# State-Transition-Aware Anomaly Detection Under Concept Drifts

Bin Li(✉) and Emmanuel Müller

TU Dortmund University, Dortmund, Germany
{bin.li,emmanuel.mueller}@tu-dortmund.de

**Abstract.** Detecting temporal abnormal patterns over streaming data is challenging due to volatile data properties and the lack of real-time labels. The abnormal patterns are usually hidden in the temporal context, which cannot be detected by evaluating single points. Furthermore, the normal state evolves over time due to concept drifts. A single model does not fit all data over time. Autoencoders are recently applied for unsupervised anomaly detection. However, they are trained on a single normal state and usually become invalid after distributional drifts in the data stream. This paper uses an Autoencoder-based approach STAD for anomaly detection under concept drifts. In particular, we propose a state-transition-aware model to map different data distributions in each period of the data stream into states, thereby addressing the model adaptation problem in an interpretable way. Our experiments evaluate the proposed method on synthetic and real-world datasets. While delivering comparable anomaly detection performance as the state-of-the-art approaches, STAD works more efficiently and provides extra interpretability.

**Keywords:** State transition · Anomaly detection · Concept drift · Autoencoder

## 1 Introduction

Anomaly detection in streaming data is gaining traction in the current big data research. Despite the high demand in a variety of real-world applications [22] (e.g., health care, device monitoring, and predictive maintenance), rare existing models show convincing performance in real-time deployment. The detection of abnormal patterns in streaming data is challenging. On the one hand, labels are unavailable or expensive to acquire in real-time, such that supervised approaches usually fail. On the other hand, the conventional batch models easily expire, while a single stationary model does not fit the ever-changing data stream.

Recently, Autoencoders have been employed for anomaly detection in an unsupervised manner [14,26]. Autoencoders are trained to reconstruct the normal data[1], such that for any unknown data instance, a high reconstruction error

[1] Unless specifically stated, instead of normally distributed data, normal data refers to the opposite of abnormal data in the anomaly detection context.


R. Wrembel et al. (Eds.): DaWaK 2023, LNCS 14148, pp. 49–63, 2023.
https://doi.org/10.1007/978-3-031-39831-5_5

indicates an anomaly. Specifically, for time series data, the temporal dependencies between data points can be captured by constructing Autoencoders using Recurrent Neural Networks (RNNs) and their variants [14,16]. Although such methods show impressive performance on time series data, they usually ignore the fact that such data is commonly collected in a streaming way and does not allow full access during the training phase. Therefore, an adaptive Autoencoder is desired, which can be initialized with a few normal data and continuously capture the latest knowledge from the real-time data stream. Another major challenge of anomaly detection in streaming data is distinguishing between abnormal patterns and concept drifts. Once the data stream drifts to a novel distribution, a stationary model trained only on outdated data may detect most of the upcoming data undesirably as anomalies.

Given the severe problems, we aim to consider the concept drift detection and anomaly detection holistically, adapt the model to the latest data distribution, and detect anomalies only concerning the temporal context where they are located. Previous concept drift detection researches focus on detecting changes of the joint probability $P(X, y)$ under a supervised setting, namely, the decision boundary changes along with the distributional changes in the input data [13]. However, for anomaly detection, the class distribution between normal and abnormal is extremely unbalanced, and labels are usually missing or delayed, so it is impractical to use traditional supervised approaches [4,11], e.g., detecting drifts based on the changes of real-time prediction error rate. Instead, the adaptation based on changes of the prior $P(X)$ will ensure the Autoencoder learns the normal data pattern from the latest data distribution.

Statistical tests are commonly used for unsupervised drift detection [13]. For instance, the two-sample tests examine whether samples from two collections are generated from the same data distribution. However, many existing methods conduct tests mostly in the original input space, which only works for linearly detectable drifts. Ceci et al. [7] introduce both PCA and Autoencoder to embed features into a latent space for the change detection in power grid data. However, they use a feed-forward Autoencoder, which does not directly capture the temporal information in the data.

In this paper, we propose STAD (State-Transition-aware Anomaly Detection). In STAD, data distribution in a time period is defined as a state. We use state transitions to model the concept drifts between periods. As Autoencoders are well-studied for non-linear time series anomaly detection, we are motivated to extend the state transition paradigm to Autoencoders. We follow the standard usage of Autoencoders for anomaly detection and novelly couple the detection of concept drifts and anomalies with the informative latent representation of Autoencoders. An existing Autoencoder can be reused when a data concept reappears in the stream. A state transition is triggered by the detection of a concept drift, and this will further guide the reuse or adaptation of Autoencoders for the next period. The states raise interpretability in understanding the decisions of Autoencoders and changes in the data stream.

## 2 Related Works

**Online Anomaly Detection.** A major category of online anomaly detection methods is based on a prediction model, which employs historical data to predict the near future. Abnormal data may not fit the normal prediction and therefore cause a large prediction error. The widely used ARIMA model in time series analysis is also used in anomaly detection [3]. However, specific adaptation strategies are to be made to use it in online fashion. The Hierarchical Temporal Memory (HTM) model [1] is designed for real-time application, while it can automatically adapt to changing statistics. One issue with models in this category is that they are usually designed for univariate data. Therefore, deep neural networks are also used recently to model higher dimensional and more complex data. [15] use LSTMs as a basic prediction model, which can capture the high-dimensional contextual information between different timestamps. [12] also employs an LSTMs-based prediction model for anomaly detection. However, their semi-supervised approach requires partial labels from the history, which is not always possible in the streaming processing scenario.

Reconstruction-based approaches train models to reconstruct the normal data so that unknown abnormal data in the test phase will cause larger reconstruction errors due to the lack of knowledge. Autoencoders are used as an unsupervised approach for anomaly detection. [26] adopts a Gaussian Mixture Model to detect anomalies from the reconstruction error. However, they use the feed-forward network, which cannot deal with inter-dependent data points as in the data stream. [14] builds the Autoencoder with LSTM units to capture temporal information. Similarly, [17] constructs the Autoencoder with Transformers. These models assume that the sequential data are generated from the same distribution. Therefore they are vulnerable to drifts. In the worst case, every data point that arrives after the drifts will be predicted as an anomaly.

**Drift Detection.** Recent drift detection approaches are well-summarized in [13]. Common processing paradigms aggregate the historical data, extract data features and conduct statistical tests. Many works contribute to the streaming data classification problem [4,18], where the real-time classification error is used as an indicator of drift detection. Unfortunately, the labels are not always immediately available in real time. On the contrary, unsupervised drift detection methods detect changes in $P(X)$, namely the distributional changes in the streaming data. Statistical tests are usually applied to detect drifts in univariate streaming data [18,20]. For multivariate streaming data, each dimension can be tested individually and aggregated afterward [19].

Finally, the model's trustworthiness and reliability are important for real-time anomaly detection, especially in safety-crucial applications. However, the interpretation of black-box anomaly detection models and complex streaming data is still under-studied. [22] interprets device anomalies by feature responsibility gained from Integrated Gradient [24]. [2] uses a graph-based framework to model recurring concepts in the data stream. None of them has a focus on the drift detection perspective.

## 3 Problem Definition

### 3.1 Terminology

**Data Stream and Concept Drift.** Let $\mathcal{X} = \{X_t\}_{t\in\mathbb{N}^*}^{D}$ be a D-dimensional data stream, where $X_t$ denotes the observation at timestamp $t$. The data stream contains unlabeled anomalies as well as distributional changes caused by concept drifts. Instead of explicitly categorizing different concept drift types [13], we uniformly consider that a concept drift occurs in the data stream between timestamps $t$ and $t+c$ if the prior probability $P_{<t}(X) \neq P_{>t+c}(X)$, where $P_{<t}$ and $P_{>t+c}$ are respectively the data distribution from the last concept drift to $t$ and from $t+c$ to the next concept drift. The period $[t, t+c]$ is the drift period, defined as the minimum period that covers the whole distributional change. The data distribution other than drift periods is assumed to be stable. Due to the lack of labels under the unsupervised setting, we only consider the prior (virtual) shifts [13] in the data stream.

**State Transition.** Imitating the automata theory, we formulate concept drifts in streaming data with a state transition model $\mathcal{M} = \langle \mathcal{X}, \mathcal{S}, \delta \rangle$ where $\mathcal{X}$ is a multivariate data stream, $\mathcal{S} = \{S_1, S_2, ..., S_N\}$ is a set of states ($N$ is the user-defined maximum number of states that can be maintained), $\delta$ is a set of transition functions $\delta : \{S_i \Rightarrow S_j\}(S_i, S_j \in \mathcal{S}, i \neq j)$. For each state $S_i = \langle P_i, AE_i \rangle (i = 1, ..., N)$, $AE_i$ is the Autoencoder trained on the current concept data, $P_i$ is the empirically estimated distribution in the Autoencoder latent space. In this work, we assume sufficient data after the concept drifts is available to learn $P_i$ and $AE_i$.

Considering that no information about the upcoming new concept is accessible, despite a potential high error rate, we still keep using the previous model for anomaly detection until the model adaptation is finished. Or in other words, the previous model is used during the upcoming *drift period*. For distributional stationary data streams where no concept drift occurs, there will be only a single state without transition, and the model reduces to a single conventional Autoencoder for stationary data.

**Anomaly.** An observed data snippet $X_t^w = \{x_{t+1}, ..., x_{t+w}\}(t, w \in \mathbb{N}^*)$ is abnormal if it significantly deviates from its temporal neighbors (data snippets in the same *state*). The significance of the deviation can be determined by thresholding or statistical techniques. Both concept drifts and anomaly snippets are distributionally deviating from their temporal neighbors. In our study, we distinguish them in terms of length. After the concept drifts, we assume that the data distribution stays stationary in the new concept for a significantly longer period. In contrast, the data stream returns to the previous distribution after a short anomaly snippet.

### 3.2 Problem Statement

Given a D-dimensional data stream $\mathcal{X} = \{X_t\}_{t\in\mathbb{N}^*}^{D}$, we aim to identify any period $[t+1, t+w]$ where the corresponding data snippet $X_t^w$ is abnormal. The detection process should be unsupervised and in real time. We also detect concept drifts in the data stream and switch to an existing Autoencoder or train a new one on the newly arrived data.

## 4 State-Transition-Aware Anomaly Detection

In this section, we propose STAD, a state-transition-aware anomaly detection model, which employs Autoencoder as the base model. The latent representations of Autoencoders are used to detect concept drifts, which consequently trigger state transitions. An overview of STAD is shown in Fig. 1.

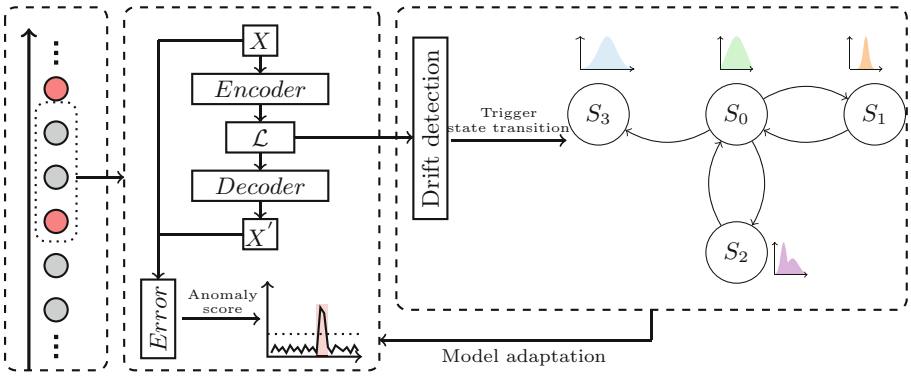


**Fig. 1.** STAD overview: The left block is a multivariate data stream, where red dots denote abnormal data points and the dashed box is a data snippet. The middle block is an conventional autoencoder-based anomaly detection module, which detects abnormal snippets from the data stream. The right block takes latent representations from the autoencoder and conducts concept drift detection, which consequently triggers state transition and model adaptation. (Color figure online)

### 4.1 Reconstruction and Latent Representation Learning

Let $f_{Enc}\colon \mathbb{R}^{w\times D} \to \mathbb{R}^H$ and $f_{Dec}\colon \mathbb{R}^H \to \mathbb{R}^{w\times D}$ be the encoder and decoder of an Autoencoder. The encoder maps a snippet $X_t^w$ of the multivariate streaming data into an $H$-dimensional latent representation $L \in \mathbb{R}^H$, while the decoder reconstructs the same format snippet $X_t'^w$ from $L$, where $w$ is the snippet length and $t, w \in \mathbb{N}^*$. A common assumption for anomaly detection using Autoencoders is that pure normal data are available for the initial model training. The reconstruction error $e_t^w = |X_t^w - X_t'^w|$ indicates the goodness of fit to the normal

**Algorithm 1.** Latent Space Drift Detection

**Input:** $\mathcal{L}_{hist}$ with maximum size $m$, $\mathcal{L}_{new}$ with maximum size $n$, minimum $\mathcal{L}_{hist}$ size $m^*$ trigger test, current state $S = \langle P, AE \rangle$, state transition model $\mathcal{M} = \langle \mathcal{X}, \mathcal{S}, \delta \rangle$

**while** stream does not end **do**
  $L_t \leftarrow$ AnomalyDetection($AE$,$X_t^{t+w}$) ▷ Get latent representation
  $\mathcal{L}_{new} \leftarrow \mathcal{L}_{new} \cup L_t$
  **if** $\mathcal{L}_{new}.size > n$ **then** ▷ Move the oldest element of $\mathcal{L}_{new}$ to $\mathcal{L}_{hist}$
    $L_{t-n+1} = \mathcal{L}_{new}.pop()$
    $\mathcal{L}_{hist} \leftarrow \mathcal{L}_{hist} \cup L_{t-n+1}$
  **end if**
  **if** $\mathcal{L}_{hist}.size > m$ **then**
    $\mathcal{L}_{hist}.pop()$
  **end if**
  **if** $\mathcal{L}_{hist}.size \geq m^*$ and $\mathcal{L}_{new}.size = n$ **then**
    **if** KSTest($\mathcal{L}^h_{hist}$,$\mathcal{L}^h_{new}$) is True **then** ▷ Equation 1
      $S \leftarrow$ StateTransition($S$, $\mathcal{L}_{new}$, $\mathcal{S}$, $\delta$) ▷ Section 4.3
      Report concept drift, clear $\mathcal{L}_{hist}$ and $\mathcal{L}_{new}$
    **end if**
  **end if**
**end while**

data. In the test phase, abnormal snippets will cause larger reconstruction errors than normal data such that they are separable. The encoder and decoder can be implemented with a variety of deep models [25,26]. Considering the temporal dependencies in streaming data, RNNs and their variants [14,16] are naturally suitable for the target. In the following illustration, as an example, we take the LSTM-Autoencoder [14], which takes data snippets as input and produces a single latent representation for each snippet. To map the multivariate reconstruction error to the likelihood of anomalies, a commonly used approach is to estimate a multivariate Gaussian distribution from the reconstruction error of normal data and measure the Mahalanobis distance between the reconstruction error of an unknown data point to the estimated distribution [14]. Moreover, the Gaussian Mixture Model (GMM) [26] and energy-based model [25] can also be used for likelihood estimation. The thresholding over the estimated anomaly likelihood in an unsupervised manner is challenging, especially in the real-time prediction scenario. A possible non-parametric dynamic thresholding technique is proposed in [12]. The unsupervised approach for the adaptive threshold in different periods is not the main focus of this paper and will be addressed in our future work. In the following sections, we focus on adapting Autoencoders based on the state transitions.

### 4.2 Drift Detection in the Latent Space

In real-time, the latent representations of the Autoencoder are accumulated for concept drift detection. Existing concept drift detection approaches mostly work in the original space, targeting linear separable concept drifts. Considering the

complex concept drifts in multivariate streaming data, even non-linear distributional changes can be observed in the Autoencoder latent space. We perform the non-parametric and distribution-free two-sample Kolmogorov-Smirnov Test (KS-Test) [8,9] on each latent space dimension to check whether two latent representations are drawn from the same continuous distribution. Algorithm 1 shows the online concept drift detection process.

Formally, let $\mathcal{L}_{hist} = \{L_{t-\hat{m}-n+1}, L_{t-\hat{m}-n+2}, ..., L_{t-n}\}$ $(m^* \leq \hat{m} \leq m)$ be the accumulated latent representation since the last concept drift and $\mathcal{L}_{new} = \{L_{t-n+1}, L_{t-n+2}, ..., L_t\}$ be the latest latent representations. $m$ and $n$ are the maximum size of $\mathcal{L}_{hist}$ and $\mathcal{L}_{new}$, $m^*$ is the minimum size of $\mathcal{L}_{hist}$ to trigger a statistical test. $F_{hist}$ and $F_{new}$ are the empirical estimated cumulative distribution functions from the two latent representation sets. The null hypothesis (i.e., the observations in $\mathcal{L}_{hist}$ and $\mathcal{L}_{new}$ are from the same distribution) will be rejected if

$$\sup_{L} |F_{hist}(L) - F_{new}(L)| > c(\alpha)\sqrt{\frac{\hat{m}+n}{\hat{m}\cdot n}} \tag{1}$$

where $sup$ is the supremum function, $\alpha$ is the significance level, $c(\alpha) = \sqrt{-\ln(\frac{\alpha}{2})\cdot\frac{1}{2}}$. We maintain both $\mathcal{L}_{hist}$ and $\mathcal{L}_{new}$ as queues. $m$ is larger than $n$ such that $\mathcal{L}_{hist}$ contains longer and more stable historical information, while $\mathcal{L}_{new}$ captures the latest data characteristic. The drift detector will only start if $\mathcal{L}_{hist}$ contains at least $m^*$ samples, such that the procedure starts smoothly.

Since the KS-test is designed for univariate data, we conduct parallel tests in each latent dimension and report concept drift if the null hypothesis is rejected on all the dimensions. Once a concept drift is detected, we will conduct the state transition procedure for model adaptation (Sect. 4.3). The historical and latest sample sets are emptied, and we further collect samples from the new data distribution.

### 4.3 State Transition Model

Modeling reoccurring data distributions (e.g., seasonal changes), coupling Autoencoders with drift detection, and reusing models based on the distributional features can increase the efficiency of updating a deep model in real time. We represent every stable data distribution (concept) and the corresponding Autoencoder as a *state* $S \in \mathcal{S}$. In STAD, for each period between two concept drifts in the data stream, the data distribution, as well as the corresponding Autoencoder, are represented in a queue $\mathcal{S}$ with limited size. The first state $S_0 \in \mathcal{S}$ represents the beginning period of the data stream before the first concept drift. After a concept drift, a new Autoencoder will be trained from scratch with the latest $m$ input data snippets, if no existing element in $\mathcal{S}$ fits the current data distribution; Otherwise, the state will transit to the existing one and reuse the corresponding Autoencoder. In our study, we assume that sufficient data after the concept drifts can be accumulated to initialize a new Autoencoder.

To compare the distributional similarity between the newly arrived latent representations $Q$ and the distributions of existing states $\{P_i | i = 1, ..., N\}$, we

employ the symmetrized Kullback-Leibler Divergence. The similarity between $Q$ and an existing state distribution $P_i$ is defined as

$$D_{KL}(P_i, Q) = \sum_{L \in \mathcal{L}} P_i(L) log \frac{P_i(L)}{Q(L)} + Q(L) log \frac{Q(L)}{P_i(L)} \tag{2}$$

The next step is to estimate the corresponding probability distributions from the sequence of latent representations. In [8,9], the probability distribution of categorical data is estimated by the number of object appearances in each category. In our case, the target is to estimate the probability distribution of fixed-length real-valued latent representations. In previous research, one possibility for density estimation of streaming data is to maintain histograms of the raw data stream [21]. In STAD, we take advantage of the fix-sized latent representation of Autoencoders and maintain histograms of each period in the latent space for the density estimation.

Let $\mathcal{L} = \{L_1, L_2, ..., L_t\}$ be a sequence of observed latent representations, where $L_i = \langle h_1^i, h_2^i, ..., h_H^i \rangle$ and $H$ is the latent space size, the histogram of $\mathcal{L}$ is

$$g(k) = \frac{1}{t} \sum_{L_i \in \mathcal{L}} \frac{e^{h_k^i}}{\sum_{j=1}^{H} e^{h_j^i}} \quad (k = 1...H) \tag{3}$$

and the density of a given period is estimated by $P(k) = g(k)$. Hence, Eq. 2 can be converted to

$$D_{KL}(P_i, Q) = \sum_{k=1...H} P_i(k) log \frac{P_i(k)}{Q(k)} + Q(k) log \frac{Q(k)}{P_i(k)} \tag{4}$$

For a newly detected concept with distribution $Q$, if there exist a state $S_i (i \in [1, N])$ with corresponding probability distribution $P_i$ satisfies $D_{KL}(P_i, Q) \leq \epsilon$, where $\epsilon$ is a tolerant factor, and $S_i$ is not the direct last state, the concept drift can be treated as a reoccurrence of the existing concept. Therefore the corresponding Autoencoder can be reused, and the state transfers to the existing state. If no Autoencoder is reusable, a new one will be trained on the latest arrived data after concept drift. To prevent an explosion in the number of states, the state transition model $\mathcal{M} = \langle \mathcal{X}, \mathcal{S}, \delta \rangle$ only maintains the $N$ latest states. Considering that no information about the upcoming new concept is accessible, despite a potentially high error rate, we still keep using the previous model for anomaly detection until the model adaptation is finished. Or in other words, the previous model is used for prediction during the upcoming *drift period*. The state transition procedure is described in Algorithm 2.

## 5 Experiment

Common time series anomaly detection benchmark datasets are often stationary without concept drift. Although some claim that their datasets contain distributional changes, the drift positions are not explicitly labeled and are hard for us

**Algorithm 2.** State Transition Procedure

**function** StateTransition($S_{hist}$, $\mathcal{L}_{new}$, $\mathcal{S}$, $\delta$)
  $P_{new}$ = DensityEstimation($\mathcal{L}_{new}$)
  **if** $\min_{S_i=\langle P_i, AE_i\rangle \in \mathcal{S}} \{D_{KL}(P_{new}, P_i)\} \leq \epsilon$ **then** ▷ Equation 4
    $\delta \leftarrow \delta \cup (S_{hist} \Rightarrow S_{min})$
    **return** $S_{min}$
  **end if**
  $S_{new} \leftarrow \langle P_{new}, AE_{new}\rangle$ ▷ $AE_{new}$: Trained on new concept data
  $\mathcal{S} \leftarrow \mathcal{S} \cup S_{new}$
  $\delta \leftarrow \delta \cup (S_{hist} \Rightarrow S_{new})$
  **if** $\mathcal{S}.size > N$ **then**
    Remove the oldest state and relevant transitions
  **end if**
  **return** $S_{new}$
**end function**

to evaluate. To this end, we introduce multiple synthetic datasets with known positions of abnormal events and concept drifts. Furthermore, we concatenate selected real-world datasets to simulate concept drifts. We evaluate the anomaly detection performance and show the effectiveness of model adaptation based on the detected drifts.

### 5.1 Experiment Setup

**Datasets.** We first generate multiple synthetic datasets from a sine and a cosine wave with anomalies and concept drifts. For initialization, we generate 5000 in purely normal data points with amplitude 1, period 25 for the two wave dimensions. For real-time testing, we generate 60000 samples containing 300 point anomalies. All synthetic datasets contain reoccurring concepts, such that we can evaluate the state-transition and model reusing of STAD. Following [18], we create the drifts in three fashions, abrupt ($A$-$*$), gradual ($G$-$*$) and incremental ($I$-$*$). For each type of drift, we create a standard version ($*$-*easy*) and a hard version ($*$-*hard*) with more frequent drifts leaving the model less time for reaction. The drifts are created by either swapping the feature dimensions (-*Swap*-) or multiplying a factor by the amplitude (-*Ampl*-). The abrupt drifts are created by directly concatenating two concepts. The gradual drifts take place in a 2000 timestamp period with partial instances changing to the new concepts. The incremental drifts also take 2000 timestamps, while the drift features incrementally change at every timestamp. Anomaly points are introduced by swapping the values on the two dimensions.

*SMD* (Server Machine Dataset) [23] is a real-world multivariate dataset containing anomalies. To simulate concept drifts, we manually compose *SMD-small* and *SMD-large*. Both only contain abrupt drifts. *SMD-small* consists of test data from *machine-1-1* to *machine-1-3*, which are concatenated in the order of *machine-1-1*⇒*machine-1-2*⇒*machine-1-1*⇒*machine-1-3*. We take each

machine as a concept and *machine-1-1* appears twice. *SMD-large* consists of data from *machine-1-1* to *machine-1-8* and is composed in the same fashion with *machine-1-1* recurring after each concept. For both datasets, the training set of *machine-1-1* is used for the model initialization.

*Forest* (Forest CoverType) [5] is another widely used multivariate dataset in drift detection. To examine the performance in a real-world scenario, we do not introduce any artificial drift here, but only consider the forest cover type changes as implicit drifts. As in [10], we consider the smallest class *Cottonwood/Willow* as abnormal.

**Evaluation Metrics.** We adopt the AUROC (AUC) score to evaluate the anomaly detection performance. An anomaly score $a \in [0, 1]$ is predicted for each timestamp. The larger $a$, the more likely it is to be abnormal. The labels are either 0 (normal) or 1 (anomaly). We evaluate the AUC score over anomaly scores without applying any threshold [6] so that the performance is not impacted by the quality of the selected threshold technique.

**Competitors.** We compare our model with two commonly used unsupervised streaming anomaly detectors. The LSTM-AD [15] is a prediction-based approach. Using the near history to predict the near future, the model is less impacted by concept drifts. The prediction deviation to real values of the data stream indicates the likelihood of being abnormal. The HTM [1] model is able to detect anomalies from streaming data with concept drifts. Neither LSTM-AD nor HTM provides an interpretation of the evolving data stream besides anomaly detection.

**Experimental Details.** We construct the Autoencoders with two single-layer LSTM units. All training processes are configured with a 0.2 dropout rate, $1e-5$ weight decay, $1e-4$ learning rate, and a batch size of 8. All Autoencoders are trained for 20 epochs with early stopping. We detect drifts with the KS-Tests at a significance level of $\alpha = 0.05$. We restrict that $\mathcal{L}_{hist}$ has to contain at least $m^* = 50$ data point to trigger the KS-Tests. We set the input snippet size as the sine curve period 25. For the *SMD*-based datasets, following [23], the snippet size is set to 100. We process the snippets of the data stream as a sliding window without overlap. All experiments are conducted on an NVIDIA Quadro RTX 6000 24GB GPU and are averaged over three runs.

### 5.2 Performance

**Overall Anomaly Detection Performance Comparison.** We compare the AUC score in the streaming data anomaly detection task between STAD and the competitors. In STAD, we set the latent representation size $H = 50$, and the sizes of the two buffers during the online prediction phase as $m = 200$ and $n = 50$. The threshold $\epsilon$ is set to 0.0005. We evaluate the performance of STAD in each *state*

and report the average AUC. The results are shown in Table 1. STAD achieves the best performance on all synthetic datasets with abrupt and gradual drifts. In the two more complicated real-world datasets, STAD outperforms LSTM-AD and stays comparable to HTM, while requiring significantly less processing time (see Sect. 5.2). LSTM-AD shows a dominating performance on the two incremental datasets. Due to the fact that the value at every single timestamp changes in *I-Ampl-easy* and *I-Ampl-hard*, LSTM-AD benefits from its dynamic forecasting at every timestamp, while STAD suffers under the delay between state transitions.

**Table 1.** Anomaly detection performance (AUC).

| | STAD (Ours) | LSTM-AD | HTM |
|---|---|---|---|
| *A-Swap-easy* | $0.986 \pm 0.005$ | $\mathbf{0.994} \pm 0.005$ | $0.535 \pm 0.008$ |
| *A-Swap-hard* | $\mathbf{0.883} \pm 0.016$ | $0.742 \pm 0.076$ | $0.440 \pm 0.017$ |
| *A-Ampl-easy* | $\mathbf{0.816} \pm 0.025$ | $0.717 \pm 0.052$ | $0.500 \pm 0.006$ |
| *A-Ampl-hard* | $\mathbf{0.810} \pm 0.012$ | $0.715 \pm 0.051$ | $0.499 \pm 0.006$ |
| *G-Swap-easy* | $\mathbf{0.948} \pm 0.019$ | $0.854 \pm 0.064$ | $0.506 \pm 0.008$ |
| *G-Swap-hard* | $\mathbf{0.926} \pm 0.030$ | $0.800 \pm 0.082$ | $0.502 \pm 0.005$ |
| *I-Ampl-easy* | $0.911 \pm 0.014$ | $\mathbf{0.975} \pm 0.018$ | $0.488 \pm 0.003$ |
| *I-Ampl-hard* | $0.970 \pm 0.017$ | $\mathbf{1.000} \pm 0.000$ | $0.470 \pm 0.003$ |
| *SMD-small* | $0.755 \pm 0.067$ | $0.562 \pm 0.001$ | $\mathbf{0.813} \pm 0.001$ |
| *SMD-large* | $\mathbf{0.763} \pm 0.016$ | $0.578 \pm 0.002$ | $0.762 \pm 0.003$ |
| *Forest* | $0.751 \pm 0.022$ | $\mathbf{0.977} \pm 0.001$ | $0.211 \pm 0.001$ |

**Parameter Sensitivity.** In this section, we conduct multiple experiments to examine the impact of several parameters to STAD. We maintain two data buffers $\mathcal{L}_{hist}$ and $\mathcal{L}_{new}$ to collect data from the Autoencoder latent space to detect drifts. We set the upper bound of $\mathcal{L}_{hist}$'s size $m = 200$ for all experiments. Depending on the computational resource, larger $m$ will lead to more stable test results. Here we examine the effect of the lower bound $m^*$. Similarly, we also experiment with different sizes $n$ of $\mathcal{L}_{new}$. Additionally, the latent representation size $H$ of Autoencoders is a parameter depending on the complexity of the input data.

In Fig. 2, we check the impact of the three parameters $H$, $n$ and $m^*$ on abrupt drifting datasets. We try different values on each parameter while keeping the other two parameters equal to 50. The model is not sensitive to either of the three parameters on abrupt drifting datasets. Specifically for the two buffers, 20 data windows of both the historical ($m^*$) and the latest ($n$) latent representations are sufficient for drift detection. Similar results have been shown on the datasets with gradual and incremental drifts. The performance is stably better than the

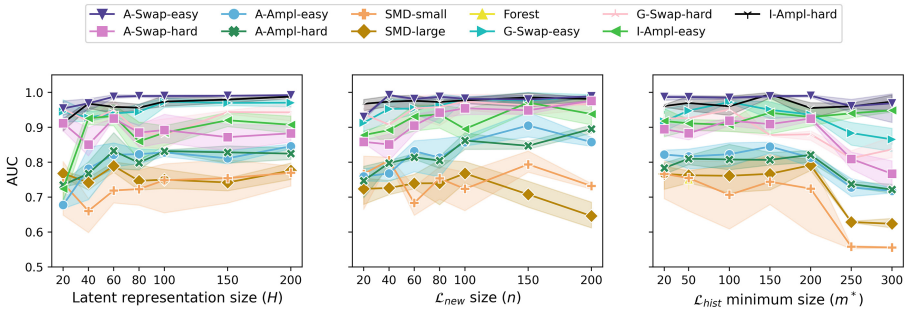


**Fig. 2.** Parameter sensitivity: AUC scores under different settings of latent representation size $H$, $\mathcal{L}_{new}$ size $n$ and minimum size $m^*$ of $\mathcal{L}_{hist}$ to trigger KS-Tests.

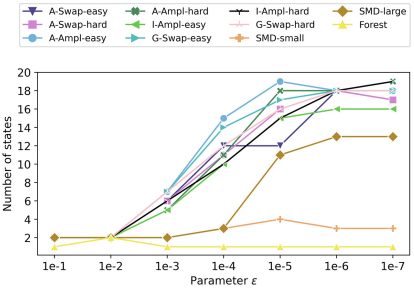


**Fig. 3.** Number of distinct states under different settings of threshold $\epsilon$.

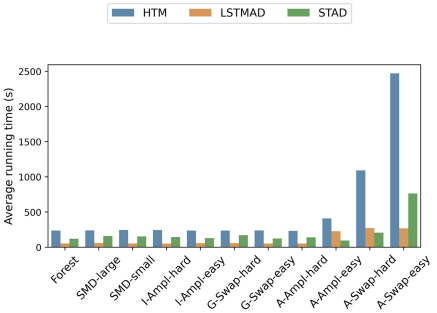


**Fig. 4.** Average running time comparison.

abrupt drifting dataset. One reason is that a longer drifting period leaves the model more time for detecting the drifts and conducting the state transition. On the contrary, the model may make mistakes after an abrupt drift until sufficient data is collected and the state transition is triggered.

The other parameter $\epsilon$ controls the sensitivity of re-identifying an existing state. The larger $\epsilon$, the more likely for the model to transfer to a similar existing state. We set all $H$, $m$, and $n$ to 50 and examine $\epsilon$ with a value that varies from 0.1 to $1e-7$, and observe the total number of distinct states created during the online prediction. As shown in Fig. 3, with large $\epsilon$'s (0.1 or 0.01), the model only creates two states and transits only between them once a drift is detected. On the contrary, too small $\epsilon$ will lead to an explosion of state. The model seldom matches an existing state but creates a new state and trains a new model after each detected drift. Currently, we determine a proper value of $\epsilon$ heuristically during the online prediction.

**Running Time Analysis.** Finally, we compare the running time (including training, prediction, and updating time) of the three models on all datasets in

Fig. 4. It turns out that the efficient reusing of existing models especially benefits large and complex datasets, where the model adaptation is time-consuming. STAD costs a similar processing time as LSTMAD in synthetic datasets and less in real-world datasets. The HTM always takes significantly more processing time.

## 6 Conclusion

We proposed the state-transition-aware streaming data anomaly detection approach STAD. With a reconstruction-based Autoencoder model, STAD detects abnormal patterns from data streams in an unsupervised manner. Based on the latent representation, STAD maintains states for concepts and detects drifts with a state transition model. With this, STAD can identify recurring concepts and reuse existing Autoencoders efficiently; or train a new Autoencoder when no existing model fits the new data distribution. Our empirical results have shown that STAD achieves comparable performance as the state-of-the-art streaming data anomaly detectors. Beyond that, the states and transitions also shed light on the complex and evolving data stream for more interpretability.

There are still some challenges in the current model. The current selection of parameter $\epsilon$ is still heuristic-based. We assume sufficient data is available to train a new Autoencoder if a drift has been detected. And we did not investigate the variety of drift types, especially gradual drifts with different lengths of *drift periods*. We plan to address the challenges above in future work.

**Acknowledgement.** This work was supported by the Research Center Trustworthy Data Science and Security, an institution of the University Alliance Ruhr.

## References

1. Ahmad, S., Lavin, A., Purdy, S., Agha, Z.: Unsupervised real-time anomaly detection for streaming data. Neurocomputing **262**, 134–147 (2017)
2. Ahmadi, Z., Kramer, S.: Modeling recurring concepts in data streams: a graph-based framework. Knowl. Inf. Syst. **55**(1), 15–44 (2018)
3. Bianco, A.M., Garcia Ben, M., Martinez, E., Yohai, V.J.: Outlier detection in regression models with Arima errors using robust estimates. J. Forecast. **20**(8), 565–579 (2001)
4. Bifet, A., Gavalda, R.: Learning from time-changing data with adaptive windowing. In: Proceedings of the 2007 SIAM International Conference on Data Mining, pp. 443–448. SIAM (2007)
5. Blackard, J.A., Dean, D.J.: Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. Comput. Electron. Agric. **24**(3), 131–151 (1999)
6. Campos, G.O., et al.: On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Min. Knowl. Disc. **30**(4), 891–927 (2016)

7. Ceci, M., Corizzo, R., Japkowicz, N., Mignone, P., Pio, G.: Echad: embedding-based change detection from multivariate time series in smart grids. IEEE Access **8**, 156053–156066 (2020)
8. Chen, C., Wang, Y., Zhang, J., Xiang, Y., Zhou, W., Min, G.: Statistical features-based real-time detection of drifted twitter spam. IEEE Trans. Inf. Forensics Secur. **12**(4), 914–925 (2016)
9. Dasu, T., Krishnan, S., Venkatasubramanian, S., Yi, K.: An information-theoretic approach to detecting changes in multi-dimensional data streams. In: In Proceedings of Symposium on the Interface of Statistics, Computing Science, and Applications. Citeseer (2006)
10. Dong, Y., Japkowicz, N.: Threaded ensembles of autoencoders for stream learning. Comput. Intell. **34**(1), 261–281 (2018)
11. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Bazzan, A.L.C., Labidi, S. (eds.) SBIA 2004. LNCS (LNAI), vol. 3171, pp. 286–295. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28645-5_29
12. Hundman, K., Constantinou, V., Laporte, C., Colwell, I., Soderstrom, T.: Detecting spacecraft anomalies using LSTMS and nonparametric dynamic thresholding. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 387–395 (2018)
13. Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., Zhang, G.: Learning under concept drift: a review. IEEE Trans. Knowl. Data Eng. **31**(12), 2346–2363 (2018)
14. Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., Shroff, G.: Lstm-based encoder-decoder for multi-sensor anomaly detection. arXiv preprint arXiv:1607.00148 (2016)
15. Malhotra, P., Vig, L., Shroff, G., Agarwal, P., et al.: Long short term memory networks for anomaly detection in time series. In: Proceedings, vol. 89, pp. 89–94 (2015)
16. Marchi, E., Vesperini, F., Weninger, F., Eyben, F., Squartini, S., Schuller, B.: Non-linear prediction with lstm recurrent neural networks for acoustic novelty detection. In: 2015 International Joint Conference on Neural Networks (IJCNN), pp. 1–7. IEEE (2015)
17. Meng, H., Zhang, Y., Li, Y., Zhao, H.: Spacecraft anomaly detection via transformer reconstruction error. In: Jing, Z. (ed.) ICASSE 2019. LNEE, vol. 622, pp. 351–362. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-1773-0_28
18. Pesaranghader, A., Viktor, H.L., Paquet, E.: Mcdiarmid drift detection methods for evolving data streams. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–9. IEEE (2018)
19. Rabanser, S., Günnemann, S., Lipton, Z.: Failing loudly: An empirical study of methods for detecting dataset shift. Adv. Neural Inf. Process. Syst. **32** (2019)
20. dos Reis, D.M., Flach, P., Matwin, S., Batista, G.: Fast unsupervised online drift detection using incremental kolmogorov-smirnov test. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1545–1554 (2016)
21. Sebastião, R., Gama, J.: Change detection in learning histograms from data streams. In: Neves, J., Santos, M.F., Machado, J.M. (eds.) EPIA 2007. LNCS (LNAI), vol. 4874, pp. 112–123. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-77002-2_10
22. Sipple, J.: Interpretable, multidimensional, multimodal anomaly detection with negative sampling for detection of device failure. In: International Conference on Machine Learning, pp. 9016–9025. PMLR (2020)

23. Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D.: Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2828–2837 (2019)
24. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: International Conference on Machine Learning, pp. 3319–3328. PMLR (2017)
25. Zhai, S., Cheng, Y., Lu, W., Zhang, Z.: Deep structured energy based models for anomaly detection. In: International Conference on Machine Learning, pp. 1100–1109. PMLR (2016)
26. Zong, B., et al.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: International Conference on Learning Representations (2018)