used to update the archive of the preserved historical models. Among the combination schemes described in Section II, the weighted voting scheme used by AUE2 is employed because AUE2 showed the best overall performance among ensemble methods for incremental learning [7]. Specifically, the weights for individual adapted trees are determined using (5). Since the new decision tree f_t is treated as a "perfect" classifier, the weight for f_t is calculated according to (6)

$$w_i^t = \frac{1}{\mathrm{MSE}_r^t + \mathrm{MSE}_i^t + \epsilon} \tag{5}$$

$$w_t = \frac{1}{\mathrm{MSE}_r^t + \epsilon} \tag{6}$$

where MSE_i^t estimates the prediction error of the adapted tree f_i on data chunk D_t, MSE_r^t is the mean square error of a random classifier, and ϵ is a very small positive value to avoid the denominator of (5) being 0. MSE_i^t and MSE_r^t are calculated as follows:

$$\mathrm{MSE}_i^t = \frac{1}{|D_t|} \sum_{\{\mathbf{x}, y\}\in D_t} \left(1 - p_i^t(y|\mathbf{x})\right)^2 \tag{7}$$

$$\mathrm{MSE}_r^t = \sum_y p^t(y)(1 - p^t(y))^2 \tag{8}$$

where $p_i^t(y|\mathbf{x})$ denotes the posterior probability given by adapted classifier of f_i on data in chunk D_t and $p^t(y)$ denotes

Algorithm 2 DTEL

```
Input: $(D_1, D_2, \cdots, D_t, \cdots)$: the data stream, $T$: model
    transfer function, $S_t$: the historical model set at time step
    $t$, $E_t$: the ensemble model set at time step $t$, $m$: the archive
    size of the historical model set, div: model maintenance
    function with diversity
Output: $F_t$, the generalized model at each time step $t$
1: while data chunk $D_t$ is available do
2:    $f_t \leftarrow$ train a new base model with $D_t$
3:    $f_i^t \leftarrow T(f_i, D_t)$, for all $f_i \in S_t$
4:    if $|S_{t-1}| < m$ then
5:       $S_t \leftarrow S_{t-1} \bigcup \{f_t\}$
6:    else
7:       $S_t^{'} \leftarrow S_{t-1} \bigcup \{f_t\}$
8:       $f_{replace} \leftarrow \arg\max_{f_i} \text{div}(S_t^{'} - \{f_i\})$
9:       $S_t \leftarrow S_t^{'} - \{f_{replace}\}$
10:   end if
11:   $w_i^t \leftarrow$ estimate the weight for each $f_i^t$ by Eq. (5)
12:   $w_t \leftarrow$ estimate the weight for $f_t$ by Eq. (6)
13:   $F_t = (\sum_i w_i^t f_i^t + w_t f_t)/(\sum_i w_i^t + w_t)$
14: end while
```

the prior probability of class y in D_t. The posterior probability in the tree model is calculated as the ratio of that class in the corresponding leaf node. When the archive of historical trees is full, lines 9–11 will be activated to decide whether a newly trained tree will replace a preserved tree in the archive.

IV. Experiment

Empirical studies have been conducted to assess the performance of DTEL. Different types of concept drift are involved, and state-of-the-art algorithms are compared in the experiment. The algorithms are evaluated mainly from three aspects, i.e., the classification performance for each chunk, the overall performance for a whole data stream, and the time efficiency.

The following algorithms are compared in the experiments: SEA [6], Learn^{++}.NSE [17], AUE2 [7], and TIX [18]. These approaches employ different methods for concept drift adaptation. SEA, Learn^{++}.NSE, and AUE2 are ensemble methods to exploit the historical knowledge, while TIX uses the historical knowledge by introducing it as new features in a transfer manner. SEA and Learn^{++}.NSE use the historical model directly in an ensemble, while AUE2 updates each historical model with the new chunk of data. In addition, for the weight assignment in the ensemble method, SEA uses the uniform weight, AUE2 employs the performance-based weight, and Learn^{++}.NSE uses the time-adjusted performance-based weight.

All of the compared approaches are frameworks. To make the comparisons fair, the decision tree was employed as the base learner in all of these approaches. Specifically, the traditional decision tree method CART [25] is applied in SEA, Learn^{++}.NSE, TIX, and DTEL. Since AUE2 needs to use an on-line model as the base learner, Hoeffding tree [26], an on-line decision tree method, was applied.

In SEA, AUE2, and DTEL, a limited number of historical models are preserved. The archive size of the historical model set is the only parameter to set in the experiment. According to the suggestion in [6], the ensemble size is set to 25 for the compared algorithms, unless mentioned otherwise.

A. Comparison on Synthetic Data

1) Data Sets: In order to comprehensively investigate the performance of DTEL, five types of concept drift were tested in the experiment. When working with synthetic data, it is known exactly what the type of concept drift is and how dramatic the data distribution changes. Hence, it is important to use the synthetic data for a detailed analysis of the approaches in concept drift adaptation. Based on the previous research [6], [7], [14], [17], [27], [28], five widely used synthetic concept drifts are employed in our experiment, described as follows.

1) *SEA Moving Hyperplane Concepts (SEA)* [6] involve three features with a value between 0 and 10. Only two features (i.e., x_1 and x_2) are relevant, and x_3 is a noisy feature with a random value. The class label of data in this concept is determined by
$$ax_1 + bx_2 \leq / > \theta.$$
To simulate the concept drift, the value of θ changes during the learning process.
2) *Rotating Concepts (ROT)* [7], [17] rotate the decision boundary or data points to simulate the concept drift. The formulation of rotating the data point in the 2-D feature space around (a, b) is shown as follows:
$$x_1 \leftarrow (x_1 - a)\cos\theta - (x_2 - b)\sin\theta + a$$
$$x_2 \leftarrow (x_1 - a)\cos\theta + (x_2 - b)\sin\theta + b.$$
In the experiment, a data set with six classes is used as the data source, and the rotation is implemented evenly in the learning process.
3) *Circle Concepts (CIR)* [14], [27] apply a circle as the decision boundary in a 2-D feature space and simulates the concept drift by changing the radius of the circle, that is
$$(x_1 - a)^2 + (x_2 - b)^2 \leq / > \theta.$$
In the experiment, data points are generated evenly locating between -5 and 5 for both dimensions, and the radius value of θ changes every 25 data chunks.
4) *Sine Concepts (SIN)* [14], [27] determine the label of data by a sine curve in a 2-D feature space, which is defined as follows:
$$a\sin(bx_1 + \theta) + c \leq / > x_2.$$
In the experiment, all of the data locate in the area of [-5, 5] for both dimensions. The value of θ is evenly changed to generate the change in the data distribution.
5) *STAGGER Boolean Concepts (STA)* [27], [28] generate the data with categorical features using a set of rules to determine the class label. According to [27] and [28],

TABLE I
ARTIFICIAL CONCEPT DRIFT

Drift Type	Fixed Value	Drift Value
SEA	$a = 1,\, b = 1$	$\theta = 10 \to 7 \to 3 \to 7 \to 10 \to 13 \to 16 \to 13$
		$\theta = 10 \to 8 \to 6 \to 8 \to 10 \to 12 \to 14 \to 12$
ROT	$a = 0,\, b = 0$	$\theta = 0,\, \Delta\theta = \pi/30$
		$\theta = 0,\, \Delta\theta = \pi/60$
CIR	$a = 0,\, b = 0$	$\theta = 3 \to 2 \to 1 \to 2 \to 3 \to 4 \to 5 \to 4$
		$\theta = 3 \to 2.5 \to 2 \to 2.5 \to 3 \to 3.5 \to 4 \to 3.5$
SIN	$a = 1,\, b = 1,\, c = 0$	$\theta = 0,\, \Delta\theta = \pi/30$
		$\theta = 0,\, \Delta\theta = \pi/60$
STA	$c = S \wedge M \wedge L$ $\wedge_2$ $=_1,\, =_2,\, =_3$	$(a = R, \wedge_1, b = C) \to (a = B, \vee_1, b = C) \to (a = G, \vee_1, b = S) \to (a = G, \wedge_1, b = T) \to (a = G, \vee_1, b = C) \to (a = R, \vee_1, b = S)$
		$(a = R, \wedge_1, b = C) \to (a = B, \wedge_1, b = C) \to (a = B, \vee_1, b = C) \to (a = B, \vee_1, b = S) \to (a = B, \wedge_1, b = S) \to (a = G, \wedge_1, b = S)$

TABLE II
SYNTHETIC DATA STREAMS IN EXPERIMENT

Data Stream	#Example	#Feature	#Label	Chunk Size
SEA200A	24,000	3	2	200
SEA200G	24,000	3	2	200
SEA500G	60,000	3	2	500
ROT200A	24,000	2	6	200
ROT200G	24,000	2	6	200
ROT500G	60,000	2	6	500
CIR200A	24,000	3	2	200
CIR200G	24,000	3	2	200
CIR500G	60,000	3	2	500
SIN200A	24,000	2	2	200
SIN200G	24,000	2	2	200
SIN500G	60,000	2	2	500
STA200A	24,000	3	2	200
STA200G	24,000	3	2	200
STA500G	60,000	3	2	500

the features and values are color ∈ {red(R), blue(B), green(G)}, shape ∈ {circle(C), square(S), triangle(T)}, and size ∈ {small(S), medium(M), large(L)}. The decision rules can be formulated as follows:

$$y = ((\text{color} =_1 / \neq_1\ a) \vee_1 / \wedge_1 (\text{shape} =_2 / \neq_2\ b)) \\ \vee_2 / \wedge_2 (\text{size} =_3 / \neq_3\ c)$$

The concept drift is simulated by changing the items in the rules.

The dramatic degree of concept drift and the size of data chunks may influence the performance of the learning algorithm. In this regard, three data streams are generated for each type of concept drift, with different dramatic degrees of concept drift and different chunk sizes. All of the synthetic data streams are generated with 120 data chunks with 10% noise introduced. The specific setting of the values for simulating the concept drifts are illustrated in Table I. The statistics of the synthetic data streams are described in Table II. For synthetic data streams, two chunks of data are generated at each learning step. The first data chunk is used for training and the other one is used to test the current prediction model.

2) Results: In order to investigate the performance of the algorithms in concept drift adaptation, the prediction accuracy in each chunk of data are evaluated and presented in Fig. 2. Generally speaking, the accuracy result obtained from the proposed DTEL is not only the highest among the compared algorithms but also relatively stable across different synthetic data streams. Specifically, for SEA data streams [as shown in Fig. 2(a)–(c)], DTEL performed steadily on each data chunk and was not affected by concept drifts, no matter how dramatic the concept drift is. Although AUE2 obtained the highest accuracy on several of the data chunks, it can be observed that AUE2 is sensitive to the concept drift with a drop in accuracy. All of the compared algorithms are sensitive to the ROT concept drift, with a dramatic fluctuation of classification performance, as shown in Fig. 2(d)–(f). However, DTEL still obtained the highest accuracy on all the data chunks, which illustrates the rapid concept drift adaptation ability of DTEL. It is interesting to note that although the ROT concept drift is generated by smoothly rotating the decision boundary, the performance of the compared algorithms periodically rises and falls, instead of decaying gradually. For the CIR concept drift [Fig. 2(g)–(i)], the performance of the compared algorithms are relatively similar. The dramatic degree of CIR concept drift appears not to influence the performance of DTEL based on the observation of results on CIR200A and CIR200G. In contrast, the other compared algorithms are affected by this type of concept drift. The results obtained from the SIN data streams [as shown in Fig. 2(j)–(l)] also demonstrate the superiority of DTEL, which performed the best on almost all the data chunks. Different from the previously analyzed data streams, STA data streams are generated from decision rules. For the results obtained from STA data streams [Fig. 2(m)–(o)], the compared algorithms can be divided into two categories. DTEL and TIX show a high accuracy with a low variance among different data chunks, while the accuracy results of SEA, AUE2 and Learn^{++}.NSE fluctuate dramatically and are more sensitive to this type of concept drift.

To evaluate the performance of the compared algorithms on the whole data stream, the classification accuracy of each algorithm is averaged over the data chunks in a data stream and presented in Table III. The standard deviations over data chunks, which indicates how stably an algorithm performs on a data stream, are also given. Generally, DTEL shows a clear advantage over other algorithms, in terms of high accuracy and low standard deviation on most data streams. On SEA200G, AUE2 obtained the highest accuracy, but its standard variance is higher than DTEL, which means AUE2 is more sensitive to concept drift on SEA200G than DTEL. TIX model performed the best on the STA data streams. The reason might be that the data in STA streams are generated by decision rules and the historical knowledge is represented

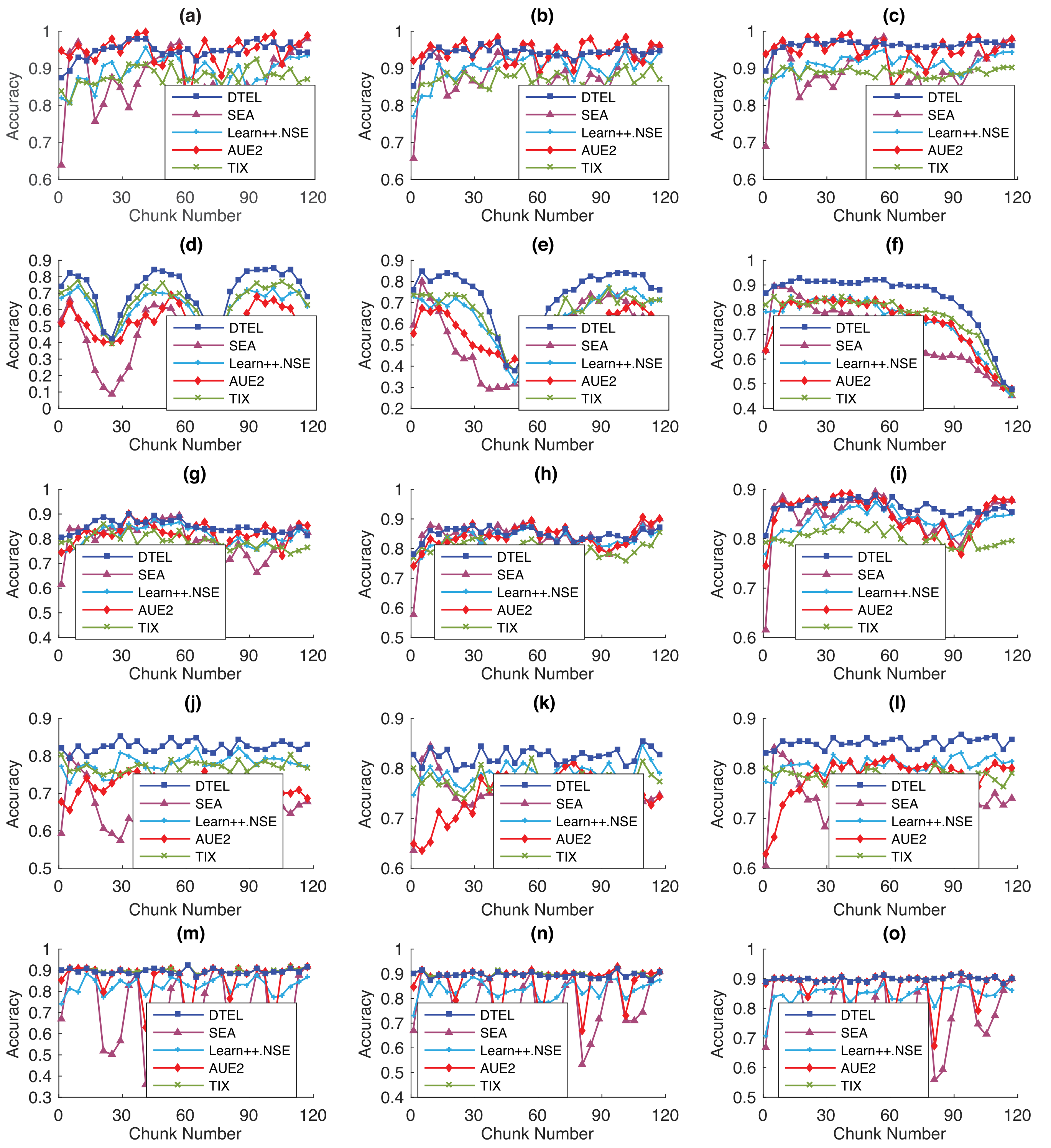

Fig. 2. Accuracy results on synthetic data streams. (a) SEA200A. (b) SEA200G. (c) SEA500G. (d) ROT200A. (e) ROT200G. (f) ROT500G. (g) CIR200A. (h) CIR200G. (i) CIR500G. (j) SIN200A. (k) SIN200G. (l) SIN500G. (m) STA200A. (n) STA200G. (o) STA500G.

as a new feature value in TIX. By employing decision tree as the learner, TIX can exploit the useful historical knowledge more easily with the tree structure. The Wilcoxon rank-sum test at the 95% confidence level has been conducted to check whether the differences (in terms of average accuracy) between DTEL and the compared algorithms are statistically significant, and the result is shown in the last row of Table III as the statistics of win-tie-loss. As shown in the statistical result, DTEL performed significantly better than other algorithms on the synthetic data streams in pairwise comparisons. The average accuracy results in Table III further verify the detailed observation results of the performance of DTEL on each data chunk shown in Fig. 2. It can be concluded that DTEL is able to handle better the concept drift of different types and different dramatic degrees, by using the historical knowledge.

TABLE III

AVERAGE ACCURACY (%) OF EVERY CHUNK (± THE STANDARD DEVIATION OF THE ACCURACY FOR EACH CHUNK) ON SYNTHETIC DATA STREAMS. ●/○ INDICATES THAT DTEL IS SIGNIFICANTLY BETTER/WORSE THAN THE CORRESPONDING ALGORITHM (WILCOXON RANK-SUM TEST AT 95 PERCENT CONFIDENCE LEVEL). THE VALUES IN BOLDFACE INDICATE THE HIGHEST ACCURACY ON THE CORRESPONDING DATA STREAM

Data	DTEL	SEA	Learn^{++}.NSE	AUE2	TIX
SEA200A	**94.77 ± 2.99**	86.31 ± 11.43●	89.07 ± 5.13●	94.66 ± 4.94	87.77 ± 3.97●
SEA200G	94.07 ± 2.69	88.90 ± 10.02●	90.02 ± 4.98●	**94.58 ± 3.80**○	86.90 ± 4.26●
SEA500G	**96.21 ± 1.76**	89.37 ± 10.17●	91.10 ± 3.45●	95.02 ± 4.05	88.85 ± 2.54●
ROT200A	**71.86 ± 13.91**	37.88 ± 18.17●	62.19 ± 11.49●	52.72 ± 9.99●	65.02 ± 11.45●
ROT200G	**72.67 ± 14.28**	54.61 ± 17.45●	63.41 ± 12.84●	55.43 ± 9.76●	64.97 ± 12.16●
ROT500G	**84.27 ± 12.20**	69.81 ± 14.29●	74.77 ± 11.57●	74.34 ± 11.34●	76.98 ± 10.44●
CIR200A	**84.84 ± 3.88**	79.90 ± 10.10●	81.33 ± 5.79●	82.21 ± 5.29●	78.98 ± 5.30●
CIR200G	**84.88 ± 3.37**	83.86 ± 8.60	83.27 ± 4.38●	84.06 ± 4.63	80.04 ± 4.65●
CIR500G	**86.42 ± 2.22**	84.32 ± 8.43●	83.60 ± 3.00●	84.87 ± 4.36●	80.20 ± 2.70●
SIN200A	**82.51 ± 2.84**	65.78 ± 8.44●	78.02 ± 4.00●	71.91 ± 3.69●	77.05 ± 3.91●
SIN200G	**82.40 ± 3.22**	74.12 ± 8.11●	79.07 ± 3.27●	74.32 ± 5.48●	77.23 ± 3.70●
SIN500G	**85.10 ± 1.91**	73.76 ± 7.80●	80.67 ± 2.51●	78.22 ± 4.71●	78.36 ± 2.84●
STA200A	89.48 ± 2.34	70.07 ± 21.57●	82.74 ± 7.88●	86.06 ± 10.93	**89.85 ± 2.01**
STA200G	89.56 ± 2.50	76.01 ± 15.71●	83.56 ± 7.54●	86.58 ± 9.06	**89.77 ± 2.32**
STA500G	**90.00 ± 1.34**	76.43 ± 15.35●	84.78 ± 7.25●	87.47 ± 7.73	**90.00 ± 1.34**
win-tie-loss		14-1-0	15-0-0	8-6-1	12-3-0

B. Comparison on Real-World Data

1) Data Sets: In addition to the synthetic data streams, five real-world data streams, namely, covertype, pokerhand, electricity, and click-through rate prediction (CTRPrediction) data were also employed in our experiments. Details of these data sets are described as follows.

1) **Covertype** [29] is a real-world data set for describing the observation of a forest area with 51 cartographic variables. Six class labels are involved to represent the corresponding forest cover type.
2) **PokerHand** [29] describes the suits and ranks of a hand of five playing cards. It involves five numerical features and five ordinal features for each example. Ten class labels exist in the data set for describing different pokerhands.
3) **Electricity**, a widely used data set [14], [7], is collected from the New South Wales Electricity Market in Australia, containing 45 312 instances dated from May 7, 1996 to December 5, 1998. Each example is described by eight features, and the class label identifies the change of the price (i.e., up and down).
4) **Christmas** [3] is a data set generated from twitter, a real-world application, and involves tweet data posted around Christmas Day in 2009 and 2010. There are three classes (music, job, and Christmas) and 15 000 examples in Christmas data, with 247 features. Since Christmas is about periodic event, the recurring concepts are involved.
5) **CTRPrediction** is a data set obtained from the Tencent company. All of the examples in the data set are in their original order collected through 30 days. After the preprocessing of the raw data, 20 000 examples are selected for each day, and totally 600 000 examples with 100 features are tested.

The chunk size was set to 1000 for the first three real-world data streams. Since the size of Christmas is small, the chunk size is set to 500. The chunk size for CTRPrediction data was set to 10 000 and each chunk of data represents half a day's observations from the real-world application. The statistics of the real-world data streams are described in Table IV. For real-world data streams, each chunk of data is first used to test the current prediction model and then to update the model in learning. Considering the number of chunks in Christmas is relatively small and the chunk size of CTRPrediction is large, the ensemble size for both data sets was set to 3 for all compared algorithms on this data set.

TABLE IV

REAL-WORLD DATA STREAMS IN EXPERIMENT

Data Stream	#Example	#Feature	#Label	#Chunk	Chunk Size
Covertype	581,000	51	7	581	1,000
PokerHand	1,000,000	10	10	1,000	1,000
Electricity	44,000	8	2	44	1,000
Christmas	15,000	247	3	30	500
CTRPrediction	600,000	100	2	60	10,000

2) Results: The accuracy of the compared algorithms is shown in Fig. 3. Since no detailed information regarding the occurrences and behaviors of concept drift is known in the real-world data streams, it is hard to conduct an exact analysis of the adaptation ability for concept drift. Nevertheless, the empirical results on real-world data streams validate the conclusion drawn from the synthetic data. For the Covertype data stream [Fig. 3(a)], DTEL showed a stable performance along the whole incremental learning process, while the compared algorithms showed an unstable classification performance, especially the SEA approach. On PokerHand data stream [Fig. 3(b)], SEA performs better than DTEL. A possible reason is that random events may happen in poker card game and lead the data distribution in PokerHand data to be almost randomly changed, which is close to the assumption in SEA approach. On PokerHand data, DTEL generally performed more stable than AUE2 and obtained a better performance on all of the data chunks than Learn^{++}.NSE and TIX. The data streams of Electricity are relatively hard to learn, on which the compared algorithms performed similarly and unstably. It is hard to distinguish the best performed algorithm in Fig. 3(c). The performance of AUE2 is obviously worse than the other compared algorithms on Christmas data [Fig. 3(d)], and DTEL is generally competitive among the algorithms.

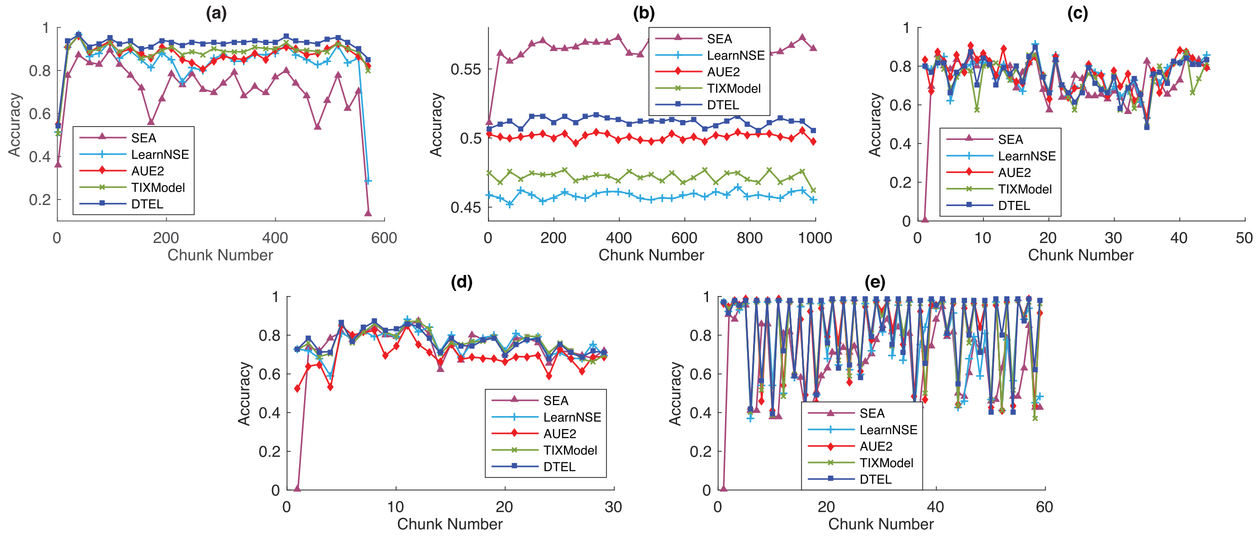

Fig. 3. Accuracy results on real-world data streams. (a) Covertype. (b) Pokerhand. (c) Electricity. (d) Christmas. (e) CTR prediction.

TABLE V

AVERAGE ACCURACY (%) OF EVERY CHUNK (± THE STANDARD DEVIATION OF THE ACCURACY FOR EACH CHUNK) ON REAL-WORLD DATA STREAMS. •/◦ INDICATES THAT DTEL IS SIGNIFICANTLY BETTER/WORSE THAN THE CORRESPONDING ALGORITHM (WILCOXON RANK-SUM TEST AT 95 PERCENT CONFIDENCE LEVEL). THE VALUES IN BOLDFACE INDICATE THE HIGHEST ACCURACY OF THE COMPARED ALGORITHMS (EXCEPT TEL) ON THE CORRESPONDING DATA STREAM. TEL REPRESENTS THE DTEL ALGORITHM WITHOUT DIVERSITY-BASED MODEL SELECTION COMPONENT. THE UNDERLINED VALUES REPRESENT THE HIGHER ACCURACY BETWEEN DTEL AND TEL

Data	DTEL	TEL	SEA	Learn^{++}.NSE	AUE2	TIX
Covertype	**91.80 ± 8.42**	91.79 ± 8.43	71.46 ± 15.14•	84.11 ± 12.45•	87.09 ± 8.74•	88.32 ± 9.15•
PokerHand	51.20 ± 1.81	51.28 ± 1.79	**56.36 ± 2.54**◦	45.86 ± 1.87•	51.31 ± 1.79	47.23 ± 1.73•
Electricity	75.05 ± 8.36	74.90 ± 8.34	72.35 ± 13.99	75.54 ± 8.09	**76.47 ± 8.70**	73.55 ± 8.72
Christmas	**76.68 ± 5.59**	75.30 ± 5.82	73.29 ± 15.33	76.63 ± 6.43	69.81 ± 8.05 •	76.50 ± 6.01
CTRPrediction	**82.02 ± 20.90**	83.15± 19.80	66.08 ± 19.36•	77.13 ± 20.96•	80.57 ± 21.34•	80.57 ± 21.58•
	win-tie-loss		2-2-1	3-2-0	3-2-0	3-2-0

TABLE VI

FRIEDMAN TEST (NEMENYI TEST, $CD = 1.4377$ AT $\alpha = 0.05$) RESULT CONSIDERING THE AVERAGE ACCURACY ON BOTH SYNTHETIC AND REAL-WORLD DATA STREAMS

algorithm	DTEL	SEA	Learn^{++}.NSE	AUE2	TIX
rank	1.3500	4.3750	3.2250	2.9000	3.1500

On CTRPrediction data [Fig. 3(e)], the performance of all of the compared algorithms fluctuates dramatically and DTEL generally obtained the best accuracy on each data chunk.

It is worth noting that the accuracy on each chunk of CTR-Prediction data may reveal an interesting rule in click habit from the website visitors. The 20 000 examples for each day are randomly extracted from the website, and each day's data are divided into two consecutive chunks in CTRPrediction. Hence, the consecutive chunks roughly embed the click habits from the first half and second half of a day, which roughly represent the working time and leisure time, respectively. From the result shown in Fig. 3(d), it can be observed that the classification performance is distinctly different on two consecutive chunks, with a close to 100% accuracy followed by a roughly 50% one. This indicates that the click habit changes in different time spans and also reveals the difficulty in learning the dynamic real-world data.

Table V presents the average accuracy obtained from the real-world data streams. DTEL shows a great advantage over other algorithms, which is consistent with the results obtained on the synthetic data streams. Specifically, DTEL demonstrates significantly best accuracy results on the real-world data streams, except for Electricity. AUE2 obtained the highest accuracy on the Electricity data stream. However, the value of the corresponding standard variance indicates that AUE2 is more sensitive to the concept drift in Electricity than DTEL and Learn^{++}.NSE. Moreover, there is no statistically significant difference among the performance of the compared algorithms on the Electricity data stream.

To deeply analyze DTEL, DTEL without the diversity-based model preservation component, named as TEL, is also evaluated in Table V. In TEL, when the size of historical model set reaches the archive size, the model with the largest mean square error will be replaced by the current one. It can be observed that DTEL is slightly better than TEL. Hence, in DTEL, the transfer operation makes the main contribution to the performance improvement. On this basis, the diversity-based model preservation component further refines the learning algorithm.

To make a comprehensive comparison, a Friedman test [30] is conducted based on the average accuracy results on both synthetic data streams (Table III) and real-world data streams (Table V), as shown in Table VI. The rank value of DTEL is 1.2895 and significantly outperforms others. The Friedman test result demonstrates that the proposed DTEL is significantly better than all of the compared approaches with regard to the average accuracy value.

TABLE VII

RUNTIME OF EACH ALGORITHM (UNIT: SECONDS). THE VALUE IN THE BRACKETS AFTER THE RUNTIME OF DTEL IS TIME COST OF THE TRANSFER OPERATIONS. LNSE REPRESENTS LEARN^{++}.NSE

Data	DTEL	SEA	LNSE	AUE2	TIX
SEA200A	2.66e2 (2.43e2)	3.17e1	2.09e1	5.09e0	2.06e1
SEA200G	2.71e2 (2.48e2)	3.20e1	1.97e1	5.73e0	2.04e1
SEA500G	6.28e2 (5.77e2)	7.27e1	2.64e1	1.21e1	2.52e1
ROT200A	2.97e2 (2.74e2)	9.00e1	8.41e1	6.51e0	3.14e1
ROT200G	3.01e2 (2.79e2)	1.03e2	8.40e1	6.70e0	3.13e1
ROT500G	7.10e2 (6.59e2)	2.63e2	1.03e2	1.53e1	3.69e1
CIR200A	2.73e2 (2.51e2)	3.42e1	1.99e1	4.91e0	2.73e1
CIR200G	2.82e2 (2.59e2)	4.14e1	2.40e1	4.83e0	3.12e1
CIR500G	6.25e2 (5.74e2)	9.68e1	2.94e1	1.25e1	3.38e1
SIN200A	2.96e2 (2.74e2)	3.83e1	2.26e1	4.93e0	2.43e1
SIN200G	3.04e2 (2.81e2)	3.90e1	2.19e1	4.90e0	2.28e1
SIN500G	6.67e2 (6.26e2)	9.59e1	2.91e1	1.07e1	2.80e1
STA200A	2.14e2 (1.91e2)	3.84e1	2.89e1	5.88e0	2.14e1
STA200G	2.07e2 (1.85e2)	5.04e1	2.10e1	5.99e0	2.12e1
STA500G	7.09e2 (6.62e2)	9.32e1	2.57e1	1.68e1	2.36e1
Covertype	2.76e3 (2.22e3)	1.55e3	1.02e3	5.41e2	9.66e2
PokerHand	1.27e4 (1.18e4)	8.83e2	7.75e3	2.86e2	6.09e3
Electricity	1.18e2 (9.02e1)	2.83e1	1.53e1	1.22e1	5.49e0
Christmas	1.75e1 (1.22e1)	7.78e0	4.79e0	9.63e0	5.08e0
CTRPrediction	3.83e2 (9.02e1)	3.03e2	1.69e2	2.36e2	2.47e2

The runtimes of the approaches are also compared under the same computing environment (2 CPUs of 2.4-GHz Intel Core i5 and 8-GB main memory) in the experiment. The time complexity of DTEL at each learning step is determined by three factors, i.e., the chunk size n, the data dimensionality d, and the archive size of the historical model set m. For DTEL impelmented with decision trees, the time complexity for building a base model is $O(d^2n)$ [31]. In the transfer operation, the new chunk of data is first placed into the leaf nodes with $O(n)$ time complexity, where the coefficient is the height of the transferred tree. Then, an update is conducted with the new chunk of data with a time complexity of roughly $O(d^2n)$. Since DTEL needs to transfer every maintained historical model with the new chunk of data, the time efficiency of DTEL is the worst among the compared approaches. The runtime results are shown in Table VII. It can be observed that DTEL takes about an order of magnitude longer time than the compared algorithms in some cases. The time consumed by the transfer operations in DTEL is also presented in Table VII, which empirically indicates that the transfer operation is time-consuming. Since all the historical models are transferred independently in DTEL, the transfer operations can be implemented in a parallel processing manner. By parallelizing the transfer operations, the speed-up ratio is about m and the runtime of DTEL could be reduced by an order of magnitude to reach a satisfactory runtime level.

C. Influence of Archive Size

The influence of the only parameter in DTEL, i.e., the archive size of the historical model set m, is studied. The appropriate size of the historical model set may be influenced by the data distribution and the types of concept drift in the whole incremental learning process. Five data streams (i.e., SEA200A, ROT200A, CIR200A, SIN200A, and STA200A) are used to test different sizes, and the test result is shown in Fig. 4. The data STA200A is generated from decision rules and the STA concept drift does not change the tree structure of each base model. Hence, a very small archive size is enough to facilitate the incremental learning with concept drift. Comprehensively considering the test result, when the archive size is smaller than 20, the average accuracy improves with the size increasing. Then, the accuracy results remain roughly stable when the value is larger. Hence, in practice, the archive size with a value bigger than 20 is recommended.

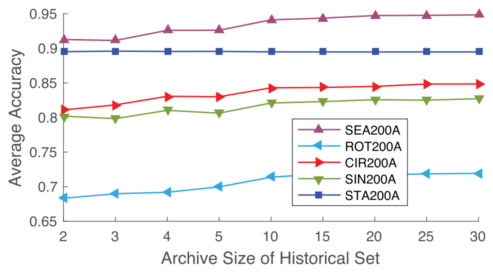

Fig. 4. Sensitive analysis of the size of the historical model set in DTEL.

V. CONCLUSION

This paper presents a new ensemble learning approach, namely, DTEL, for incremental learning with concept drift. DTEL employs a diversity-based selection criterion to preserve previously trained models. Instead of being applied directly to form an ensemble for the current concept, the preserved models are further adapted to the current concept through transfer learning. Empirical studies on both synthetic and real-world data streams demonstrate the advantages of DTEL over a number of state-of-the-art incremental learning methods.

The main potential drawback of DTEL is that it is computationally more costly than the compared methods. Although this disadvantage could be alleviated by parallel implementation of DTEL since it can be naturally parallelized, it is still worth investigating other methods to reduce the complexity of DTEL. For example, it might be unnecessary for all the preserved models to be further trained. Alternatively, some heuristic rules can be designed to identify the preserved models that is most worthy of further training. Besides, this paper only considers decision tree as the base learners of DTEL. Other base learners should be investigated. Although the general framework of DTEL (as in Algorithm 1) is not restricted to decision tree, specific transfer learning (further training) methods need to be designed for different base learners. This would also be an interesting direction for research in the future.

REFERENCES

[1] G. Stiglic and P. Kokol, "Interpretability of sudden concept drift in medical informatics domain," in *Proc. IEEE 11th Int. Conf. Data Mining Workshops*, Dec. 2011, pp. 609–613.

[2] J. Sun, H. Li, and H. Adeli, "Concept drift-oriented adaptive and dynamic support vector machine ensemble with time window in corporate financial risk prediction," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 4, pp. 801–813, Jul. 2013.

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

[4] R. Polikar, L. Upda, S. S. Upda, and V. Honavar, "Learn++: An incremental learning algorithm for supervised neural networks," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 31, no. 4, pp. 497–508, Nov. 2001.

[5] H. He, S. Chen, K. Li, and X. Xu, “Incremental learning from stream data,” *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 1901–1914, Dec. 2011.

[6] W. N. Street and Y. Kim, “A streaming ensemble algorithm (SEA) for large-scale classification,” in *Proc. KDD*, Aug. 2001, pp. 377–382.

[7] 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.

[8] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, “A survey on concept drift adaptation,” *ACM Comput. Surv.*, vol. 46, no. 4, pp. 44:1–44:37, Apr. 2014.

[9] I. Žliobaite, M. Pechenizkiy, and J. Gama, “An overview of concept drift applications,” in *Big Data Analysis: New Algorithms for a New Society* (Studies in Big Data), vol. 16, N. Japkowicz and J. Stefanowski, Eds. Cham, Switzerland: Springer, 2016, pp. 91–114.

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

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

[12] G. Hulten, L. Spencer, and P. Domingos, “Mining time-changing data streams,” in *Proc. KDD*, Aug. 2001, pp. 97–106.

[13] A. Bifet and R. Gavalda, “Learning from time-changing data with adaptive windowing,” in *Proc. SDM*, 2006, pp. 443–448.

[14] J. A. Gama, P. Medas, G. Castillo, and P. Rodrigues, “Learning with drift detection,” in *Advances in Artificial Intelligence—SBIA* (Lecture Notes in Computer Science), vol. 3171. Berlin, Germany: Springer 2004, pp. 286–295. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-540-28645-5_29

[15] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-bueno, “Early drift detection method,” in *Proc. 4th ECML PKDD Int. Workshop Knowl. Discovery Data Streams (IWKDDS)*, Berlin, Germany, 2006, pp. 77–86.

[16] L. L. Minku and X. Yao, “DDD: A new ensemble approach for dealing with concept drift,” *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 4, pp. 619–633, Apr. 2012.

[17] 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.

[18] G. Forman, “Tackling concept drift by temporal inductive transfer,” in *Proc. SIGIR*, Aug. 2006, pp. 252–259.

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

[20] G. Brown, J. Wyatt, R. Harris, and X. Yao, “Diversity creation methods: A survey and categorisation,” *Inf. Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[21] E. K. Tang, P. N. Suganthan, and X. Yao, “An analysis of diversity measures,” *Mach. Learn.*, vol. 65, no. 1, pp. 247–271, 2006.

[22] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[23] G. U. Yule, “On the association of attributes in statistics: With illustrations from the material of the childhood society, &c,” *Philos. Trans. Roy. Soc. London A, Math. Phys. Sci.*, vol. 194, no. 1900, pp. 257–319, 1900.

[24] G. Li, X. Liu, J. Feng, and L. Zhou, “Efficient similarity search for tree-structured data,” in *Proc. SSDBM*, Jul. 2008, pp. 131–149.

[25] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*. Belmont, CA, USA: Wadsworth, 1984.

[26] P. Domingos and G. Hulten, “Mining high-speed data streams,” in *Proc. KDD*, Aug. 2000, pp. 71–80.

[27] 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.

[28] J. C. Schlimmer and R. H. Granger, Jr., “Incremental learning from noisy data,” *Mach. Learn.*, vol. 1, no. 3, pp. 317–354, 1986.

[29] K. Bache and M. Lichman. (2013). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml

[30] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[31] J. Su and H. Zhang, “A fast decision tree learning algorithm,” in *Proc. 21st Nat. Conf. Artif. Intell.*, vol. 1. 2006, pp. 500–505.

Yu Sun (S’14) received the B.Eng. degree in software engineering from the Dalian University of Technology, Dalian, China, in 2010, the M.Eng. degree in software engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2013, and the Ph.D. degree from the School of Computer Science and Technology, USTC, in 2017.

He has been a short-term Visiting Researcher at the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His current research interests include concerns incremental learning and data stream mining.

Ke Tang (M’07–SM’13) received the B.Eng. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2002, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2007.

From 2007 to 2017, he was with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China, first as an Associate Professor (2007–2011) and later a Professor (2011–2017). He then joined the Southern University of Science and Technology, Shenzhen, China, as a Professor with the Department of Computer Science and Engineering. His current research interests include evolutionary computation, machine learning, and their real-world applications. He has authored more than 130 papers in refereed journals and conferences in these areas.

Dr. Tang is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and a member of Editorial Boards for a few other journals. He was a recipient of the Royal Society Newton Advanced Fellowship (2015) and the 2018 IEEE Computational Intelligence Society Outstanding Early Career Award.

Zexuan Zhu received the B.S. degree in computer science and technology from Fudan University, Shanghai, China, in 2003, and the Ph.D. degree in computer engineering from Nanyang Technological University, Singapore, in 2008.

He is currently a Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include computational intelligence, machine learning, and bioinformatics.

Dr. Zhu is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE, and an Editorial Board Member of the *Memetic Computing Journal* and the *Soft Computing Journal*.

Xin Yao (M’91–SM’96–F’03) is currently a Chair Professor of computer science with the Southern University of Science and Technology, Shenzhen, China, and a part-time Professor with the University of Birmingham, Birmingham, U.K. He is a Distinguished Lecturer of the IEEE Computational Intelligence Society (CIS). His current research interests include evolutionary computation and ensemble learning, especially online ensemble learning and class imbalance learning, and their applications in software engineering and fault diagnosis.

Dr. Yao was the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2003 to 2008, and the President of the IEEE CIS from 2014 to 2015. He was a recipient of the 2001 IEEE Donald G. Fink Prize Paper Award, the 2010, 2016, and the 2017 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Awards, the 2010 BT Gordon Radley Award for the Best Author of Innovation (Finalist), the 2011 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and many other Best Paper Awards. He was also a recipient of the Royal Society Wolfson Research Merit Award in 2012 and the IEEE CIS Evolutionary Computation Pioneer Award in 2013.