ADTCD: An Adaptive Anomaly Detection Approach Toward Concept Drift in IoT

Lijuan Xu, Xiao Ding, Haipeng Peng, Dawei Zhao, *Member, IEEE*, and Xin Li

***Abstract*—The data collected by sensors is streaming data in the Internet of Things (IoT). Although existing deep-learning-based anomaly detection methods generally perform well on static data, they struggle to respond timely to streaming data after distribution changes. However, streaming data suffers from conceptual drift due to the highly dynamic nature of IoT. In network security, concept drift-oriented anomaly detection is a crucial task, because it can adjust the model to adapt to the latest data, and detect attacks in time. Existing streaming anomaly detection methods are confronted with some challenges, including the latency of model updates, the uneven importance of new data, and the self-poisoning due to model self-updates. To tackle the above challenges, we propose a knowledge distillation-based adaptive anomaly detection model toward concept drift, ADTCD. ADTCD transfers the knowledge of the teacher model to the student model and only updates the student model to reduce the delay. We construct an algorithm of dynamically adjusting model parameters, which dynamically adjusts model weights through local inference on new samples, in order to improve the model's responsiveness to new distribution data, meanwhile solving the problem of uneven importance of new data. In addition, we adopt a one-class support vector-based outlier removal method to tackle the self-poisoning problem. In comprehensive experiments on seven high-dimensional data sets, ADTCD achieves an AUC improvement of 12.46% compared to the state-of-the-art streaming anomaly detection methods. Our future direction will focus on exploring the concept-drift problem using methods beyond autoencoders.**

***Index Terms*—Anomaly detection, concept drift, network security, time series.**

I. Introduction

IN THE network security field, anomaly detection is a method to detect unknown behavior based on the established normal behavior model, which has been the key technology to maintain the security of the Internet of Things (IoT) [1], [2], [3]. Deep learning algorithms have made great progress in improving the performance of anomaly detection in recent years [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15]. However, most of anomaly detection approaches based on deep learning algorithms are offline, and usually conform to the independent identically distribution (i.i.d.) assumption [4], [16], [17]. The i.i.d. assumption is invalid in the face of evolving data streams. More specifically, for the incoming normal streaming data that no longer conform to the preconstructed normal behavior model, the offline anomaly detection approaches will misclassified it as attack data, and generate false alarms. The above phenomenon is often referred as concept drift [18]. In offline anomaly detection approaches, the normal behavior model needs to be retrained by combining the original data oriented to the concept drift. Obviously, it inevitably consumes a large amount of time and resources.

In IoT, data collected by sensors is nonstationary, and the distributions of which change over time and special situation, such as, the traffic flow suddenly decreased during the morning rush hour due to the traffic control and the increase in water flow and velocity because of the cleanup of the equipments in water treatment plant. As a result, resolving such concept drift problem is a crucial task in IoT.

Ensemble learning is an usual concept drift processing method that uses the difference of data to train multiple different weak learners, and combines multiple different weak learners to construct a strong learner to make decisions [19]. However, this method has the defect of not responding in time when the data distribution changes. Recently, incremental learning methods based on deep neural networks (DNNs) have been proposed. It allows models adaptive to the new data distribution without retrained from the entire accumulated data [20], [21], [22], [23]. Many methods designed for concept drift assume that only the most recent data points are considered to always contain the most useful information, no matter how the data stream evolves [20], [23]. However, recent data streams may contain information with varying degrees of importance, such as new data that has already appeared

Manuscript received 29 March 2023; accepted 6 April 2023. Date of publication 10 April 2023; date of current version 7 September 2023. This work was supported in part by the Natural Science Foundation of Shandong Province under Grant ZR2021MF132 and Grant ZR2020YQ06; in part by the National Natural Science Foundation of China under Grant 62172244; in part by the National Major Program for Technological Innovation 2030-New Generation Artifical Intelligence under Grant 2020AAA0107700; in part by the Innovation Ability Pormotion Project for Small and Medium-Sized Technology-Based Enterprise of Shandong Province under Grant 2022TSGC2098; in part by the Pilot Project for Integrated Innovation of Science, Education and Industry of Qilu University of Technology (Shandong Academy of Sciences) under Grant 2022JBZ01-01; and in part by the Taishan Scholars Program under Grant tsqn202211210. *(Corresponding author: Dawei Zhao.)*

Lijuan Xu, Xiao Ding, Dawei Zhao, and Xin Li are with the Shandong Provincial Key Laboratory of Computer Networks, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan 250014, China (e-mail: xulj@sdas.org; dingxiao_2020@163.com; zhaodw@sdas.org; lixin@sdas.org).

Haipeng Peng is with the Information Security Center, State Key Laboratory of Networking and Switching Technology, and the National Engineering Laboratory for Disaster Backup and Recovery, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: penghaipeng@bupt.edu.cn).

Digital Object Identifier 10.1109/JIOT.2023.3265964

has a lower significance. Consequently, blindly updating the model incrementally with the recent data by the same degree could lead to suboptimal results. Inevitably, the data streaming used for incremental learning may contain a small number of anomalies. As the increase of anomalies, the self-poisoning problem will lead to suboptimal results of the model [21].

Motivated by the above, we propose an *Adaptive* anomaly *Detection* approach *Toward Concept Drift* ADTCD in IoT. ADTCD adopts teacher-student model to address the latency problem during the model update. Aiming to the suboptimal problem caused by uneven importance of data streaming, ADTCD constructs a dynamically adjusting weights algorithm according to local inference on new samples. In addition, an anomalies elimination method used for incremental training data is proposed, to avoid self-poisoning caused by self-updating of the deep model. In conclusion, ADTCD has the following contributions.

1) We propose an adaptive teacher–student model based on autoencoder (AE) toward concept drift and adopt a dynamically adjusting parameters algorithm for improving the model update efficiency. A teacher model is pretrained on a sufficiently large data set and then a student model with small-scale parameters is guided on the incoming data. Taking the advantages of small parameter scale and fast training speed of student model, we promote the retraining speed of newly distributed data. Furthermore, we improve model weights calculating of student model through local inference on new samples, to improve the model's responsiveness to newly distributed data.
2) We propose an anomalies elimination method-based one-class support vector machine (SVM) to avoid the self-poisoning problem caused by self-training of a single deep learning model. The method learns a maximum-margin hyperplane that separates normal classes from anomalies, to remove anomalies from the training data in the retraining phase.
3) We verify the universality and validity of ADTCD by comparing with five state-of-the-art anomaly detection methods on seven data sets with known and unknown drifts accordingly. The experimental results show that the overall efficiency of ADTCD outperforms offline anomaly detection methods, including EncDec-AD [24], DAGMM [16], REBM [17], USAD [25], and online anomaly detection methods toward concept drift, such as ARCUS [20], MemStream [26], and OGMMF-VRD [27].

II. Related Work

According to the detection principle, anomaly detection methods can be divided into rules-based methods [28], [29], [30], machine-learning-based methods [31], [32], [33], and deep-learning-based methods [13], [14]. With the continuous growth of data dimension and data scale, DNN has achieved good performance in anomaly detection. Existing deep-learning-based anomaly detection techniques can be roughly categorized as offline deep anomaly detection and anomaly detection with drift. The offline deep anomaly detection trains a deep anomaly detection model to detect data with unknown distribution without considering concept drift. While anomaly detection with drift updates the deep learning model in real time to fit data distribution changes.

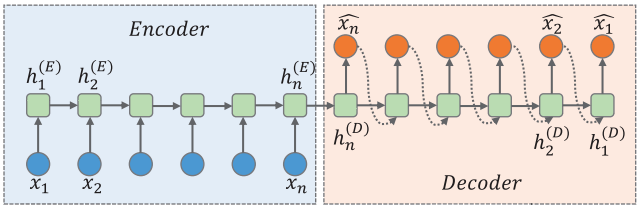

Fig. 1. Schematic diagram of the EncDec-AD method.

A. Offline Deep Anomaly Detection

With the high-speed development of DNN technology, various anomaly detection techniques have emerged [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. Deep learning models, including generative adversarial networks (GAN) [34], [35], AEs, long short-term memory neural network (LSTM) [36], gated recurrent unit (GRU) [37], Transformer [6], [38], Autoformer [39], deep variational graph convolutional recurrent network (GCN) [12], and other methods, have attracted advanced research in the field of anomaly detection. GAN has been proven effective in discovering normal patterns of data streaming, while the training phase of it costs high computation and slow convergence. LSTM, GRU, Transformer, Autoformer, and GCN focus on detecting multivariate time series.

In deep learning, AE has achieved remarkable performance in the field of anomaly detection. AE, usually trained on data samples without anomalies, utilizes reconstructed error on anomalous events to distinguish anomalies. This article introduces the principles of AE-based anomaly detection methods using the EncDec-AD approach as an example. EncDec-AD [24] is a popular LSTM-based AE method for anomaly detection in time series. Fig. 1 shows the EncDec-AD method. Let $X = [x_1, x_2, \ldots, x_n]$ be a time series of length n, and the hidden state $h_n^{(E)}$ obtained by the encoder can be expressed as

$$h_n^{(E)} = \text{LSTM}\left(\left[x_n; h_{n-1}^{(E)}\right]\right). \tag{1}$$

In the decoding phase, the time sequence is decoded in reverse order, i.e., the reconstruction of input $X = [x_1, x_2, \ldots, x_n]$ to $\hat{X} = [\hat{x_n}, \ldots, \hat{x_2}, \hat{x_1}]$. So the reconstruction value of each input data is as shown in (2), and the corresponding hidden state of the decoder is as shown in (3)

$$\hat{x_n} = Wh_n^{(D)} + b \tag{2}$$

$$h_{n-1}^{(D)} = \text{LSTM}\left(\left[\hat{x_n}; h_n^{(D)}\right]\right) \tag{3}$$

where W and b are weight and bias, respectively, both of which are trainable parameters. The anomaly score is calculated using the reconstruction error $e_n = \hat{x_n} - x_n$.

In addition, there are also other AE-based anomaly detection methods. DAGMM [16] proposes an unsupervised anomaly detection method that includes a compression network and an estimation network. It uses a compression network to generate

a low-dimensional representation for each input data point, and then uses an estimation network to estimate the sample energy. USAD [25] constructs an encoder–decoder architecture within an adversarial training framework for anomaly detection, by learning to amplify the reconstruction error of anomalous data instances during adversarial training of two AEs sharing an encoder network. RAMED [40] researches the decoders with different lengths to extract time series information captured at multiresolution, for mitigating error accumulation during decoding. Above AE-based methods are all designed for offline anomaly detection. Our adaptive teacher–student model based on AE toward concept drift, is fundamentally different from above AE-based offline methods.

B. Anomaly Detection With Drift

Based on the principle of anomaly detection with drift, we divide it into general method and deep method.

1) General Anomaly Detection With Drift: General anomaly detection with drift have made some progress in processing data streams, while they primarily focus on reducing computational overhead by reducing feature dimensionality. STORM [41] detects distance-based outliers in data streams, by processing a sliding window model to detect anomalies in the current window. DILOF [42] improves the local outlier factor (LOF), and employs a density-based sampling algorithm to summarize past data, without prior knowledge about the data distribution. RRCF [43] focuses on the anomalies for dynamic data streams through the lens of random forests. DAMP [44] extends the time series discord algorithm to realize detection on a data stream. MStream [45] utilizes locality-sensitive hash functions (LSH) to process streaming data, and combines principal component analysis (PCA), information bottleneck (IB), and AEs to map the original features into a low-dimensional space. SAND [46] extends K-shape to handle distribution drift.

2) Anomaly Detection With Drift Based on Deep Learning: Anomaly detection with drift based on deep learning adapts DNN-based models to evolving data streams by using efficient approaches [47]. Ensemble learning or incremental learning methods are two effective approaches toward concept drift in existing research. In this section, we focus on approaches that detect anomalies in data streams containing concept drift. Typical ensemble learning methods, such as [19], divide stream data into fixed-size data blocks, then build individual classifiers on continuous training data blocks, last adopt a heuristic replacement strategy to replace the classifier with the worst performance. Toward concept drift in streaming data, MemStream [26] applies denoising AE for feature extraction and sets up a memory module to learn the dynamic trend of data, and if it detects that the data stream is completely different from historical data, it retrains the model. INSOMNIA [21] reduces the delay of model update in an incremental learning method and obtains estimated labels in an active learning way. IADM [22] embeds attention weights into the Fisher information matrix to implement an adaptive deep model. ARCUS [20] proposes the concept of a model pool and dynamically adjusts it to adapt to concept drift by continuously monitoring the performance of the model pool. Oliveira et al. [27] proposed an adaptive Gaussian mixture model approach (OGMMF-VRD) that stores previous GMMs in a pool and utilizes a concept-drift test to detect any occurrence of concept drift. In the event that concept drift is detected, a new GMM is created and replaces the previous one. Wahab [48] discussed the use of PCA technique to detect the occurrence of concept drift by comparing the variance of the features observed at the current time with that observed at the previous time. If a significant change occurs, it is considered as concept drift, and the model is updated online. While these researches pioneer anomaly detection methods in the context of concept drift, they only focus on the local information of the latest data points and ignore the global uneven importance of data streaming, reducing the detection performance of deep learning algorithms on learning the distribution of stationary streaming data. Table I shows the comparative results of related work on drift anomaly detection.

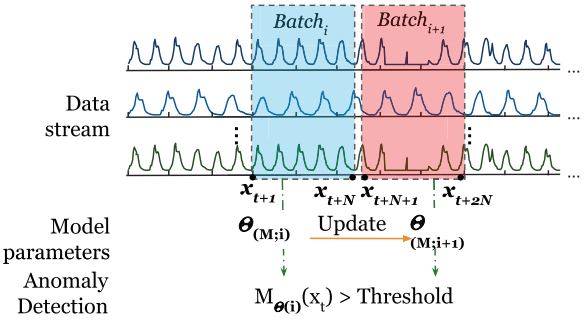

Fig. 2. Anomaly detection in evolving data streams.

III. Preliminaries

A. Problem Formulation

Let $X = \{\ldots, x_{t-1}, x_t, x_{t+1}, \ldots\}$ be a sequence of data points of unknown length arriving in a streaming manner. Each data point $x_t = (x_{t1}, \ldots, x_{td})$ consists of d attributes or dimensions. Our purpose is to calculate an anomaly score for each data point in the sequence of data points. As the distribution of normal data evolves, so does the distribution of anomalous data. Therefore, we require an anomaly detection approach that can adapt to evolving data distributions. Given an anomaly detection model M with parameters Θ, computes the anomaly score $S = \{\ldots, M_{\Theta(x_{t-1})}, M_{\Theta(x_t)}, M_{\Theta(x_{t+1})}, \ldots\}$ for each data point of a sequence X of data points. The model parameters Θ of the unsupervised model M need to be continuously updated, according to the new data distribution to adapt to the evolving data stream. Specifically, the approach first uses a batch of streaming data Batch$_i$ for model inference, and updates the model parameters $\theta_{(M;i)}$ based on the inference results. We then consider data points with anomaly scores above a threshold to be anomalies. Fig. 2 shows an approach for detecting anomalies in streams containing concept drift.

B. Concept Drift

Changes in the environment, equipment wear and other factors cause the distribution of data to change over time,

TABLE I
COMPARATIVE RESULTS OF RELATED WORK ON DRIFT ANOMALY DETECTION

Method	Adapted drift type	Method type	Advantages	Disadvantages
STORM	Abrupt, Recurrent	Sliding window	Drift samples can be detected in a short time	Insufficient use of past knowledge.
DILOF	Abrupt, Gradual, Recurrent	Density-based sampling	The memory space occupation is small.	Long running time.
RRCF	Abrupt, Gradual	Random cut forest sketch	High tolerance for exceptions	The effect of running time is not considered.
DAMP	Abrupt, Incremental	Discord Aware Matrix Profile	High time efficiency	The processing effect of gradual and recurrent drift needs to be improved.
MStream	Recurrent	Hash function	High time efficiency.	Rely on feature engineering
SAND	Abrupt, gradual	Sliding window	The model is updated incrementally, which can adapt to distribution drift.	High resource consumption
MemStream	Abrupt	Memory update strategy	High time efficiency.	Rely on feature engineering
INSOMNIA	Abrupt, gradual, Incremental	Incremental learning	Active learning reduces the time delay problem.	Low time efficiency
IADM	Incremental	Incremental learning	Updating models while preserving prior knowledge to overcome catastrophic forgetting	Failure to address self-poisoning
ARCUS	Abrupt, gradual, Incremental, Recurrent	Dynamic model pool	Strong self-optimization ability.	High time complexity
OGMMF-VRD	Abrupt, gradual, Incremental, Recurrent	Online Gaussian Mixture Model with Noise Filter	Adapt to various types of conceptual drift.	High time cost

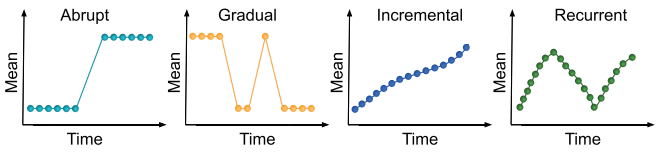

Fig. 3. Four types of conceptual drift.

which is called concept drift. This change can be defined in terms of probability [49]. Assume that the joint probability of data points X and their label y arriving at time point t is $p^t(X, y)$. For data points at two different time points, if $p^t(X, y) \neq p^{t+1}(X, y)$, it means that concept drift has occurred. Concept drift can be divided into four forms: abrupt, gradual, incremental, and reoccurring, according to the form of data distribution over time [50]. Fig. 3 presents four types of concept drift due to different changes in the mean of the data. As the figure shows, concept drift can persist over a long period of time, or it can occur at a precise point in time. Therefore, a general approach is needed to deal with different concept drifts.

IV. METHODOLOGY

In this section, we introduce ADTCD, an adaptive anomaly detection approach toward concept drift in IoT. Anomaly detection models should accommodate the emergence of new behaviors without losing the ability to infer older behaviors. ADTCD trains a teacher model on historical data with known old behaviors, that guides the student model to train on each new batch of data, and then updates the student model over time. ADTCD uses knowledge distillation as its central underlying learner, and learns the overall distribution characteristics of streaming data through a teacher–student model, for improving the ability of anomaly detection models to cope with concept drift. ADTCD judges the importance of data in real time and incrementally update the model according to the importance. In addition, in order to avoid the self-poisoning problem caused by self-training of DNN models, it adopts one-class SVM to remove anomalies.

A. Overview

ADTCD initially learns an anomaly detection model, acting as a teacher model, from historical normal data. The teacher model that learns a known data distribution from historical data, provides instructive information for subsequent student models. Meanwhile, a one-class SVM, acts as a label estimator, is trained using historical data, to remove the anomalies of the training data in the incremental training phase. Guided by the teacher model, the student model performs inference on a batch of data stream, and then is updated with the new batch of data stream. The overall procedure of ADTCD is outlined in Algorithm 1 and Fig. 4. ADTCD consists of initialization, anomaly detection, model inference, and model update phases. In the initialization phase, the normal historical data is used to train the teacher anomaly detection model and one-class SVM. The teacher model trained with abundant data, has the different structure from the student model. In the anomaly detection phase, new data streams are consumed continuously and processed in batches of equal size. Each batch of data streams is used for training a student model under the guidance of the teacher model. The student model performs anomaly detection by the operations, including calculating the anomaly score of each instance of streaming data, selecting the data points whose anomaly score is lower than the threshold, and using a one-class SVM to remove these data points, for reducing the impact of anomalies on incremental training. The model inference phase estimates the reliability of the student model

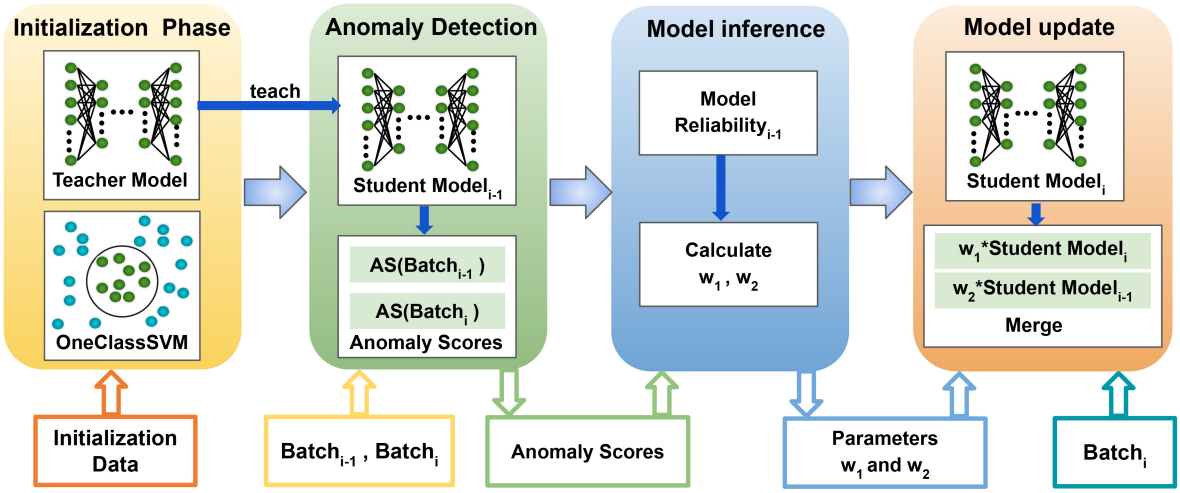

Fig. 4. Overview of ADTCD.

Algorithm 1: Overall Procedure of ADTCD

Input: initial historical normal data $X_{initial}$, a data stream D, a teacher model TM, a student model SM, a one-class support vector machine model SVM, a reliability threshold τ_R

Output: anomaly score set S of the current batch of data

Initialize a teacher model TM from the initial historical normal data $X_{initial}$;
Initialize a SVM from the initial historical normal data $X_{initial}$;
for *each batch B of data points from D* **do**
    /* 1. Anomaly score calculation */
    $S_B, B_{normal'} \leftarrow SM_{i-1}(B)$;
    $S_{B_{i-1}} \leftarrow SM_{i-1}(B_{i-1})$;
    $B_{normal} \leftarrow SVM(B_{normal'})$;
    /* 2. Model inference */
    $R_{SM_{i-1}} \leftarrow Hoffding(S_B, S_{B_{i-1}})$;
    /* 3. Train a new student model */
    $SM_i \leftarrow SM(B_{normal})$;
    /* 4. Model update */
    $w_1, w_2 \leftarrow (R_{SM}, \tau_R)$;
    $SM_{new} \leftarrow (w_1 \times SM_i + w_2 \times SM_{i-1})$;
    return S;
end

for the current batch. Next, the model update phase assigns importance to the current batch of data based on the reliability of the model, and then updates the model each time according to the importance of the current batch of data.

B. Initialization

The teacher–student model is formalized as follows.

Definition 1 (Teacher Model): A teacher model TM consists of an encoder E, a decoder D, and other required components that make up a specific AE.

Definition 2 (Student Model): A student model SM consists of an encoder E, a decoder D, and other required components that make up a specific AE. The SM and TM may not share the same network architecture.

Let X_{initial} be an initial training set x that contains normal behaviors.

1) Outlier Removal Method Based on One Class Support Vector Machine: To build an anomaly detection model capable of adapting to concept drift, we continuously update the model with new data. The first thing to do is to use the deep anomaly detection model to obtain the labels of the current batch, which will provide useful information for updating the model. Specifically, we need to retrain the normal data in the current batch according to an effective strategy to update the model. However, the normal data identified by the deep anomaly detection model is unreliable because there may be a small number of outliers. In order to mitigate the catastrophic impact of outliers on model updates, we employ a one-class SVM to clean outliers from the normal data obtained by the deep anomaly detection model. The one-class SVM learns a hyperplane with a maximum margin that separates the normal class from the origin in the best possible way. We adopt the one-class SVM to remove outliers in the current batch of retraining data to avoid the self-poisoning problem caused by the self-update of a single deep model.

C. Anomaly Detection

1) Training the Student Model: We implement an anomaly detection model using a knowledge distillation method based on a teacher–student network. Knowledge distillation is the transfer of knowledge learned from a larger teacher model to a smaller student model. In this section, the student model learns AE from the features of the known distribution data from the offline-trained teacher model, and of the unknown distribution data by itself.

The encoder E of the AE maps the input X into a set of latent variables Z, and the decoder D maps the latent variable Z back to the input space as a reconstruction. Let X_{cur} be a data stream of the current batch. Let SM (X_{cur}) be the reconstruction of the current batch of data in the student model, and TM(X_{cur}) be the reconstruction of the current batch of data in the teacher

model. The training objectives of the student model are

$$\ell_1 = ||\mathrm{TM}(X_{\mathrm{cur}}) - \mathrm{SM}(X_{\mathrm{cur}})||_2 \tag{4}$$

$$\ell_2 = ||X_{\mathrm{cur}} - \mathrm{SM}(X_{\mathrm{cur}})||_2 \tag{5}$$

where $||\cdot||_2$ denotes the L2-norm.

ℓ_1 minimizes the difference between the teacher model and the student model. ℓ_2 minimizes the reconstruction error of X for the student model. The final training objective is expressed as

$$\ell = \alpha||\mathrm{TM}(X_{\mathrm{cur}}) - \mathrm{SM}(X_{\mathrm{cur}})||_2 + \beta||X_{\mathrm{cur}} - \mathrm{SM}(X_{\mathrm{cur}})||_2 \tag{6}$$

where $\alpha + \beta = 1$.

α and β are used to parameterize the tradeoff between the importance of the teacher model and the importance of the student model.

2) Judging Anomaly: The reconstruction error of the model determines the anomaly score of the batch. We divide the newly arrived normal data into three parts: 1) D_1; 2) D_2; and 3) D_3. D_1 is used to train the student model. D_2 is for early stopping, where training is stopped once the model's performance on the validation set no longer improves. D_3 is the validation set. The reconstruction error vector for V_t is given by $e_t = |x_t - x'_t|$. For a normal distribution $N(\mu, \Sigma)$, where the parameters μ and Σ are obtained from the statistics of the reconstruction error vectors of all points in the set D_2. For each time point X_t in the current batch time series, the anomaly score can be calculated by (7)

$$A_t = (e_t - \mu)^T \Sigma^{-1}(e_t - \mu). \tag{7}$$

The anomaly score is calculated for each time point. The data points whose anomaly score exceeds the threshold τ, that is $A_t > \tau$, is marked as abnormal, and τ is selected by the grid search method. These abnormal data points are then removed using a one-class SVM to obtain the data points for incremental training.

D. Model Inference

Since drift detection is triggered when the classification rate decreases significantly [51], a straightforward measurement for estimating model reliability is detection error rate. However, the above measurement requires knowing the label of the sequence to calculate the error rate, usually achieved by additional labeling work. Statistical methods based on Hoeffding's inequality have been widely used to detect drift points and proved effective [20], [23]. In this section, we follow the method based on Hoeffding's inequality to calculate the reliability of the model. Let $S(B_{\mathrm{cur}})$ be the set of abnormal scores for the current batch of input, and $S(B_{\mathrm{pre}})$ be the set of abnormal scores for the previous batch of input. The reliability of the model M can be expressed by the bounds of Hoeffding's inequality.

Given the independent anomaly scores S_{cur} and S_{pre} sampled from $S(B_{\mathrm{cur}})$ and $S(B_{\mathrm{pre}})$ bounded by $[S_{\min}, S_{\max}]$, (8) calculates the reliability of M

$$\mathrm{pr}\left\{|\overline{S}_{\mathrm{cur}} - \overline{S}_{\mathrm{pre}}| \geq \epsilon\right\} \leq e^{\frac{-b\epsilon^2}{(S_{\max}-S_{\min})^2}} \tag{8}$$

where $\overline{S}_{\mathrm{cur}} = (1/b)\sum_{i=1}^{b} S^b_{\mathrm{cur}}$, $\overline{S}_{\mathrm{pre}} = (1/b)\sum_{i=1}^{b} S^b_{\mathrm{pre}}$, $b = |B_{\mathrm{cur}}| = |B_{\mathrm{pre}}|$, $\epsilon = |\mathrm{avg}(S(B_{\mathrm{cru}})) - \mathrm{avg}(S(B_{\mathrm{pre}}))|$, $S_{\min} = \min(\min(S(B_{\mathrm{cur}})), \min(S(B_{\mathrm{pre}})))$, and $S_{\max} = \max(\max(S(B_{\mathrm{cur}})), \max(S(B_{\mathrm{pre}})))$.

Given the current batch B_{cur}, the reliability R_M of the model M is calculated by (9)

$$R_M = e^{\frac{-b\epsilon^2}{(S_{\max}-S_{\min})^2}}. \tag{9}$$

The reliability of the model can reflect the degree of data drift. We apply the reliability of the model to the "model update" module to further obtain the weight of the model parameters.

E. Model Update

In the model update phase, the data of the current batch and of the previous batch may be non-IID. We perform a weighted average of their parameters to update two models trained on non-IID data. In federated learning, local models trained separately on multiple non-IID data can converge to the global model by the federated averaging algorithm (fedavg) [52]. Therefore, merging two models trained on different batches can train a model that converges to the global by averaging their parameters. Algorithm 2 describes the process of dynamically adjusting model parameters.

Given two models SM_1 and SM_2, trained on different batches, such as SM_1 is trained on the current batch, and SM_2 is trained on the previous batch. Their parameters are θ_1 and θ_2, the parameters of the combined model can be calculated by (10)

$$\theta = \omega_1 * \theta_1 + \omega_2 * \theta_2 \tag{10}$$

where $\omega_1 = (1/[(1 + \gamma * e^{R_M - \tau_R})])$, $\omega_2 = 1 - \omega_1$, τ_R is the reliability threshold of the model, and γ is a predefined adjustable parameter.

Taking the student model in this article as an example, we merge the encoder and decoder parameters of the old and new student models based on (10). Specifically, we need to merge the parameters of the encoder layer by layer for both the old and new student models, and similarly for the decoder. Once merged, we can reload these parameters into the network model to obtain the updated student model.

This merging strategy helps us to find the most convergent model. The reliability of the model who is larger than the threshold, indicates that the current batch of data has a high similarity with the previous data, and the current model is still reliable. While, the reliability who is less than the threshold, indicates that the current batch of data is different from the previous data, and the current model can not make reliable inferences. When the previous model is more reliable, we decrease ω_1 to reduce the proportion of new models, since the current batch is not a newly emerged concept. When the previous model is less reliable, we increase ω_2 to raise the proportion of the new model, since the current batch is a newly concept.

V. Experiments

In this section, we evaluate on a real data set with known concept drift type and duration (INSECTS [53]) and three real

TABLE II
DETAILS OF THE DATA SETS USED FOR EVALUATION

Data set	Concept dirft type	Features	Training data length
INSECTS-Abr	Abrupt	33	10000
INSECTS-Inc	Incremental	33	10000
INSECTS-IncGrd	Incremental and gradual	33	10000
INSECTS-IncRec	Incremental and recurrent	33	10000
SWAT	Unknown	51	496800
WADI	Unknown	127	784571
BATADAL	Unknown	43	8761

Algorithm 2: Dynamic Adjustment of Model Parameters

Input: the set of abnormal scores for the current batch $S(B_{cur})$, the set of abnormal scores for the previous batch $S(B_{pre})$, model parameters θ_1 of SM_1, model parameters θ_2 of SM_2

Output: Updated Model SM

for *each batch B* **do**

/* 1. Calculation Model Reliability */

$b \leftarrow |B_{cur}| = |B_{pre}|$

$\epsilon \leftarrow |avg(S(B_{cru})) - avg(S(B_{pre}))|$

$S_{min} \leftarrow min(min(S(B_{cur})), min(S(B_{pre})))$

$S_{max} \leftarrow max(max(S(B_{cur})), max(S(B_{pre})))$

$R_M \leftarrow e^{\frac{-b\epsilon^2}{(S_{max}-S_{min})^2}}$

/* 2. Model update */

$\omega_1 \leftarrow \frac{1}{(1+\gamma * e^{R_M - \tau_R})}$

$\omega_2 \leftarrow 1 - \omega_1$

$\theta \leftarrow \omega_1 * \theta_1 + \omega_2 * \theta_2$

return model SM;

end

data sets with unknown concept drift type and duration, including secure water treatment testbed (SWAT), water distribution testbed (WADI), and battle of the attack detection algorithms (BATADALs). We extensively evaluate the performance of ADTCD on seven data sets. The results are summarized as follows.

1) ADTCD outperforms the five unsupervised anomaly detection algorithms in terms of AvgAUC on seven data sets.
2) ADTCD is able to accurately detect the real concept drift points.
3) The three main techniques, including an adaptive teacher–student model based on AE toward concept drift, a dynamically adjusting parameters algorithm, and an anomalies elimination method-based one-class SVM, adopted in ADTCD are all very effective to make the accuracy improve.
4) ADTCD has the effect of balancing performance and time efficiency.
5) ADTCD is robust to changes in the value of its own hyperparameters.

A. Experiment Setup

1) Data Sets: The details of the seven multivariate concept drift data sets are described in Table II.

1) *Data Sets With Known Drifts:* INSECTS [53] is a concept drift data set that uses optical sensors to collect data on different species in a nonsmooth environment over a period of about three months. Temperature is a significant factor in the flight behavior of insects. INSECTS obtained four data sets with different concept drift types by varying the temperature. The four data sets with different conceptual drift types include, "Abrupt" (INSECTS-Abr), "Incremental" (INSECTS-Inc), "Incremental and Gradual" (INSECTS-IncGrd), and "Incremental and Recurrent" (INSECTS-IncRec).
2) *Data Sets With Unknown Drifts:* SWaT is a test platform for safe water treatment established by iTrust Research Center. SWaT consists of six stages, each process stage includes sensors, actuators such as pumps and valves. SWaT collected sensor and actuator values for 11 consecutive days of operation, including seven days of normal operation and four days of attack scenarios. WADI is an extension of SWaT, designed by iTrust Research Center. The WADI test stand includes water treatment, storage and distribution. WADI collected sensor and actuator values for 16 consecutive days of operation, including 14 days of normal operation and two days of attack scenarios. BATADAL is an attack detection competition organized by iTrust Center and its international partners. BATADAL collected data for one year of normal operations and approximately six months of data containing attacks.

2) Compared Algorithms: We compare ADTCD with streaming variants of three state-of-the-art anomaly detection approaches based on RNNs, including DAGMM [16], REBM [17] and EncDec-AD, an offline anomaly detection algorithms USAD [25], and three flow anomaly detection algorithm ARCUS [20], MemStream [26], and OGMMF-VRD [27]. The hyperparameters for above approaches are set to the default values provided by this article.

3) Evaluation Metrics: We do not use a threshold-based evaluation method in this article because the anomaly score

TABLE III
PERFORMANCE COMPARISON OVER ALL DATA SETS. RESULTS IN BOLD ARE THE HIGHEST AVGAUC IN EACH DATA SET

Methods	known drifts				Unknown drift		
	INSECTS-Abr	INSECTS-Inc	INSECTS-IncGrd	INSECTS-IncRec	SWAT	WADI	BATADAL
USAD	0.082	0.089	0.070	0.080	0.756	0.458	0.066
sEncDec-AD	0.423	0.402	0.716	0.478	0.715	0.603	0.709
sDAGMM	0.401	0.472	0.584	0.603	0.501	0.510	0.50
sREBM	0.471	0.383	0.575	0.491	0.296	0.465	0.853
ARCUS	0.558	0.621	0.641	0.634	0.671	0.602	0.723
OGMMF-VRD	0.399	0.539	0.569	0.653	0.769	0.632	0.597
MemStream	0.587	**0.721**	0.587	0.648	0.746	0.528	0.704
ADTCD	**0.588**	0.612	**0.888**	**0.888**	**0.808**	**0.644**	**0.894**

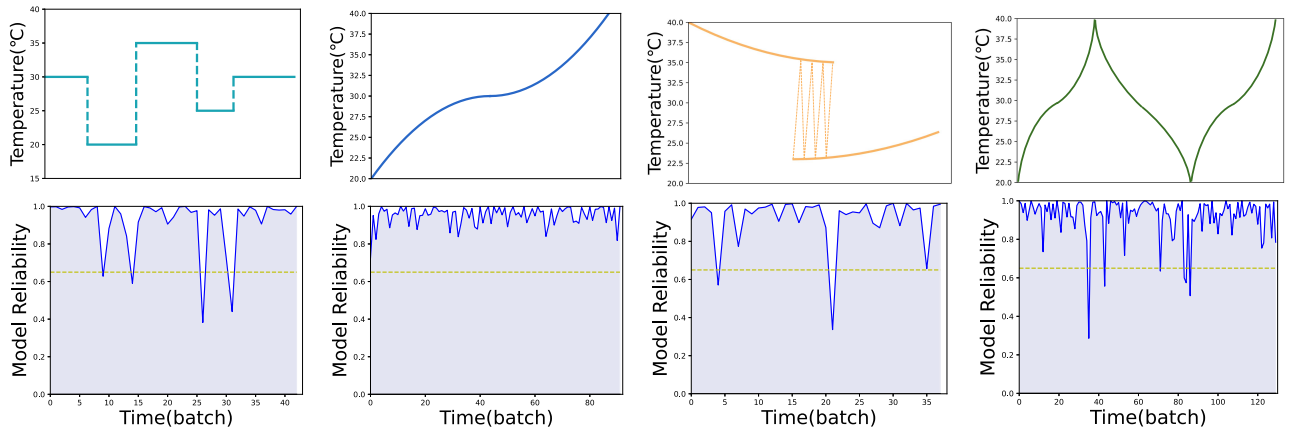

Fig. 5. Model reliability inferenced by ADTCD in concept drift data sets. (a) INSECTS_Abr. (b) INSECTS_Incr. (c) INSECTS_IncrGrd. (d) INSECTS_IncrRecr.

threshold used to detect anomalies may vary depending on the context. The evaluation metric used to determine the performance of the detection model is the average Area Under Receiver Operating Characteristic AvgAUC. AvgAUC represents the real-time AUC value of the model, reflecting the overall classification performance of the model. It is defined as follows:

$$\text{AvgAUC} = \frac{1}{T}\sum_{t=1}^{T}\text{AUC}_t \tag{11}$$

where T represents the number of all time periods, and AUC_t represents a real-time AUC value of a model at time t.

B. Overall Performance

We compared the AvgAUC values of ADTCD with other algorithms on all data sets. As shown as Table III, ADTCD achieves the highest AvgAUC in anomaly detection on most data sets. As expected, methods trained offline perform poorly and are completely failing to adapt to concept drift over time. ADTCD outperforms several streaming models in terms of AvgAUC on three data sets with known drift and three data sets with unknown drift. However, the performance of ADTCD on INSECTS-Inc is worse than the best compared algorithms. To explore the cause of this phenomenon, we consider the accuracy of the model reliability.

C. Model Reliability

Fig. 5 shows the trends of ADTCD reliability by evaluating on four real concept drift data streams. In the INSECTS data set, temperature affects changes in insect flight behavior. At this point, temperature can be described as a concept. As concept drift occurs, the reliability of the model decreases. We use this change to track the correctness of the reliability trend of the ADTCD. The reliability of ADTCD at the four abrupt drift points of INSECTS-Abr has dropped sharply, which prove that ADTCD has the ability to detect abrupt drift. We observe that the reliability trend shows a smooth fluctuation on the INSECTS-Inc data set. Therefore, this may be the reason for the underperformance of the ADTCD model on the INSECTS-Inc data set. As concept drift occurs incrementally, the anomaly scores fluctuate in a smaller interval, making the model less sensitive to new concepts. We consider that it is related to the performance of our AE algorithm. We find an interesting thing, the reliability on INSECTS-IncGrd shows two unexpected declines, which may be caused by other changes in the environment, such as light or humidity. In addition, the reliability of ADTCD at the drift point

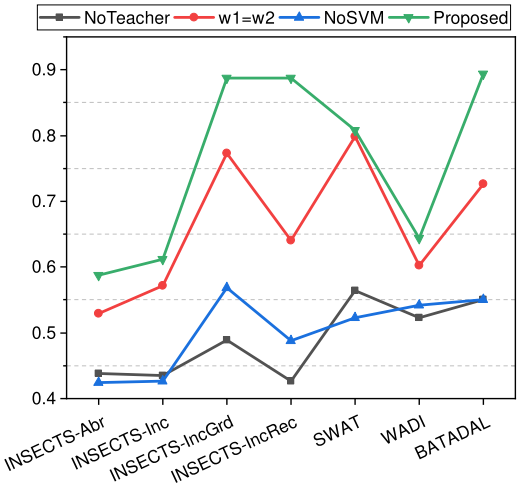

Fig. 6. Ablation study results.

of INSECTS-IncRec has a corresponding decline, and proves ADTCD has the excellent detection performance for recurrent drift. Above observations that ADTCD can detect and effectively adapt to real concept drift in time.

D. Ablation Analysis

Using seven data sets, we investigate the effects of the three main techniques used in ADTCD. For evaluating the validity of the teacher–student mechanism, we only use the student model for incremental training, the ℓ_2 loss function for training the student model. This means that the model only learns the distribution from recent data streams, and does not obtain knowledge from data streams with known distributions. Fig. 6 shows that the teacher–student mechanism improves accuracy by 27.08% compared to training with only the student model. The reason for the superior results is that the teacher model is able to learn useful knowledge from the data streams with the known distribution, thereby helping the student model to adapt to new behaviors without losing the ability to reason about old behaviors. This result confirms the effectiveness of using the teacher model to guide the student model in making decisions.

In order to evaluate the effectiveness of a dynamically adjusting parameters algorithm according to the reliability of the models as merging models, we compare it to the performance of parameters with equal weights. As shown in Fig. 6, the dynamically adjusting parameters algorithm achieves a higher AvgAUC, which is 9.73% higher than that of parameters with equal weights. The reason for that is ADTCD can effectively balance the recognition ability of known behaviors and unknown behaviors. It is concluded that dynamically adjusting parameters algorithm is applicable to concept drift more effectively.

For evaluating the efficacy of an anomalies elimination method-based one-class SVM, we compared it with a variant without SVM. As shown as Fig. 6, the AvgAUC of our anomalies elimination method is 25.7% higher than that the variant ADTCD-NoSVM. Self-poisoning reduces the detection performance of ADTCD. Therefore, it is necessary to use the anomalies elimination method-based one-class SVM, to solve the self-poisoning problem in anomaly detection approach toward concept drift.

E. Runtime

Fig. 7 demonstrates the time taken to process four phases for a batch of 1000 data points on each data set. These four phases are initialization, anomaly detection, model inference, and model update. During the initialization phase, the size of epochs of the teacher model can be flexibly controlled at will. In our experiment, the epoch of the teacher model is set to 50. The detailed settings of the training data are described in Table II, is defined by the user. In the model inference phase, the size of epochs is determined by the batch size of the current batch of data. If the batch size is too small while the epoch number is too large, overfitting may occur. Therefore, we set the epoch of this phase be 2. The running time of anomaly detection, model inference, and model update is shown in Fig. 7(b). Compared with the offline training method, that is incorporating new data into existing historical data for retraining, ADTCD reduces a lot of time overhead. ADTCD takes less than 4 s to process a batch of 1000 data points. In conclusion, ADTCD can keep a balance between performance and time efficiency.

F. Parameter Sensitivity Study

We perform the sensitivity study on the parameter α in the loss function and the reliability threshold τ_R of the model. A low α means that the model attachs importance to the latest data distribution. Specially speaking, when α is 0, only the student model is used for training. While, $\alpha = 1$ means that ADTCD only learns normal patterns of data with known distribution and pays little attention to data with unknown distribution. The AvgAUC with $\alpha = 0$ and $\alpha = 1$ is shown in Fig. 8(a), both special cases exhibiting extreme performance degradation. This result shows that it is beneficial for both the student model and the teacher model to improve the detection efficiency, and increase of the value of α may help improve the accuracy of the model. For τ_R, Fig. 8(b) shows that the AvgAUC peaks at 0.95. The accuracy of the model may be improved by adjusting the reliability threshold τ_R. Therefor, ADTCD is robust to changes in the value of its own hyperparameters α and τ_R.

VI. Conclusion

In this article, we propose ADTCD, an adaptive anomaly detection approach toward concept drift in IoT. Inspired by the challenges of the low efficiency of model updates, the uneven importance of new data, and the self-poisoning due to model self-updates, ADTCD adopts a teacher–student model based on AE toward concept drift. Then, ADTCD constructs a dynamically adjusting parameters algorithm to judge the importance of data in real time and assign weights to model parameters according to the importance, for improving the model's responsiveness to new distribution data, meanwhile addressing the problem of uneven importance of new data. We

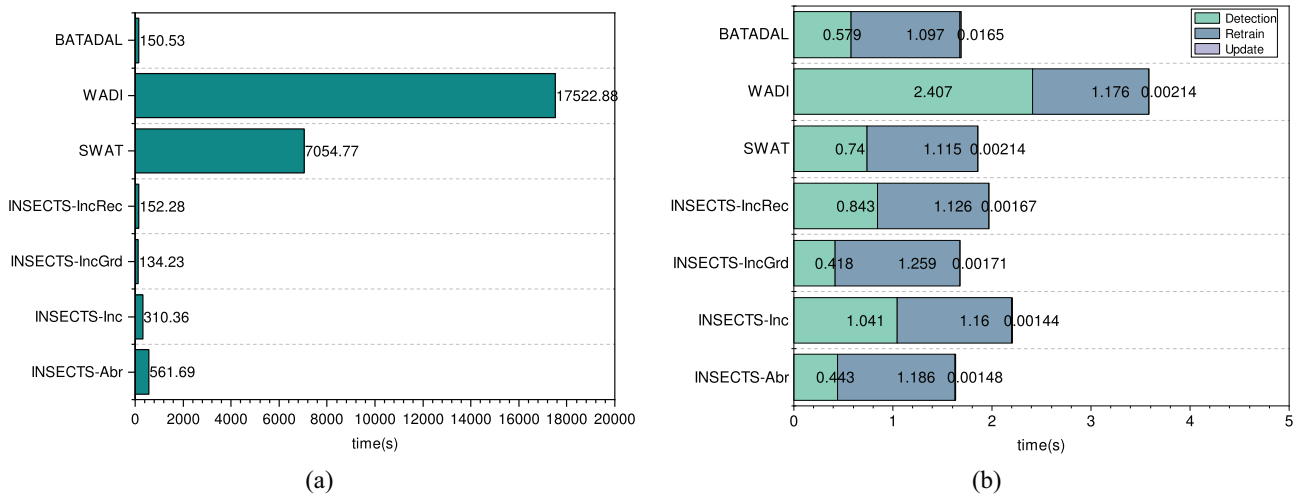

Fig. 7. Processing time breakdown results. (a) Running time of the initialization phase. (b) Running time of the detection phase.

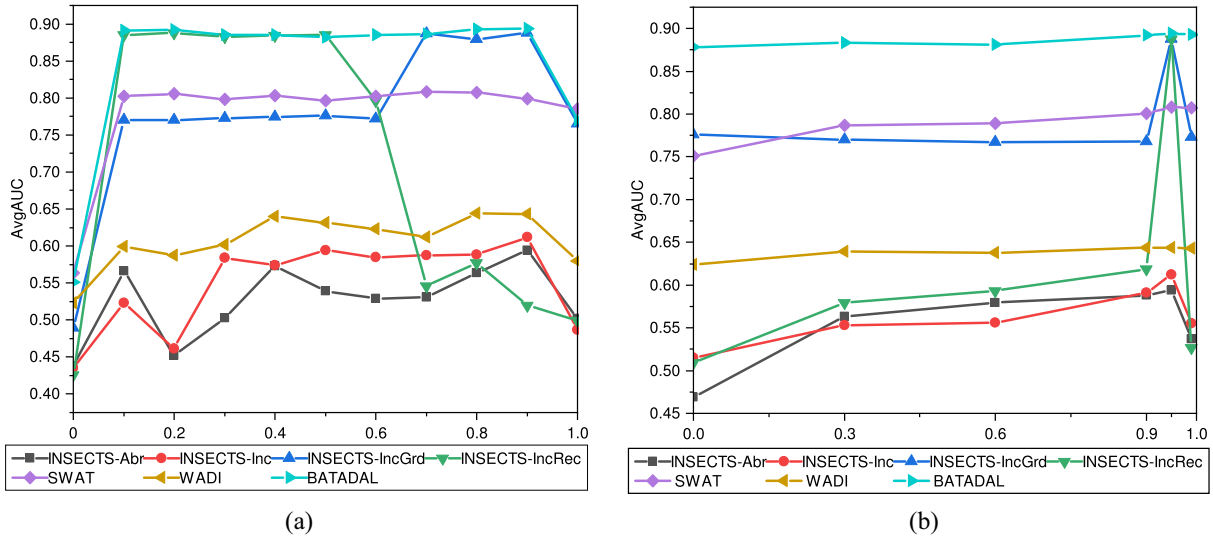

Fig. 8. Sensitivity analysis results. (a) Parameter α. (b) Reliability threshold τ_R.

verify the universality and validity of ADTCD through comparing with five state-of-the-art anomaly detection approaches on seven data sets with known and unknown drifts accordingly. The experimental results demonstrate that the overall efficiency of ADTCD outperforms offline anomaly detection methods, including EncDec-AD [24], DAGMM [16], REBM [17], and USAD [25], and online anomaly detection methods toward concept drift, such as ARCUS[20], MemStream [26], and OGMMF-VRD [27]. Finally, Aiming at the self-poisoning problem, ADTCD adopts an one-class SVM-based anomalies elimination method to remove anomalies from the training data in the retraining phase. Our experimental results show that ADTCD outperforms the state-of-the-art anomaly detection methods in terms of AvgAUC, and has the ability of accurately detecting the real concept drift points. The three main techniques, including an adaptive teacher–student model based on AE toward concept drift, a dynamically adjusting parameters algorithm, and an anomalies elimination method-based one-class SVM, applied in ADTCD are all highly effective in improving accuracy. In addition, the time efficiency analysis experiment shows that ADTCD has the effect of keeping the balance between performance efficiency and time efficiency. In the end, we verifies that ADTCD is robust to changes in the value of its own hyperparameters α and τ_R.

There are primarily two limitations in this study. First, the industrial control scenario used in the experiments is relatively homogeneous. Second, our anomaly detection model only uses normally labeled data and pays little attention to abnormal data due to the scarcity of abnormal data. In the future, we will utilize data sets from a variety of domains, such as smart grid, natural gas transportation, and address the limitations mentioned above. In the future, we intend to develop some interesting aspects. These include using self-supervised strategies such as contrastive learning to obtain more robust models for the problem of anomaly label scarcity. ADTCD only investigated deep anomaly detection models based on AEs, which can be extended to other models in the future. ADTCD is based on a single anomaly detector, which can improve decision-making ability by constructing multiple student models with different network structures.

REFERENCES

[1] D. Zhao, L. Wang, Z. Wang, and G. Xiao, "Virus propagation and patch distribution in multiplex networks: Modeling, analysis, and optimal allocation," *IEEE Trans. Inf. Forensics Security*, vol. 14, pp. 1755–1767, 2019.

[2] W. Li, W. Hu, N. Chen, and C. Feng, "Stacking VAE with graph neural networks for effective and interpretable time series anomaly detection," 2021, *arXiv:2105.08397*.

[3] D. Zhao, G. Xiao, Z. Wang, L. Wang, and L. Xu, "Minimum dominating set of multiplex networks: Definition, application, and identification," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 12, pp. 7823–7837, Dec. 2021.

[4] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, New York, NY, USA, 2019, pp. 2828–2837.

[5] L. Dai et al., "SDFVAE: Static and dynamic factorized VAE for anomaly detection of multivariate CDN KPIS," in *Proc. Web Conf.*, New York, NY, USA, 2021, pp. 3076–3086.

[6] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," in *Proc. 10th Int. Conf. Learn. Represent.*, 2022, pp. 1–20.

[7] L. Shen, Z. Li, and J. Kwok, "Timeseries anomaly detection using temporal hierarchical one-class network," in *Proc. NIPS*, vol. 33, 2020, pp. 13016–13026.

[8] A. Abdulaal, Z. Liu, and T. Lancewicki, "Practical approach to asynchronous multivariate time series anomaly detection and Localization," in *Proc. 27th ACM SIGKDD Conf. Knowl. Disc. Data Min.*, New York, NY, USA, 2021, pp. 2485–2494.

[9] Z. Li et al., "Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding," in *Proc. 27th ACM SIGKDD Conf. Knowl. Disc. Data Min.*, New York, NY, USA, 2021, pp. 3220–3230.

[10] C. Feng and P. Tian, "Time series anomaly detection for cyber-physical systems via neural system identification and Bayesian filtering," in *Proc. 27th ACM SIGKDD Conf. Knowl. Disc. Data Min.*, New York, NY, USA, 2021, pp. 2858–2867.

[11] X. Zhang, Y. Gao, J. Lin, and C. Lu, "TapNet: Multivariate time series classification with attentional prototypical network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 6845–6852.

[12] W. Chen, L. Tian, B. Chen, L. Dai, Z. Duan, and M. Zhou, "Deep variational graph convolutional recurrent network for multivariate time series anomaly detection," in *Proc. Int. Conf. Mach. Learn.*, vol. 162, 2022, pp. 3621–3633.

[13] A. Yehezkel, E. Elyashiv, and O. Soffer, "Network anomaly detection using transfer learning based on auto-encoders loss normalization," in *Proc. 14th ACM Workshop Artif. Intell. Security*, New York, NY, USA, 2021, pp. 61–71.

[14] K. Wang, A. Zhang, H. Sun, and B. Wang, "Analysis of recent deep-learning-based intrusion detection methods for in-vehicle network," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 1843–1854, Feb. 2023.

[15] C. Tang, L. Xu, B. Yang, Y. Tang, and D. Zhao, "GRU-based interpretable multivariate time series anomaly detection in industrial control system," *Comput. Security*, vol. 127, Apr. 2023, Art. no. 103094.

[16] B. Zong et al., "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–19.

[17] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, "Deep structured energy based models for anomaly detection," in *Proc. Int. Conf. Mach. Learn.*, vol. 48, 2016, pp. 1100–1109.

[18] G. Krempl et al., "Open challenges for data stream mining research," *ACM SIGKDD Explorations Newslett.*, vol. 16, no. 1, pp. 1–10, 2014.

[19] W. N. Street and Y. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2001, pp. 377–382.

[20] S. Yoon, Y. Lee, J. Lee, and B. S. Lee, "Adaptive model pooling for online deep anomaly detection from a complex evolving data stream," in *Proc. 28th ACM SIGKDD Conf. Knowl. Disc. Data Min.*, 2022, pp. 2347–2357.

[21] G. Andresini, F. Pendlebury, F. Pierazzi, C. Loglisci, A. Appice, and L. Cavallaro, "INSOMNIA: Towards concept-drift robustness in network intrusion detection," in *Proc. 14th ACM Workshop Artif. Intell. Security*, 2021, pp. 111–122.

[22] Y. Yang, D. Zhou, D. Zhan, H. Xiong, and Y. Jiang, "Adaptive deep models for incremental learning: Considering capacity scalability and sustainability," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2019, pp. 74–82.

[23] I. I. F. Blanco, J. del Campo-Ávila, G. Ramos-Jiménez, R. M. Bueno, A. A. O. Díaz, and Y. C. Mota, "Online and non-parametric drift detection methods based on Hoeffding's bounds," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 3, pp. 810–823, Mar. 2015.

[24] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "LSTM-based encoder-decoder for multi-sensor anomaly detection," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1–5.

[25] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "USAD: UnSupervised anomaly detection on multivariate time series," in *Proc. 26th ACM SIGKDD Conf. Knowl. Disc. Data Min.*, 2020, pp. 3395–3404.

[26] S. Bhatia, A. Jain, S. Srivastava, K. Kawaguchi, and B. Hooi, "MemStream: Memory-based streaming anomaly detection," in *Proc. ACM Web Conf.*, 2022, pp. 610–621.

[27] G. H. F. M. Oliveira, L. L. Minku, and A. L. I. Oliveira, "Tackling virtual and real concept drifts: An adaptive Gaussian mixture model approach," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 2, pp. 2048–2060, Feb. 2023.

[28] D. Yang, A. Usynin, and J. W. Hines, "Anomaly-based intrusion detection for SCADA systems," in *Proc. 5th Int. Topical Meeting Nucl. Plant Instrum., Control Human Mach. Interface Technol.*, 2006, pp. 12–16.

[29] L. Xu, B. Wang, L. Wang, X. Han, and D. Zhao, "PLC-SEIFF: A programmable logic controller security incident forensics framework based on automatic construction of security constraints," *Comput. Security*, vol. 92, May 2020, Art. no. 101749.

[30] A. Khalili and A. Sami, "SysDetect: A systematic approach to critical state determination for industrial intrusion detection systems using Apriori algorithm," *J. Process Control*, vol. 32, pp. 154–160, Aug. 2015.

[31] M. Kalech, "Cyber-attack detection in SCADA systems using temporal pattern recognition techniques," *Comput. Security*, vol. 84, pp. 225–238, Jul. 2019.

[32] L. Xu, B. Wang, X. Wu, D. Zhao, L. Zhang, and Z. Wang, "Detecting semantic attack in SCADA system: A behavioral model based on secondary labeling of states-duration evolution graph," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 2, pp. 703–715, Mar./Apr. 2022.

[33] L. Xu, B. Wang, M. Yang, D. Zhao, and J. Han, "Multi-mode attack detection and evaluation of abnormal states for industrial control network," *J. Comput. Res. Develop.*, vol. 58, no. 11, pp. 2333–2349, 2021.

[34] S. Ren, D. Li, Z. Zhou, and P. Li, "Estimate the implicit likelihoods of GANs with application to anomaly detection," in *Proc. Web Conf.*, 2020, pp. 2287–2297.

[35] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S. Ng, "MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks," in *Proc. Int. Conf. Artif. Neural Netw.*, 2019, pp. 703–716.

[36] G. Lai, W. C. Chang, Y. Yang, and H. Liu, "Modeling long- and short-term temporal patterns with deep neural networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 95–104.

[37] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, 2014, pp. 1–15.

[38] Z. Chen, D. Chen, Z. Yuan, X. Cheng, and X. Zhang, "Learning graph structures with transformer for multivariate time series anomaly detection in IoT," *IEEE Internet Thing J.*, vol. 9, no. 12, pp. 9179–9189, Jun. 2022.

[39] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *Proc. 35th Conf. Neural Inf. Process. Syst.*, 2021, pp. 1–20.

[40] L. Shen, Z. Yu, Q. Ma, and J. T. Kwok, "Time series anomaly detection with multiresolution ensemble decoding," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, 2021, pp. 9567–9575.

[41] F. Angiulli and F. Fassetti, "Detecting distance-based outliers in streams of data," in *Proc. 16th ACM Conf. Inf. Knowl. Manage.*, 2007, pp. 811–820.

[42] G. S. Na, D. H. Kim, and H. Yu, "DILOF: Effective and memory efficient local outlier detection in data streams," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2018, pp. 1993–2002.

[43] S. Guha, N. Mishra, G. Roy, and O. Schrijvers, "Robust random cut forest based anomaly detection on streams," in *Proc. 33nd Int. Conf. Mach. Learn.*, vol. 48, 2016, pp. 2712–2721.

[44] Y. Lu, R. Wu, A. Mueen, M. A. Zuluaga, and E. J. Keogh, "Matrix profile XXIV: Scaling time series anomaly detection to trillions of datapoints and ultra-fast arriving data streams," in *Proc. 28th ACM SIGKDD Conf. Knowl. Disc. Data Min.*, 2022, pp. 1173–1182.
[45] S. Bhatia, A. Jain, P. Li, R. Kumar, and B. Hooi, "MStream: Fast anomaly detection in multi-aspect streams," in *Proc. Web Conf.*, 2021, pp. 3371–3382.
[46] P. Boniol, J. Paparrizos, T. Palpanas, and M. J. Franklin, "SAND: Streaming subsequence anomaly detection," *Proc. VLDB Endow.*, vol. 14, no. 10, pp. 1717–1729, 2021.
[47] W. Li, X. Yang, W. Liu, Y. Xia, and J. Bian, "DDG-DA: Data distribution generation for predictable concept drift adaptation," in *Proc. 36th AAAI Conf. Artif. Intell.*, 2022, pp. 4092–4100.
[48] O. A. Wahab, "Intrusion detection in the IoT under data and concept drifts: Online deep learning approach," *IEEE Internet Things J.*, vol. 9, no. 20, pp. 19706–19716, Oct. 2022.
[49] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Min. Knowl. Disc.*, vol. 30, no. 4, pp. 964–994, 2016.
[50] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Wozniak, "Ensemble learning for data stream analysis: A survey," *ACM Comput. Surveys*, vol. 50, no. 2, pp. 132–156, 2017.
[51] N. Lu, J. Lu, G. Zhang, and R. L. de Mántaras, "A concept drift-tolerant case-base editing technique," *Artif. Intell.*, vol. 230, pp. 108–133, Jan. 2016.
[52] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *Proc. 8th Int. Conf. Learn. Represent.*, 2020, pp. 1–26.
[53] V. M. A. de Souza, D. M. dos Reis, A. G. Maletzke, and G. E. A. P. A. Batista, "Challenges in benchmarking stream learning algorithms with real-world data," *Data Min. Knowl. Disc.*, vol. 34, no. 6, pp. 1805–1858, 2020.

Lijuan Xu received the master's degree in computer science and technology from Shandong University, Jinan, China, in 2007.

She is currently an Associate Professor with Shandong Computer Science Center (National Supercomputer Center), Jinan. Her main research interests include network security, industrial Internet security, and computer forensics.

Xiao Ding is currently pursuing the master's degree with the School of Shandong Computer Science Center (National Supercomputer Center), Qilu University of Technology (Shandong Academy of Sciences), Jinan, China.

Her research interests include network security, industrial Internet security, and anomaly detection.

Haipeng Peng received the M.S. degree in system engineering from Shenyang University of Technology, Shenyang, China, in 2006, and the Ph.D. degree in signal and information processing from Beijing University of Posts and Telecommunications, Beijing, China, in 2010.

He is currently a Professor with the School of CyberSpace Security, Beijing University of Posts and Telecommunications. He has the coauthored 200 scientific papers. His research interests include information security, network security, and complex networks.

Dawei Zhao (Member, IEEE) received the Ph.D. degree in cryptology from Beijing University of Posts and Telecommunications, Beijing, China, in 2014.

He is currently a Professor with Shandong Computer Science Center (National Supercomputer Center), Jinan, China. His main research interests include network security, complex network, and epidemic spreading dynamics.

Xin Li received the Ph.D. degree in cyberspace security from Beijing University of Posts and Telecommunications, Beijing, China, in 2022.

He is a Lecturer with Shandong Computer Science Center (National Supercomputer Center), Jinan, China. His research interests include machine learning and software security.