Engineering Applications of Artificial Intelligence 95 (2020) 103919

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Policy-based reinforcement learning for time series anomaly detection

Mengran Yu, Shiliang Sun *

School of Computer Science and Technology, East China Normal University, 3663 North Zhongshan Road, Shanghai 200062, PR China

ARTICLE INFO

Keywords:
Time series anomaly detection
Reinforcement learning
Policy-based methods

ABSTRACT

Time series anomaly detection has become a crucial and challenging task driven by the rapid increase of streaming data with the arrival of the Internet of Things. Existing methods are either domain-specific or require strong assumptions that cannot be met in realistic datasets. Reinforcement learning (RL), as an incremental self-learning approach, could avoid the two issues well. However, the current investigation is far from comprehensive. In this paper, we propose a generic policy-based RL framework to address the time series anomaly detection problem. The policy-based time series anomaly detector (PTAD) is progressively learned from the interactions with time-series data in the absence of constraints. Experimental results show that it outperforms the value-based temporal anomaly detector and other state-of-the-art detection methods whether training and test datasets come from the same source or not. Furthermore, the tradeoff between precision and recall is well respected by the PTAD, which is beneficial to fulfill various industrial requirements.

1. Introduction

Anomaly detection has been a hot spot since critical systems need to detect anomalies as early as possible to avoid big troubles (Chandola et al., 2009). An anomaly is considered as a point or a change that differs from other data. With the arrival of the Internet of Things, innumerable sensors are installed to collect amounts of data which change over time and are called time-series data. Since there exist complicated abnormal patterns which may be spatial or temporal depending on whether they are contextual or not in time-series data and this type of data often occur with a periodic or seasonal mode, it is a challenging issue to detect anomalies precisely in these data.

Time series anomaly detection has been investigated with numerous applications. Most of them rely on sensors, which leads to the use of sensors in our daily life increasing. However, these sensors are susceptible to faults. A faulty sensor may cause process performance degradation, process shutdowns, or fatal accidents. Sensor fault detection aims to detect faults as early as possible. Nowadays, sensor fault detection plays a more and more important role in relevant applications such as process control (Borghesi et al., 2019), signal recognition (Hu et al., 2017) and monitoring (Dong, 2019; Fernández-Sanjurjo et al., 2019).

Currently, not only the supervised learning methods (Chauhan and Vig, 2015; Tuor et al., 2018) but also the semi-supervised learning and unsupervised learning approaches (Ahmad et al., 2017; Gorokhov et al., 2017; Ghasedi Dizaji et al., 2018) for time series anomaly detection have been proposed. Existing approaches are generally designed for specific applications and particular datasets (Zhu and Laptev, 2017; Oh and Yun, 2018; Wei et al., 2018). A specialized anomaly detector is often applied to only one use-case. It is troublesome to construct an anomaly detector for multiple use-cases. There are also a few generic methods which achieve great performance on time-series benchmark datasets. Unfortunately, these approaches require especially analyzing the characteristic of the full data or making strong assumptions, e.g., the training data is anomaly-free which is yet unrealistic in most real datasets (Laptev et al., 2015; Venkataraman et al., 2006).

Reinforcement learning (RL) (Sutton and Barto, 2018) conforms to the learning process of human beings. The agent receives positive feedback when useful information is learned and acquires a negative signal when learning useless or harmful information to instruct the following behaviors. It follows the incremental self-learning process that the agent autonomously learns a generic framework from interactions with the environment without any assumption and constraint. Hence, it provides a novel way of solving the anomaly detection problem. Due to the consecutive characteristic of the time-series data, it is natural to adapt the time series anomaly detection process into the framework of RL.

However, explorations of this field are extremely insufficient in the current literature. Huang et al. (2018) attempted a value-based deep reinforcement learning (DRL) time series anomaly detector which adopted the Deep Q-Function Network (DQN) algorithm (Mnih et al., 2015), however, just with a brief instruction. Their experimental results demonstrated the possibilities of detecting abnormal behaviors with RL methods. As the other type of methods, policy-based DRL algorithms, also show excellent performance in sequential prediction and robot

* Corresponding author.
E-mail addresses: mengranyu97@gmail.com (M. Yu), shiliangsun@gmail.com (S. Sun).

https://doi.org/10.1016/j.engappai.2020.103919
Received 8 December 2019; Received in revised form 2 July 2020; Accepted 25 August 2020
Available online 28 August 2020

control since they directly operate in the policy space (Bahdanau et al., 2016; Pane et al., 2016; Wang et al., 2015). Hence, we wonder how well the policy-based DRL detection methods perform when compared with the value-based DRL detection methods and whether RL is an effective solution to the time series anomaly detection problem.

This paper adapts the policy-based RL framework into the time series anomaly detection problem and proposes a general anomaly detector. Experimental results demonstrate the effectiveness of the proposed detector on homologous and heterologous datasets when compared with the value-based RL time series anomaly detector and other advanced detection methods. The contributions of this paper are listed as follows. Firstly, we propose a novel policy-based DRL time series anomaly detector (PTAD) based on the asynchronous advantage actor–critic (A3C) algorithm (Mnih et al., 2016), which is one of the most advanced DRL algorithms to address the anomaly detection problem. Secondly, compared with the value-based DRL detector and other state-of-the-art anomaly detection techniques, the proposed PTAD performs superiorly whether on the same or the different source and target datasets. Furthermore, the optimal stochastic detection policy acquired from the PTAD allows adjusting the criterion of distinguishing normal and abnormal behaviors, which controls the tradeoff between precision and recall to meet some particular demands in various applications, i.e., avoiding incorrect anomaly detections which result in unnecessary interruptions.

The remainder of this paper is organized as follows. Section 2 describes various anomaly detection strategies and algorithms. The basic concepts of RL and the formalization of RL based time series anomaly detection problem are illustrated in Section 3. Section 4 introduces the proposed PTAD in detail and compares the traits between the value-based and policy-based DRL detection approaches. Descriptions of datasets, experimental settings and results are shown in Section 5. Section 6 concludes the work of this paper and sketches directions for possible future work.

2. Related work

There are numerous algorithms and strategies about anomaly detection over the recent years, which could be roughly classified into two categorizations: statistical based methods and machine learning based approaches.

Statistical based methods construct a statistical model from the given data and operate a statistical inference test to justify whether new data fit the model. Non-parametric models are generally histogram based (Yamanishi et al., 2004) and kernel function based (Kumar et al., 2016), which learn the underlying distribution of normal behaviors from the given data directly. Gaussian model (Shekhar et al., 2001), regression model (Bianco et al., 2001), mixture model of parametric distributions (Agarwal, 2005, 2007) are classical parametric statistical models, which instead assume that the underlying distribution of normal data matches the presupposed distribution. However, these methods depend on the assumption that normal behaviors suit the predefined distribution, which is not often true in realistic datasets.

Machine learning based approaches learn a model from the labeled training data and distinguish new data between normal class and abnormal class with the model. There are two ways to learn the model where one is classification and the other is clustering. Bayesian networks (Das and Schneider, 2007), support vector machines (Ma and Perkins, 2003), rule based (Tandon and Chan, 2007) and neural networks (Mukkamala et al., 2002) are common machine learning algorithms to build the anomaly detection classifiers. Clustering anomaly detections are mainly based on k-nearest-neighbors algorithm (Ramaswamy et al., 2000), which is expanded by local outlier factor (LOF) (Breunig et al., 2000) and connectivity based outlier factor (COF) algorithm (Tang et al., 2002). Because new anomalous behaviors might break out in nature and relate to the contextual information, machine learning based approaches are suitable for domain-specific applications where informative training data are available.

Due to the complexity of time-series data, e.g., the pattern of data is continuously changing and temporal dependencies are contained in them, some specific techniques and algorithms are explored. Skyline (Esty, 2014) is a real-time anomaly detection system developed by Esty Inc in 2014. Twitter Inc. released its package to detect anomalies which is robust, from a statistical standpoint, in the presence of seasonality and an underlying trend (Twitter, 2015). ContextOSE (Mikhail, 2015) is based on contextual anomaly detection, which captures the local rather than global information. Numenta and Numenta TM (Ahmad et al., 2017) are expanded from the hierarchical temporal memory (HTM), which is a detailed computational theory of the neocortex and the core is storing and recalling spatial and temporal patterns. With the development of deep learning, RNN or LSTM based time series anomaly detectors were proposed (Malhotra et al., 2015) where they learn a predictive model from the normal training time stamps and mark normal or abnormal depending on the error between the predictive values and true values. There are also some variants based on the autoencoder (Malhotra et al., 2016; Amarbayasgalan et al., 2018).

Recently, RL is taken into consideration for solving the time series anomaly detection problem because of its generic framework and incremental self-learning property. Bourdonnaye et al. (2017) learned binocular fixations with informative reward requiring little supervised information where the reward computation was based on an anomaly detection mechanism which used convolutional autoencoders. They just regarded anomaly detection as an auxiliary technique for generating the feedback signals. Huang et al. (2018) attempted a value-based DRL time series anomaly detector with the DQN algorithm, which built a bridge between RL and anomaly detection. However, the acquired deterministic policy was unsatisfactory to dynamically modulate the threshold of justifying an anomaly for several requirements in different applications. Therefore, we propose the PTAD based on the A3C algorithm, which is dynamically adjustable for controlling the tradeoff between precision and recall in various circumstances.

3. Preliminaries

Time series anomaly detection can be modeled as a sequential decision process, which is formulated to the Markov decision process (MDP) in RL. Hence, the MDP bridges the gap between time series anomaly detection and RL. The significant feature of RL is that the agent interacts with the environment. That is, an action taken by the agent at the current time step t will affect the state at time step $t+1$. Then the following actions will be influenced by the reward received from the environment. In this section, we firstly introduce basic conceptions in RL and then illustrate the formalization of RL based time series anomaly detection problem in detail.

3.1. Background: Reinforcement learning

A reinforcement learning problem is usually represented as an MDP whose specific form is a tuple of five elements: $\langle S, \mathcal{A}, P, R, \gamma \rangle$. At time step t, assume that the agent is at the state $s_t \in S$ and selects the action $a_t \in \mathcal{A}$ according to the policy π, where π is a mapping from a state s_t to an action a_t. The agent will receive an immediate reward r_t where $r_t = R(s_t, a_t)$ and obtain the next state s_{t+1} according to the state transition probabilities function $P(s_{t+1}|S = s_t, A = a_t)$. These interactions with environment $\mathcal{E}$ come into being a trajectory τ until the agent reaches a terminal state. The goal of the agent is to maximize the expected return $\mathbb{E}[R_t]$ from each state s_t where the return is $R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$. γ represents the discount factor which ranges from 0 to 1 and measures the importance of current rewards on future rewards. The agent finally obtains an optimal policy π^* via learning from experiences.

There are two methods to train the policy π which are the valued-based RL methods and the policy-based RL approaches.

In the value-based RL methods, the agent optimizes the target policy indirectly by maximizing the corresponding value function where

the action-value function $Q(s,a)$ is usually chosen. We now consider the off-policy value-based learning that means the behavior policy is for sampling and the target policy is for optimizing, whose typical algorithm is called Q-learning. The target policy π is greedy, that is,

$$\pi(s_{t+1}) = \underset{a'}{\operatorname{argmax}}\, Q(s_{t+1}, a'). \tag{1}$$

The behavior policy μ is ϵ-greedy where the agent chooses the greedy action with probability $1-\epsilon$ and randomly chooses an action with probability ϵ, that is,

$$\mu(s_{t+1}) = \begin{cases} \frac{\epsilon}{|\mathcal{A}|} + 1 - \epsilon & \text{if } a^* = \operatorname{argmax}_{a\in\mathcal{A}} Q(s,a) \\ \frac{\epsilon}{|\mathcal{A}|} & \text{otherwise,} \end{cases} \tag{2}$$

where $|\mathcal{A}|$ is the cardinality of the action space. The iterative formula in training process is given as

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right). \tag{3}$$

In the policy-based RL methods, the agent directly parameterizes and optimizes the target policy $\pi(a|s;\theta)$ with the parameters θ. Compared with the value-based algorithms, the policy-based approaches are effective for high-dimensional or continuous action spaces and can learn stochastic policies which are more practical than deterministic policies. The most famous framework is the actor–critic where an actor network $\pi(a|s;\theta)$ with parameters θ estimates the target policy π to take an action under a specific state and a critic network $V(s;\upsilon)$ with parameters υ approximates the state-value function $V(s)$ to evaluate the current policy. The critic uses the least-squares policy evaluation where the minimum loss function is written as

$$L(\upsilon) = \mathbb{E}\left[\left(r + \gamma V^{\pi_\theta}(s';\upsilon) - V^{\pi_\theta}(s;\upsilon)\right)^2\right]. \tag{4}$$

The actor updates policy parameters θ with the policy gradient theorem which instructs to iterate in directions of suggestions of the critic. Furthermore, it is beneficial to consider the entropy of the policy for improving exploration. The final maximum loss function including the entropy regularization term is shown as below

$$L(\theta) = \mathbb{E}\left[\log\pi(a|s;\theta)\delta_\upsilon + \beta H(\pi(a|s;\theta))\right], \tag{5}$$

where $\delta_\upsilon = r + \gamma V^{\pi_\theta}(s';\upsilon) - V^{\pi_\theta}(s;\upsilon)$ is the advantage function, $H(\pi(a|s;\theta)) = -\pi(a|s;\theta)\log\pi(a|s;\theta)$ is the entropy of the policy $\pi(a|s;\theta)$.

3.2. Formalization of RL based time series anomaly detection problem

Time series anomaly detection could be considered as an MDP because the decision of normal or abnormal at the current time step will change the environment by whether it triggers an anomaly detection or not. And the next decision will be influenced by the changing environment. Hence, it is natural to adapt the temporal anomaly detection into the framework of RL (Huang et al., 2018). Next, we instruct the concrete time series anomaly detection formulation.

State Since the next action taken by the agent is affected by the changing environment which is comprised of the previous decisions and the current time series, the state includes two parts where one is the sequence of the previous actions, i.e., $s_{action} = \langle a_{t-m}, a_{t-m+1}, \ldots, a_{t-1}\rangle$ and the other is the current time series, i.e., $s_{time} = \langle x_{t-m+1}, x_{t-m+2}, \ldots, x_t\rangle$. We want to know the action a_t with the previous m actions and m time stamps. The state space $\mathcal{S}$ is regarded as infinite because the real time series have a variety of alterations.

Action It is simple to define the action space $\mathcal{A} = \{0,1\}$ where 0 represents the normal behavior and 1 means an anomaly is detected.

Reward Designing a proper reward function is important for the agent to learn an effective policy. There are several types of temporal data, such as the labeled, the semi-labeled and the unlabeled, which correspond to supervised learning, semi-supervised learning and unsupervised learning separately in conventional machine learning. The RL agent needs relatively correct instructions for learning an effective detection policy. Therefore, we consider the labeled training data and construct the reward function with labels.

Table 1
Confusion matrix.

Prediction/True value	Positive	Negative
Positive	TP	FP
Negative	FN	TN

We utilize the confusion matrix of the prediction problem in traditional machine learning to design the reward function $R(s,a)$. The confusion matrix is shown as Table 1 where the positive means detecting an anomaly and the negative represents normal behaviors. The reward function is designed below where A, B, C, and D are the positive number and can be set different values according to the characteristic of the training time series data and realistic demands. For example, if a wrong anomaly detection is forbidden in some real applications, we can fix the B slightly larger to give apparent negative feedback.

$$R(s,a) = \begin{cases} A & \text{if the action is a TP} \\ -B & \text{if the action is a FP} \\ -C & \text{if the action is a FN} \\ D & \text{if the action is a TN} \end{cases} \tag{6}$$

Policy The target policy is represented by the anomaly detector which has two forms. One is for the deterministic policy acquired from the value-based detector. It gives the explicit action under the current state and is written as

$$\pi(s) = a, \quad s \in \mathcal{S}, a \in \mathcal{A}. \tag{7}$$

The other is for the stochastic policy obtained from the policy-based detector. It provides the probability of each action under the present state that means the criterion of determining an action is adjustable. It is formulated as

$$\pi(s,a) = p(a|s), \quad s \in \mathcal{S}, a \in \mathcal{A}. \tag{8}$$

Value function The state-value function of a detection policy is the same as the standard form which we specialize into a particular formula as follow

$$V_\pi = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} | s_t = s\right]. \tag{9}$$

And the action-value function is defined as

$$Q_\pi = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} | s_t = s, a_t = a\right]. \tag{10}$$

Optimal policy The optimal policy is our ideal anomaly detector maximizing the expected reward whose forms are following:

$$\pi^* = \operatorname{agrmax}_\pi V_\pi \quad or \quad \pi^* = \operatorname{agrmax}_\pi Q_\pi. \tag{11}$$

4. Policy-based time series anomaly detector (PTAD)

In this section, we present a policy-based DRL time series anomaly detector, called PTAD, for discovering abnormal behaviors in time-series data. Furthermore, we compare the traits between the policy-based and value-based DRL time series anomaly detectors.

For the RL based time series anomaly detection setting, the environment is a time series repository which contains a large population of labeled time-series data. With these data, the environment is able to generate specific states for training the agent and determine the goodness of the actions taken by the agent. Another essential component of the setting is the agent that simulates how the time series anomaly detector operates and optimizes. It takes the current n time stamps

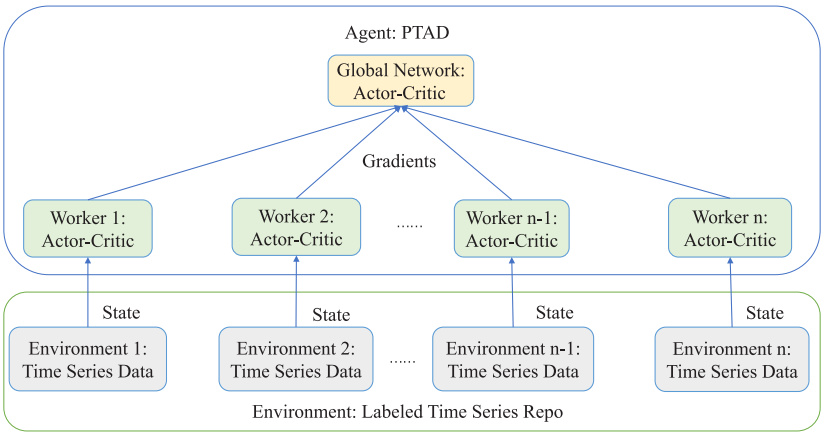


Fig. 1. The asynchronous interactions between the PTAD (the agent) and the labeled time series repository (the environment).

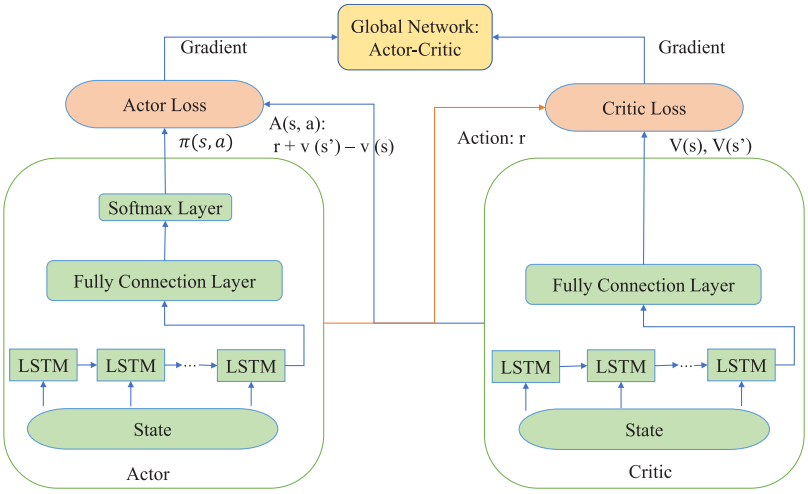


Fig. 2. The internal structure of the PTAD.

and previous n decisions as input and outputs a new decision for the next time stamp. Since the whole optimal process is not related to any assumptions and constraints, the agent could be applied in similar time series anomaly detection tasks.

4.1. The proposed approach

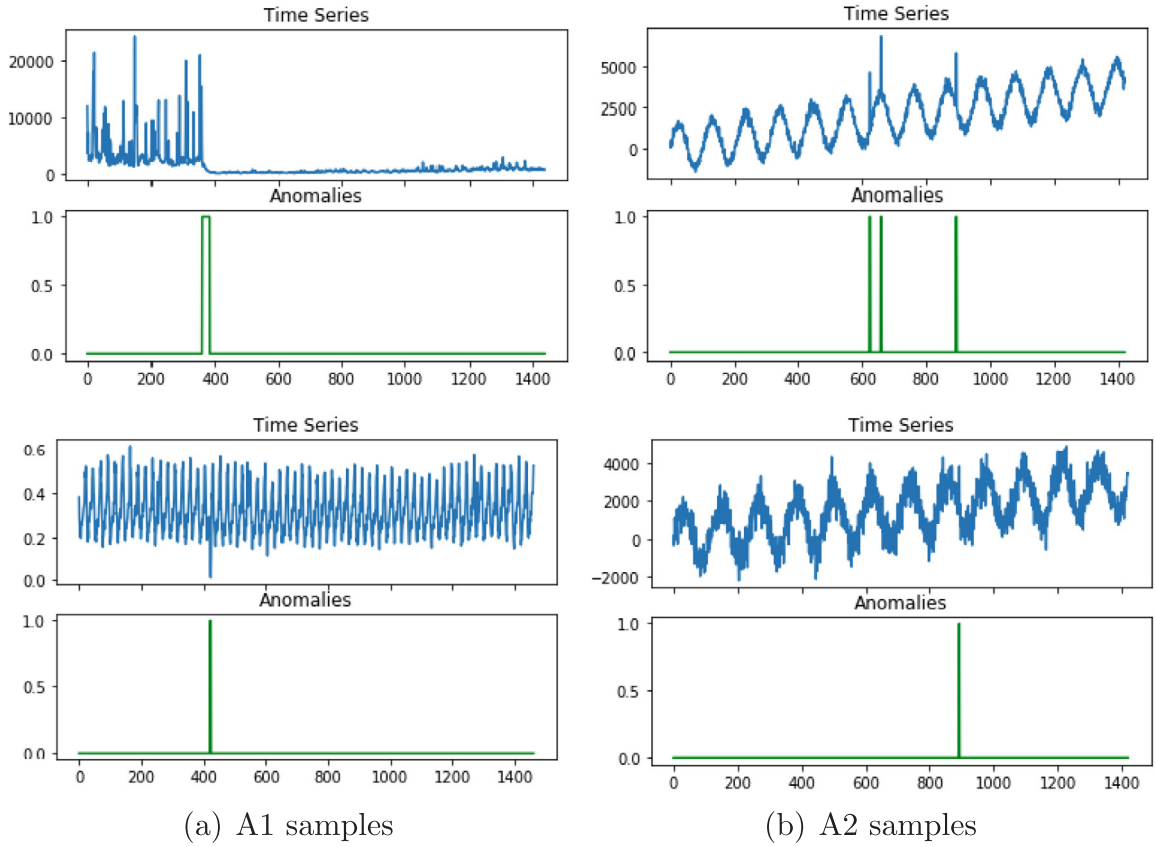
We construct the PTAD with the A3C algorithm, which adopts an asynchronous mechanism for decreasing correlations between the successive examples. Fig. 1 illustrates the overall asynchronous interactions between the PTAD (the agent) and the labeled time series repository (the environment). The below box marks the environment. There are n independent environments which contain the whole labeled time-series data but rank these sequences inconsistently. Each environment provides time stamps of distinct time series as states for its worker and change itself by the received actions. The upper box indicates the PTAD which possesses a global network and n local network, also called workers. All networks take the actor–critic framework. Every worker explores an individual environment and calculates the gradients with the rewards sent by its corresponding environment. In our experiment, every agent owns a different initial environment that could improve the anomaly detection performance because different agents learn from different time series at the same time in order to avoid overfitting some specific abnormal patterns. The global network collects the gradients from the workers and optimizes the targeted policy.

Fig. 2 shows the internal structure of the PTAD, which maintains three major components. A Recurrent Neural Network (RNN) implemented by the Long–Short Term Memory (LSTM) is used to extract the sequential information within the state and output the encoded features to the next unit. The inputs of states are processed similarly by the actor network which estimates the policy and the critic network which approximates the state-value function. Taking the outputs from the RNN as inputs, the fully connection layer yields two values, i.e., $P(a = 0|s)$ and $P(a = 0|s)$, indicating the possibilities of two actions in the actor network and outputs a value of the current state in the critic network. There exists a softmax layer to normalize the outputs of the fully connection layer into the range $[0, 1]$ and give the final decision of the action for the current states by probabilities in the actor network.

These workers do not update the target policy but collect samples to compute gradients. The losses and the gradients in the actor network are calculated with the policy gradient theorem that is involved with the policy π and the advantage function $A(s, a)$ given by the critic network. The losses are the differences between $r+v(s')$ and $v(s)$ where r is obtained after the action taken by the actor network under the given state s in the critic network. Once the global network updates its parameters, it should pass them to local networks to keep consistent.

4.2. Trait comparison

Compared with the value-based DRL anomaly detector, the proposed PTAD has several potential advantages. The value-based detector generates a deterministic policy which takes each action under specific states unchangeably. However, the PTAD could modulate the threshold of judging whether the current state is an anomaly to alter the decision because it yields a stochastic policy. It operates an



(a) A1 samples (b) A2 samples

Fig. 3. Yahoo benchmark samples. For each subfigure, the upper line means the original time series and the lower line indicates the current state where 0 represents normal and 1 marks abnormal.

advantageous tradeoff between precision and recall for some certain demands, e.g., assuring higher precision in cloud operation anomaly detection to relieve heavy work burdens of operation engineers or greater recall is essential in anomaly detection for healthcare because missing anomalies may lead to increases in health risks of patients.

5. Experiments

In this section, we will demonstrate the effectiveness of PTAD by performing experiments on two classical time series datasets compared to the state-of-the-art anomaly detection methods.

5.1. Datasets

We experiment on two classical temporal anomaly detection datasets which are Yahoo benchmark dataset and Numenta Anomaly Detection (NAB) dataset.

Yahoo Benchmark dataset[1] The dataset consists of real and synthetic tagged anomaly time series and contains four subsets, called A1, A2, A3, and A4. In our experiment, Yahoo A1 and A2 benchmark datasets are selected for testing the capability of different anomaly detectors on the same source and target datasets. There are 67 time series and 100 time series in A1 and A2, respectively. A1 benchmark dataset contains real Yahoo membership login data and has complex temporal patterns. Each time series has various anomaly types, even has different lengths. Compared with the A1 benchmark dataset, it is relatively easy to detect the anomalies because there are almost point anomalies and all time series have the same lengths in the A2 benchmark dataset.

Fig. 3 shows some examples of the two datasets. Fig. 3(a) indicates that the A1 Benchmark time series data have no obvious anomaly behaviors through artificial recognition and relatively Fig. 3(b) shows the distinct anomaly patterns even though there exist some disturbances on the A2 benchmark dataset. When comparing the performance of various anomaly detectors on different source and target datasets, we select the whole Yahoo benchmark dataset which comprises 367 time series as the training set.

NAB dataset[2] NAB dataset is usually used for evaluating anomaly detection algorithms in streaming, real-time applications. It includes 58 labeled real-world or synthetic time series, each with 1000–22000 time stamps. For example, realTraffic subset contains real-time traffic data from the Twin Cities Metro area in Minnesota and artificialNoAnomaly subset is artificial without anomalies. The anomaly patterns are also complicated and there is no single anomaly detection method that can distinguish all anomalies. We draw some representative examples as Fig. 4. It shows that some of these time series are periodic and whether a sharp increase is an anomaly or not is related to the contextual information.

5.2. Evaluation metric

We employ the widely used F_1 scores to measure the quality of different anomaly detection models. The metric F_1 is defined as follows

$$F_1 = \frac{2 * precision * recall}{precision + recall}, \tag{12}$$

where $precision = \frac{TP}{TP+FP}$ and $recall = \frac{TP}{TP+FN}$. From the definition, we can see that a larger F_1 scores indicates a better performance.

[1] https://webscope.sandbox.yahoo.com/catalog.php?datatype=a.

[2] https://github.com/numenta/NAB.

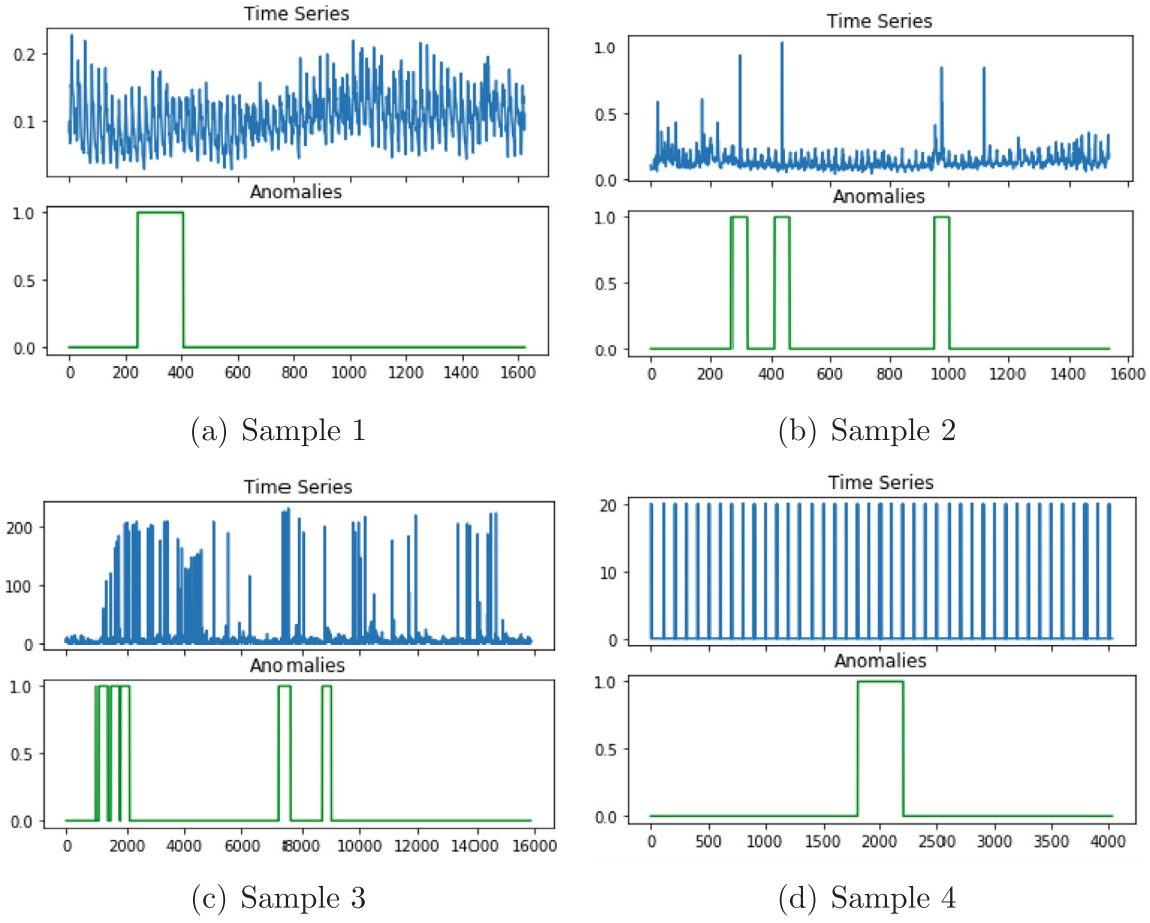


Fig. 4. Numenta benchmark samples. The anomaly patterns are difficult to grasp and whether a sharp increase is an anomaly or not is related to the contextual information.

Table 2
Parameter settings of the PTAD model.

Notation	Value	Description
n_steps	25	The number of LSTM cells
n_input_dim	2	The dimension of the input for each LSTM cell
n_hidden_dim	64	The dimension of hidden layer in each LSTM cell
n_output_dim	2	The dimension of the output in Softmax layer
n_workers	8	The number of threads in a multi-core CPU
update_global_iter	5	The number of steps for delivering gradients
γ	0.99	The discount factor
entropy_beta	0.01	The coefficient of entropy regularization
lr_a	0.001	The learning rate of the actor
lr_c	0.0001	The learning rate of the critic
N	20000	The number of training episodes
A	5	The instant reward for TP
-B	−1	The instant reward for FP
-C	−5	The instant reward for FN
D	1	The instant reward for TN

Table 3
Comparisons of the required memory and the simulation time for testing the "realAdExchange/null-exchange-2-cpm-results" time series in Numenta. The bold indicates the best and the underline means the worst.

Models	Memory (MB)	Time (s)
Skyline	352.5	<u>32.07</u>
Twitter	**103.7**	4.93
ContextOSE	347.1	6.90
Numenta	467.5	10.22
Numenta TM	409.6	7.97
RNN-TAD	231.8	**1.77**
VTAD	<u>996.7</u>	10.87
PTAD	770.5	2.10

5.3. Comparing methods and experimental setups

We consider the following methods to compare:

Skyline (Esty, 2014): This method calculates time series anomaly scores by voting from different expert detectors.

Twitter (Twitter, 2015): It is based on seasonal hybrid extreme studentized deviate algorithm and performs excellent in seasonal univariate time series.

ContextOSE (Mikhail, 2015): It is based on contextual anomaly detection, which captures the local rather than global information. The procedure is selecting a subset of time series, calculating the centroid of the selected time series and then predicting the value of time series with the centroid and other features.

Numenta and Numenta TM (Ahmad et al., 2017): These detection methods are based on Hierarchical Temporal Memory (HTM). At the core of HTM are time-based continuous learning algorithms that store and recall spatial and temporal patterns.

RNN-TAD (Malhotra et al., 2015): This method is a neural network based anomaly detector. It learns a predictive model from the normal training time stamps based on the LSTM and marks normal or abnormal depending on the error between the predictive values and true values.

VTAD (Huang et al., 2018): It is the value-based DRL time series anomaly detector which adopts the DQN algorithm.

The model parameters of these methods are set as consistent as the references. Notably, our implement of the VTAD is a standard Q-learning process with the scalar reward, which is slightly different from what Huang et al. (2018) do where they calculated the loss with all

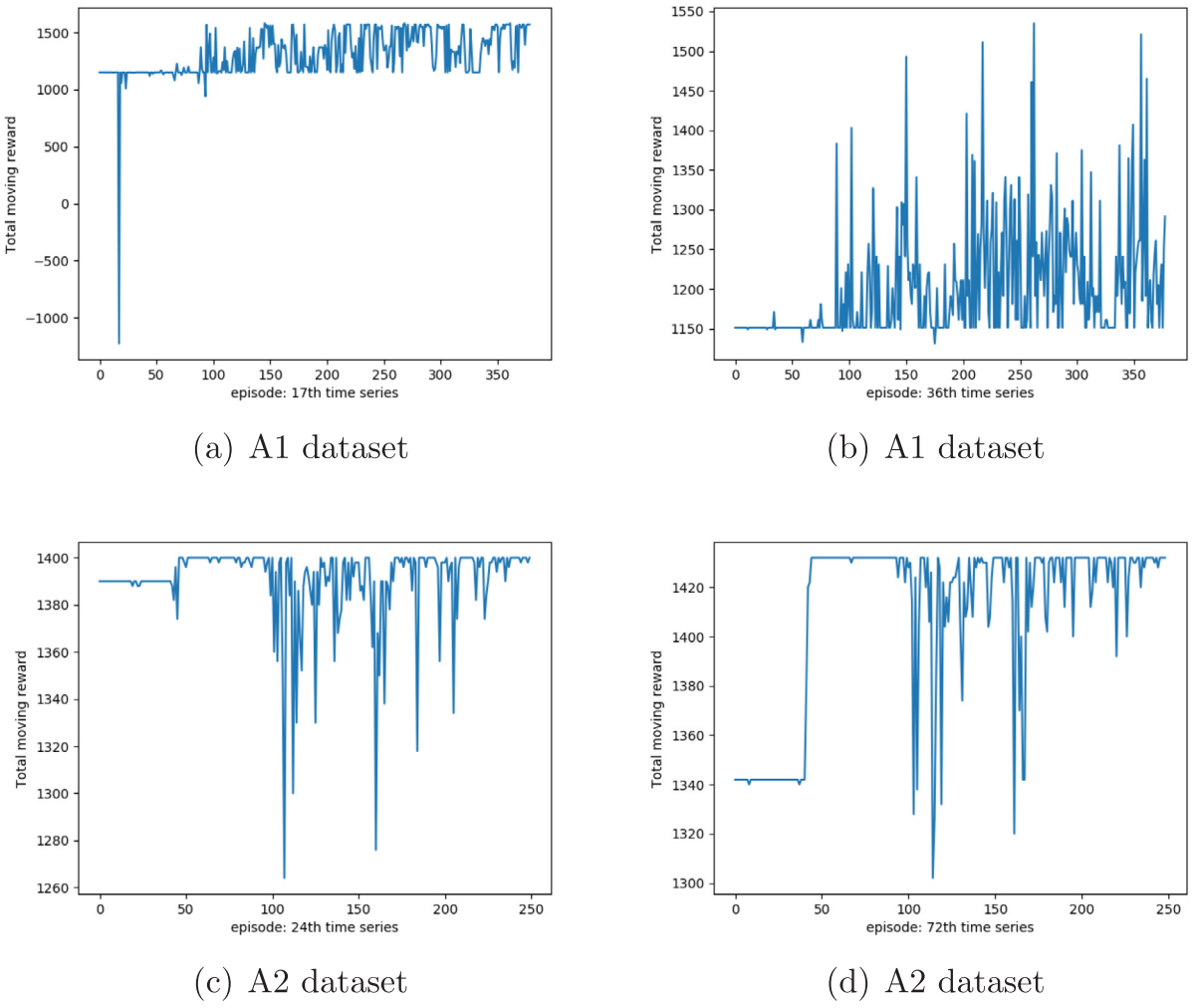


Fig. 5. Training rewards of single time series with PTAD model in the Yahoo A1 and A2 benchmark datasets.

Table 4
Performance comparisons among Twitter anomaly detection, RNN-TAD, VTAD and PTAD on the test part of the Yahoo A1 benchmark datasets. The bold indicates the best.

Index of examples	Twitter	RNN-TAD	VTAD	PTAD
1	0.07	**1.00**	0.50	0.67
2	0.80	0.93	**0.95**	0.86
3	0.02	0.56	**0.81**	0.56
4	0.50	**0.95**	0.81	0.79
5	**0.63**	0.40	0.50	0.50
6	0.55	0.49	**0.80**	0.70
7	0.04	0.09	0.08	**0.63**
8	0.35	0.73	**1.00**	**1.00**
9	**0.62**	0.35	0.35	0.60
10	**0.25**	0.00	0.08	0.06
11	**0.34**	0.29	0.21	0.29
12	0.67	0.78	**0.90**	**0.90**
13	0.79	0.70	**1.00**	**1.00**
Arithmetic Mean	0.43	0.56	0.61	**0.66**
Standard Variation	0.27	0.31	0.33	**0.26**
Numbers of $F_1 > 0.5$	6	7	7	**11**

actions

$$\mathbb{E}_{a\in\mathcal{A},batch}\left[\left(r+\gamma\max_{a'}Q(s',a'|\theta_{i-1})-Q(s,a|\theta_i)\right)^2\right],$$

not the action a^* selected by the target policy

$$\mathbb{E}_{batch}\left[\left(r+\gamma\max_{a'}Q(s',a'|\theta_{i-1})-Q(s,a^*|\theta_i)\right)^2\right],$$

because they adopted a vector reward function.

Detailed settings of PTAD's hyperparameters are provided in Table 2.

Our PTAD model is trained with a multi-core CPU (8 threads) without the GPU, which is convenient for implementation. Compared with the deep learning based methods which are the RNN-TAD and the VTAD, the training time of our model is longer. To better evaluate the capability of the proposed PTAD model, we present Table 3 to compare the required memory and the simulation time for testing the "realAdExchange/null-exchange-2-cpm-result" time series in Numenta. On one hand, our model requires a relatively large memory, but it is better than the VTAD model. On the other hand, it costs less test time, which is only a little worse than the RNN-TAD.

5.4. Comparison results

The comparisons among different time series anomaly detectors are not only on the same but also on different training and test datasets.

5.4.1. Performance on the same source and target datasets

We compare the performance of Twitter anomaly detection, RNN-TAD, VTAD and PTAD on the test part of the Yahoo A1 and A2 benchmark datasets. For the Twitter anomaly detector, it just analyzes statistical characteristics on each time series without training. For the other three detectors, all data in each benchmark dataset are originally divided into training and test parts by a ratio of 8:2. Hence, there are 13 test time series in the A1 benchmark dataset and 20 test time series

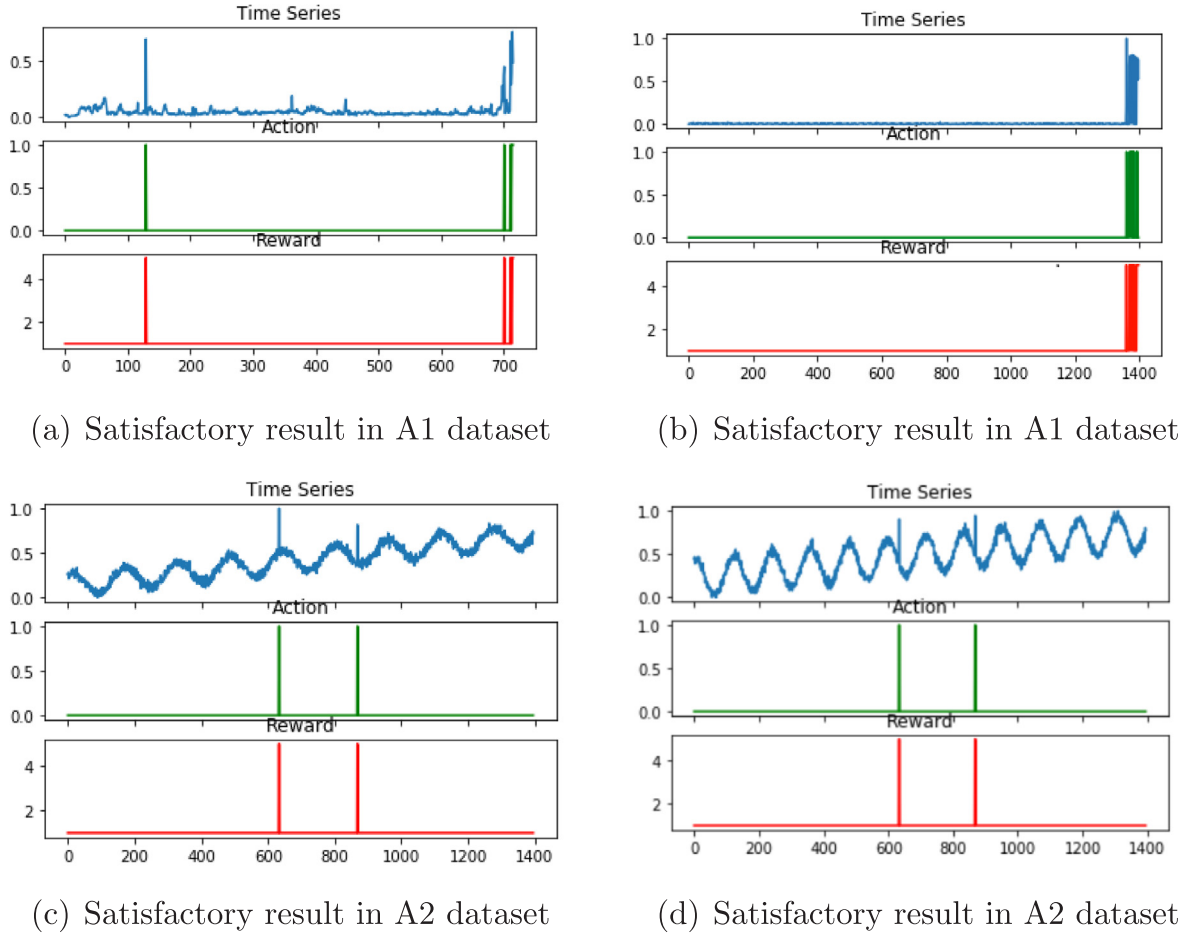


Fig. 6. Satisfactory detection results of PTAD in the Yahoo A1 and A2 benchmark datasets. For each subfigure, the raw time series are shown in the upper lines and the middle lines mean decisions of the PTAD and the lower lines evaluate judgments via rewards. The reward is 5 if the detector correctly distinguishes an anomaly, otherwise −1. The reward is 1 if the detector correctly judges a normal, otherwise −5.

Table 5
Performance comparisons among twitter anomaly detector, RNN-TAD, VTAD and PTAD on the test part of the Yahoo A2 benchmark datasets. The bold indicates the best.

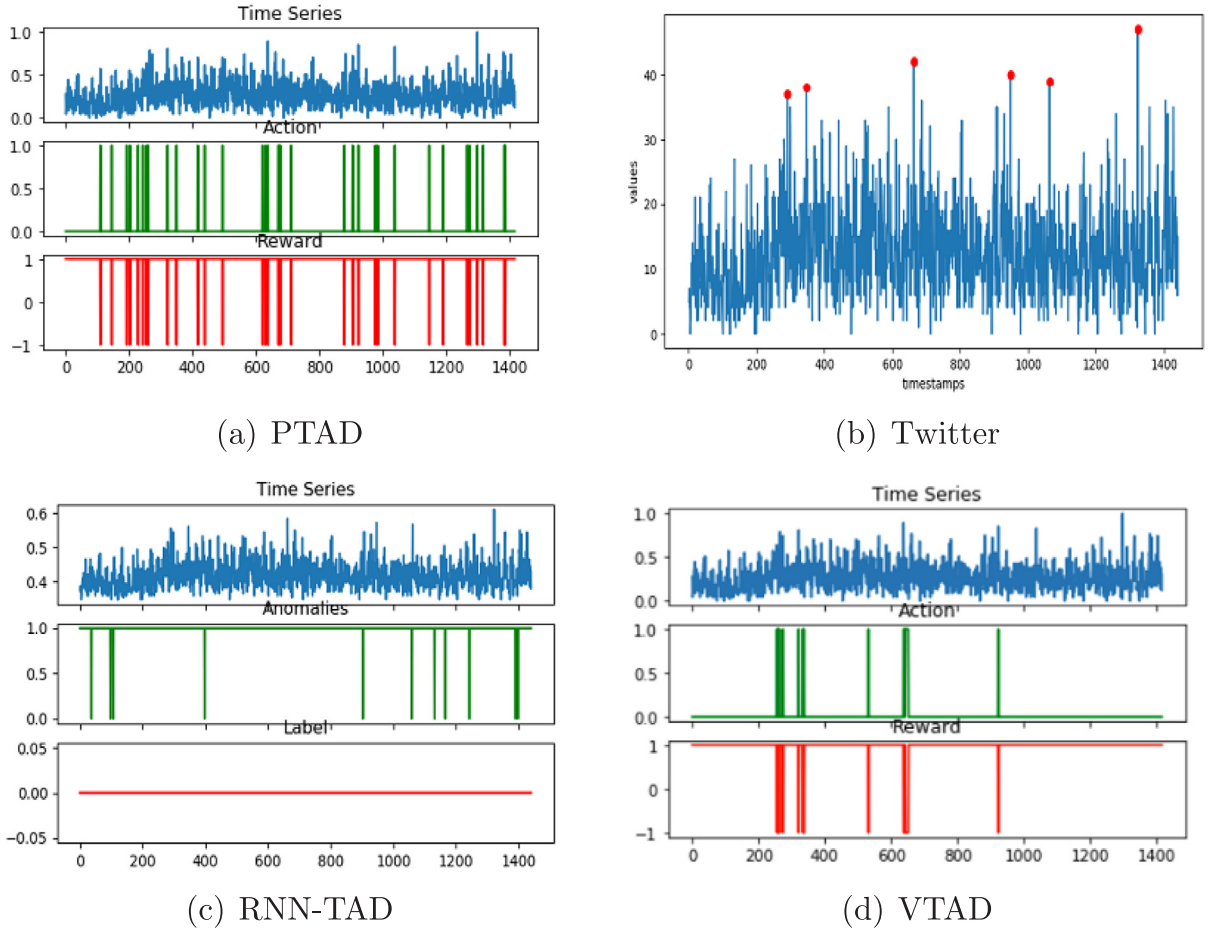
Index of examples	Twitter	RNN-TAD	VTAD	PTAD
1	0.33	**1.00**	**1.00**	**1.00**
2	0.67	**1.00**	**1.00**	**1.00**
3	0.33	**1.00**	**1.00**	**1.00**
4	0.70	**1.00**	**1.00**	**1.00**
5	0.67	**1.00**	**1.00**	**1.00**
6	0.33	**1.00**	**1.00**	**1.00**
7	0.18	**1.00**	**1.00**	**1.00**
8	0.67	**1.00**	**1.00**	**1.00**
9	0.33	**1.00**	**1.00**	**1.00**
10	0.70	**1.00**	**1.00**	**1.00**
11	0.67	**1.00**	**1.00**	**1.00**
12	0.60	**1.00**	**1.00**	**1.00**
13	0.18	**1.00**	**1.00**	**1.00**
14	0.50	**1.00**	**1.00**	**1.00**
15	0.33	**1.00**	**1.00**	**1.00**
16	0.59	**1.00**	**1.00**	**1.00**
17	0.67	**1.00**	**1.00**	**1.00**
18	0.60	**1.00**	**1.00**	**1.00**
19	0.18	**1.00**	**1.00**	**1.00**
20	0.67	**1.00**	**1.00**	**1.00**
Arithmetic Mean	0.50	**1.00**	**1.00**	**1.00**
Standard Variation	0.19	**0.00**	**0.00**	**0.00**
Numbers of $F_1 > 0.5$	12	**20**	**20**	**20**

in the A2 benchmark dataset. We eliminate abnormal time stamps in training set for the RNN-TAD to construct a normal predictor.

Since the environment of the RL setting contains various time series of different lengths and anomaly patterns, there are high fluctuations in the overall trends of cumulative expectation rewards. For example, it is a good performance for a sequence of length 1420 to achieve the total reward of 1430, while it is worse for another of length 1600. Therefore, we show the training reward curves of single time series as Fig. 5. Because the number of anomalies is relatively small, a random policy can be highly rewarded at the beginning. The rewards are promoted in an oscillatory manner during the training process.

Table 4 shows the comparison results of test examples on the Yahoo A1 benchmark dataset. Although the PTAD is not the best in every test time series, it outperforms other methods averagely. The minimal standard variation is also achieved by the proposed PTAD. At the same time, the detection results of Twitter anomaly detector, RNN-TAD, and VTAD contain only 6, 7, 7 over 0.5 F_1 scores in the 13 test time series, which illustrates that they cannot generalize well in the A1 benchmark detection task. The PTAD almost gets F_1 scores over 0.5 except the 10th and 11th examples, which shows more stable anomaly detection performance. Furthermore, the two RL based detectors achieve better results compared with Twitter anomaly detector and RNN-TAD, which illustrates that RL is an effective tool for the time series anomaly detection problem.

Table 5 shows the comparison results of test examples on the Yahoo A2 benchmark dataset. Since the data in Yahoo A2 benchmark dataset have relatively simple anomaly patterns which almost are the point anomalies, the RNN-TAD, VTAD, and PTAD get perfect results on all test time series. However, the Twitter anomaly detector receives a poor result in this detecting task.



(a) PTAD (b) Twitter

(c) RNN-TAD (d) VTAD

Fig. 7. Comparing the unsatisfactory detection results of the 10th time series in the A1 dataset. For subfigures of PTAD and VTAD, the raw time series are shown in the upper lines and the middle lines mean decisions of the PTAD and the lower lines evaluate judgments via rewards. For Twitter, the anomalies are marked as dots and the raw time series are shown in lines. For RNN-TAD, the raw time series are marked as the upper lines and the middle lines mean decisions of the RNN-TAD and the lower lines evaluate judgments via true labels.

Table 6
Performance comparisons among Twitter, Skyline, Numenta, Numenta TM, contextOSE, RNN-TAD, VTAD and PTAD where the last three methods are trained on Yahoo Benchmark dataset and are tested on Numenta dataset. The bold indicates the best.

Name of subsets	Twitter	Skyline	Numenta	Numenta TM	contextOSE	RNN-TAD	VTAD	PTAD
artificialNoAnomaly	0.72	**1.00**	0.80	**1.00**	0.87	**1.00**	0.21	0.20
artificialWithAnomaly	0.00	0.05	0.02	0.02	0.01	0.19	**0.73**	0.59
realAdExchange	0.01	0.02	0.05	0.05	0.03	0.16	**0.74**	0.71
realAWSCloudwatch	0.06	0.12	0.05	0.03	0.04	0.24	0.59	**0.60**
realKnownCause	0.02	0.01	0.02	0.02	0.01	0.25	0.33	**0.56**
realTraffic	0.03	0.10	0.04	0.05	0.03	0.25	0.71	**0.73**
realTweets	0.00	0.04	0.01	0.01	0.01	0.09	0.62	**0.67**
Arithmetic mean	0.12	0.19	0.14	0.17	0.14	0.31	0.56	**0.58**
Standard variation	0.25	0.33	0.27	0.34	0.30	0.29	**0.19**	**0.19**
Numbers of $F_1 > 0.5$	1	1	1	1	1	1	5	**6**

Fig. 6 shows several satisfactory detection results of PTAD in Yahoo A1 and A2 benchmark datasets where the PTAD model precisely finds out all the anomalies.

There is also an unsatisfactory result shown in Fig. 7(a). Since the raw time series seems too complicated and has no obvious anomaly patterns, it is troublesome to justify whether a sharp increase is an anomaly. We also give other detection results that are obtained by applying the Twitter, TNN-TAD and VTAD models in the same 10th time series in the A1 benchmark dataset. Fig. 7(b) shows the anomaly detections with the Twitter model, which gets the highest F1 score of 0.25 in this example. However, as 7(c) displays, the RNN-TAD model completely cannot identify the normal and abnormal behaviors. The VTAD model achieve a little better performance than PTAD, which is shown in Fig. 7(d).

5.4.2. Performance on different source and target datasets

We present comparison results of the Twitter, Skyline, Numenta, Numenta TM, contextOSE, RNN-TAD, VTAD and PTAD on source the Yahoo benchmark dataset and target the NAB dataset. For Twitter, Skyline, Numenta, Numenta TM, and contextOSE anomaly detector, they predict whether next time stamp is abnormal by modeling the current sequences in a given time series, which means that there is no need to train with the Yahoo dataset. For RNN-TAD, VTAD and PTAD, the whole Yahoo benchmark dataset is used for training and similarly, we remove the abnormal behaviors for the RNN-VTAD to learn a normal predictor.

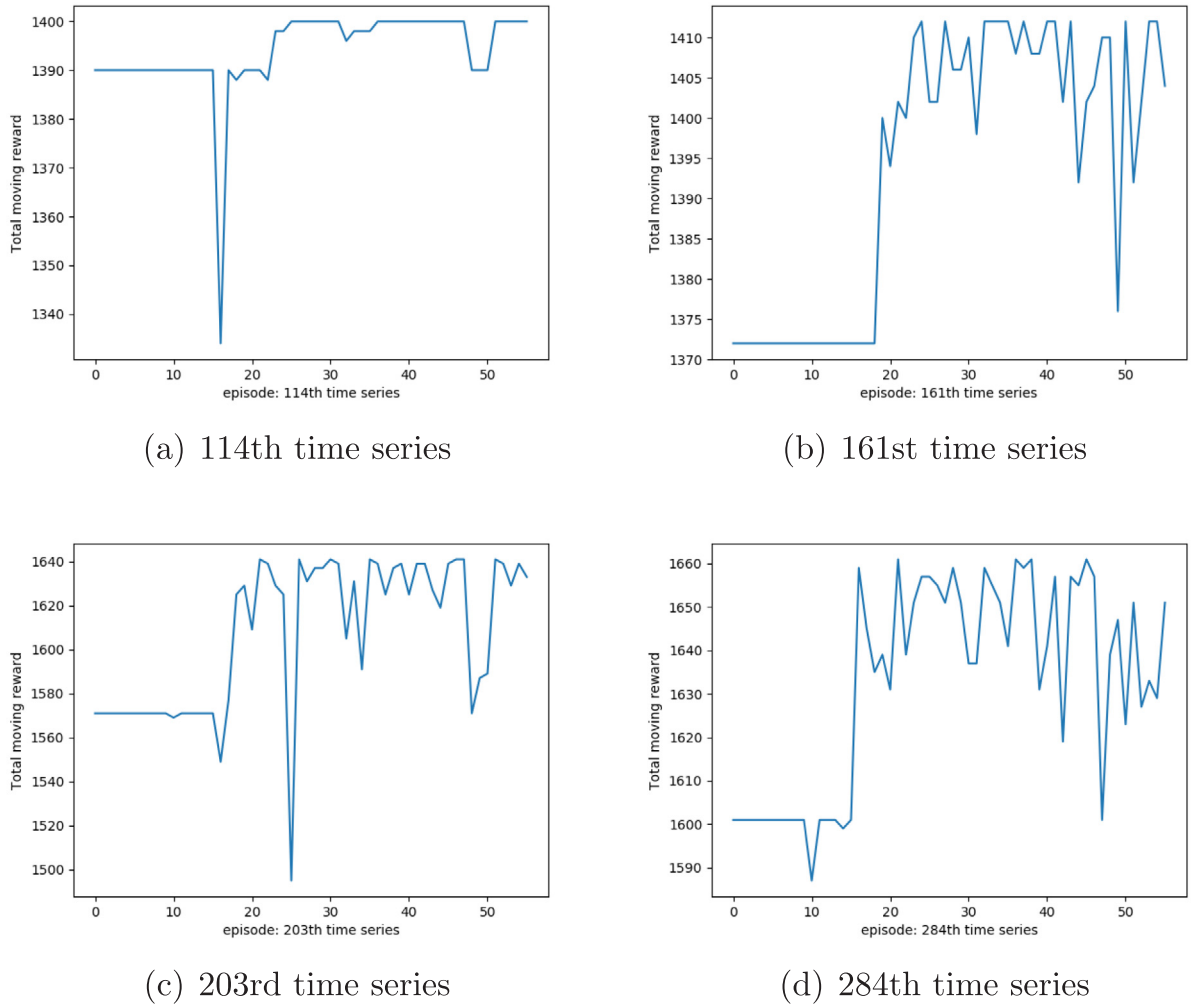


Fig. 8. Training rewards of single time series with PTAD model in the source Yahoo benchmark datasets.

Fig. 8 shows the expectation rewards of single time series on the whole Yahoo benchmark dataset in the training phase. Similarly, each time series acquires a basic reward with the initial policy since the number of anomalies is relatively small. The reward then climbs at about the 20th episode and fluctuates during the subsequent training episodes.

Table 6 shows the average detection results of seven subsets and the best performance is marked as the bold. PTAD outperforms other detectors including the VTAD in most subsets of NAB datasets. Compared with these state-of-the-art open-source detection techniques, deep learning based methods including RNN-TAD, VTAD, and PTAD discover time series anomalies more precisely and completely. Other than the artificialNoAnomaly subset, the RL based time series anomaly detectors achieve superior performance than other non-RL methods. For the artificialNoAnomaly subset, the first six techniques, especially Skyline, Numenta TM and RNN-TAD, successfully make a decision that there exists no anomaly, while VTAD and PTAD fail to give a correct judgment. It is probably because these RL based detectors do not learn the periodicity from the whole Yahoo benchmark which lacks periodic data and regard every periodic increase as anomaly mistakenly.

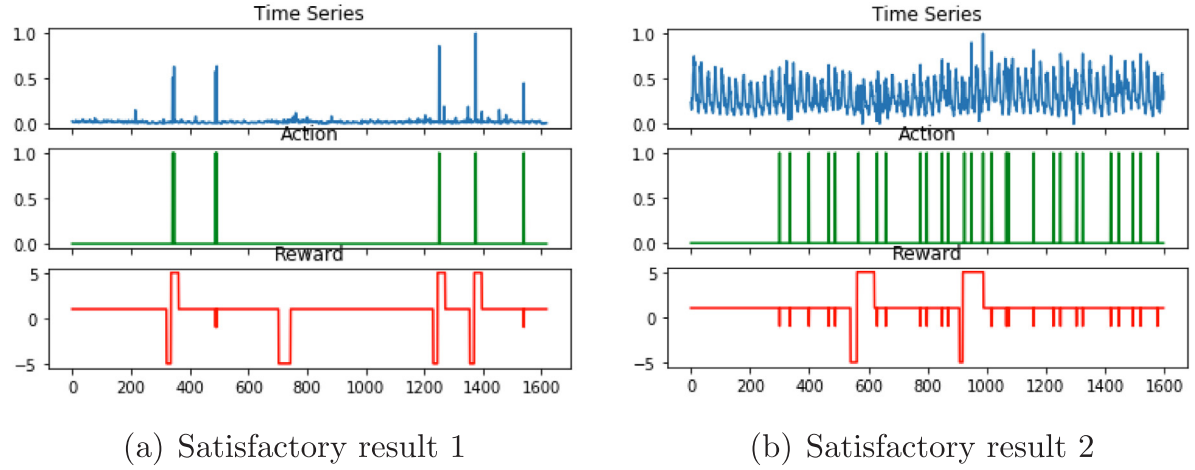
We show some satisfactory detecting results of PTAD on the NAB datasets as 9(a) and 9(b). Fig. 9(a) illustrates whether a sharp increase is an anomaly depends on the context and practical information since the first rise in 200–400 time stamp is labeled as an anomaly, while the second climb in 400–600 time stamp is not. Fig. 9(b) exposes the complicated data patterns and unseen anomalies.

Fig. 10(a) shows an unsatisfactory detection result of the "null-art-daily-perfect-square-wave" time series which is extracted from the artificialNoAnomaly subset. The feature of periodic variations is not learned by the PTAD and the predicting results seem a bit bad. VTAD, the RL-based model, also gets a poor $F1$ score which is displayed as Fig. 10(d). However, Twitter and RNN-TAD grasp the periodicity and predict no abnormal behaviors in this time series, which are shown in Figs. 10(b) and 10(c), respectively.

5.5. Adjustability of the PTAD

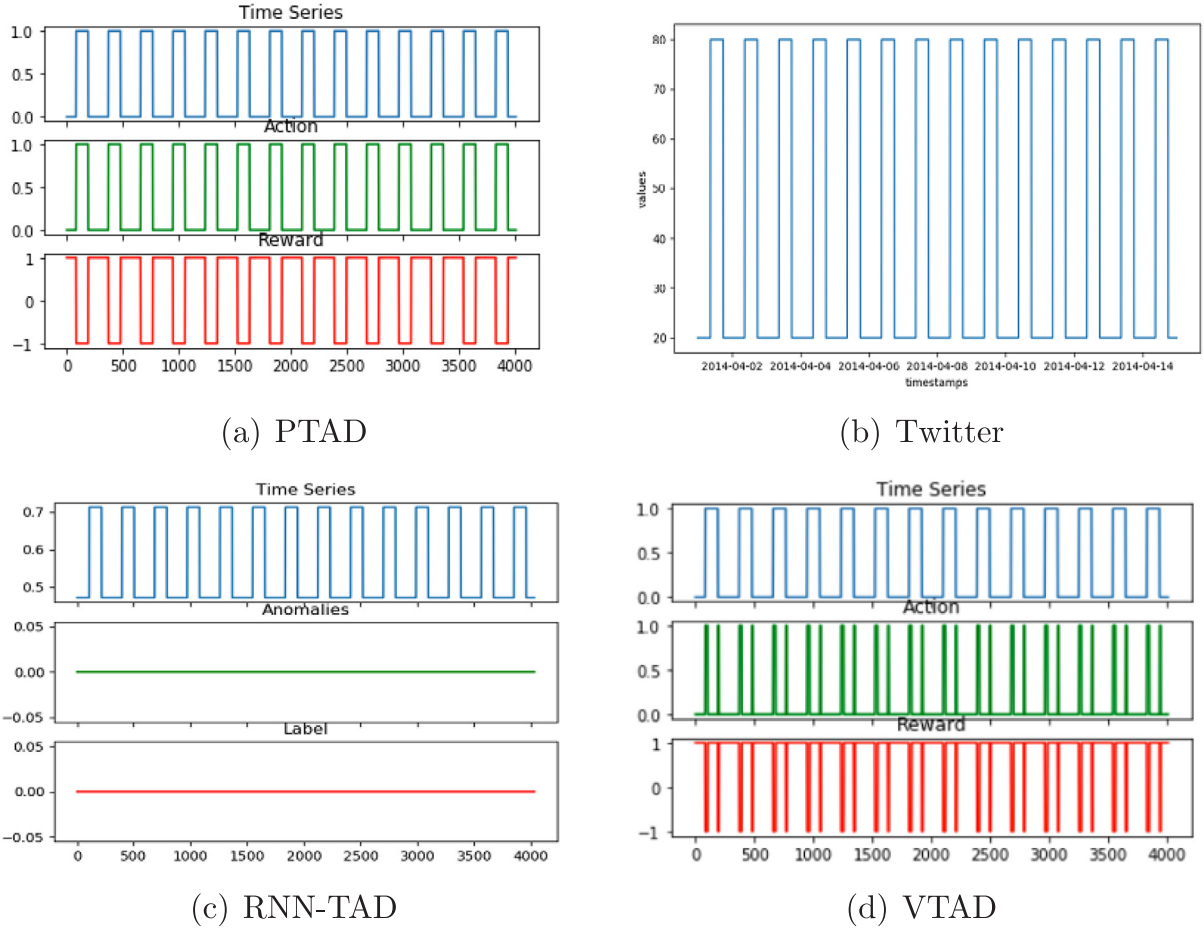
Since the optimal policy of the PTAD gives the probabilities of different decisions, the threshold of justifying an anomaly can be adjusted for the higher precision or the higher recall. When the threshold is set higher, the detector reports an anomaly more discreetly, which is suited for applications with high precision. Conversely, if a higher recall is required, the threshold of judging an anomaly can be set lower. We illustrate the impact of setting different thresholds by try and error. Fig. 11 presents variations of the precision, the recall and the $F1$ score under different thresholds in the "realKnownCause/null-nyc-taxi" and "realTraffic/null-occupancy-6005" time series. With the increase of the threshold, the precision improves while the recall decreases. It seems that our PTAD can control the tradeoff between the precision and the recall to meet realistic requirements in different applications. However, the previous VTAD does not possess this merit.

Fig. 12 shows some precision-improved results where we determine to give a chance of reporting an anomaly when the probability of



(a) Satisfactory result 1 (b) Satisfactory result 2

Fig. 9. Satisfactory detection results of PTAD in NAB datasets. For each subfigure, the raw time series are shown in the upper lines and the middle lines mean decisions of the PTAD and the lower lines evaluate judgments via rewards. The reward is 5 if the detector correctly distinguishes an anomaly, otherwise −1. The reward is 1 if the detector correctly judges a normal, otherwise −5.



(a) PTAD (b) Twitter

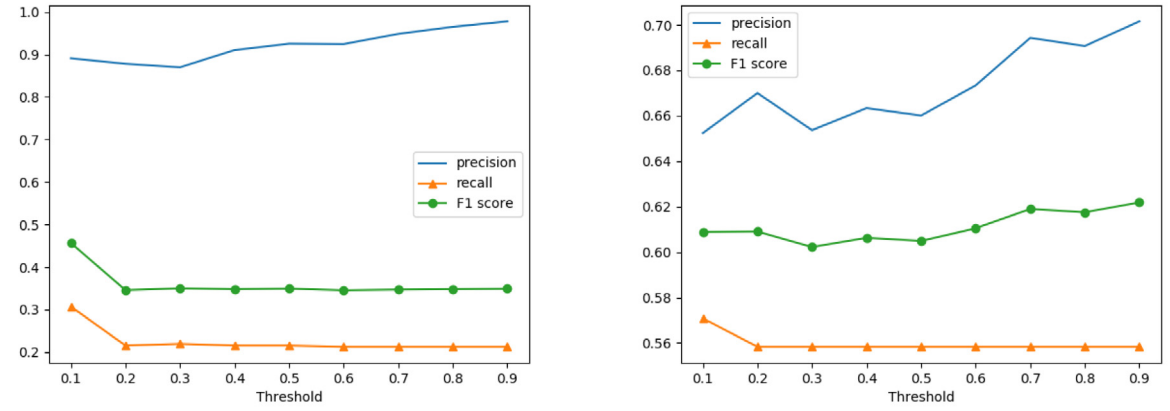
(c) RNN-TAD (d) VTAD

Fig. 10. Comparing the unsatisfactory detection results of the "null-art-daily-perfect-square-wave" time series in the Numenta dataset. For subfigures of PTAD and VTAD, the raw time series are shown in the upper lines and the middle lines mean decisions of the PTAD and the lower lines evaluate judgments via rewards. For Twitter, the anomalies are marked as dots and the raw time series are shown in lines. For RNN-TAD, the raw time series are marked as the upper lines and the middle lines mean decisions of the RNN-TAD and the lower lines evaluate judgments via true labels.

justifying an anomaly exceeds 0.9. Fig. 12(a) shows the original detection results and Fig. 12(b) illustrates the precision-improved detection results. It is obvious that the improved detector eliminates numerous uncertain detections, which significantly raises the precision.

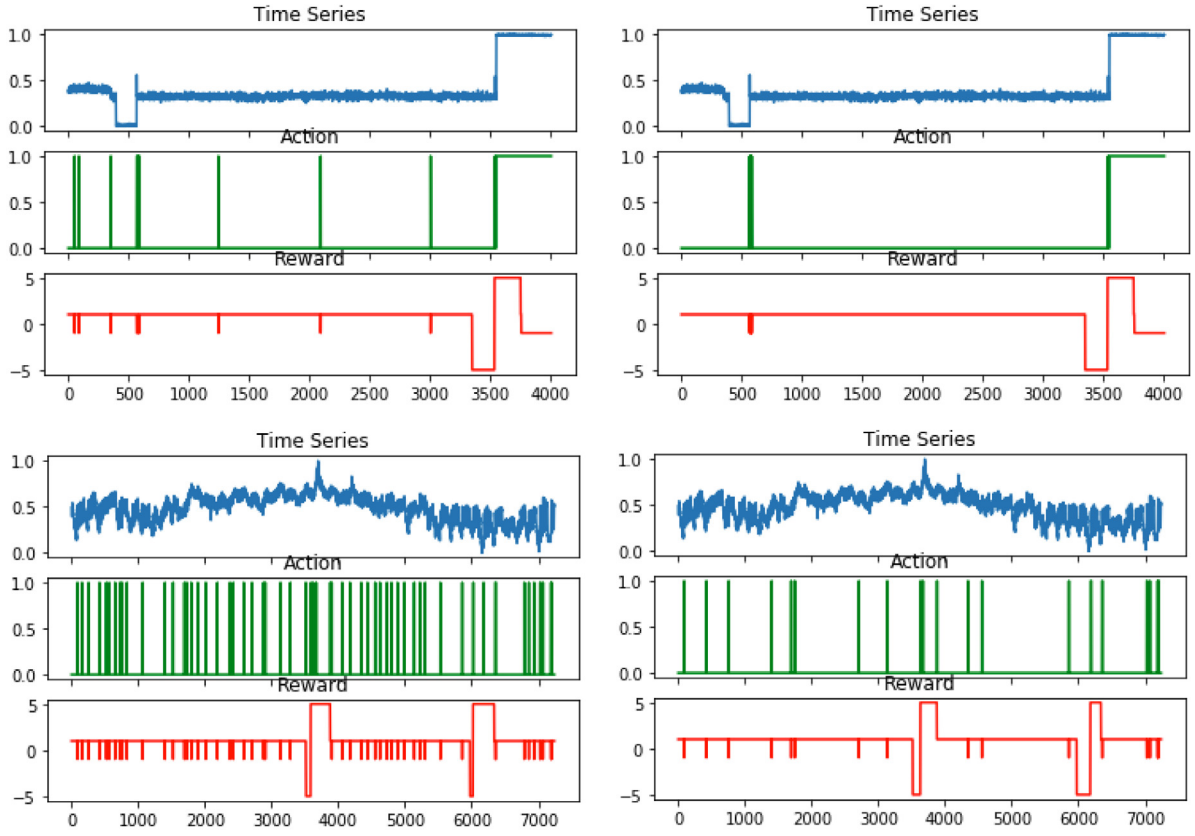
6. Conclusion

We have proposed a general policy-based RL framework to settle the time series anomaly detection problem, which utilizes the A3C



(a) realKnownCause/null-nyc-taxi (b) realTraffic/null-occupancy-6005

Fig. 11. Variations of the precision, the recall and the F1 score under different thresholds in the "realKnownCause/null-nyc-taxi" and "realTraffic/null-occupancy-6005" time series.



(a) Original Results (b) Precision-improved Results

Fig. 12. Precision improved NAB results with the PTAD. For each subfigure, the raw time series are shown in the upper lines and the middle lines mean decisions of the PTAD and the lower lines evaluate judgments via rewards. The reward is 5 if the detector correctly distinguishes an anomaly, otherwise −1. The reward is 1 if the detector correctly judges a normal, otherwise −5. The improved detector eliminates numerous uncertain detections, which significantly raises the precision.

algorithm to model a sequential anomaly detector. Compared with the value-based RL time series anomaly detector, our detector performs better in detection performance and occupies less computation complexity. It also outperforms other time series anomaly techniques not only on the same but also on different source and target datasets with no requirements of analyzing the characteristic of the full data and making strong assumptions. Furthermore, our detector generates a stochastic policy which can explore the tradeoff between the precision and the recall for meeting practical requirements.

As future work, we will investigate how to integrate more information, e.g., periodicity, into the design of anomaly detectors. Furthermore, it is interesting to model a normal RL predictor which outputs the value of the next time stamp in the training phase and gives the final detection results on each test time series according to the error between predictive state and true state. It could reduce the dependency on labels and then extend to the more common semi-supervised and unsupervised settings.

CRediT authorship contribution statement

Mengran Yu: Methodology, Software, Writing - original draft, Writing - review & editing. **Shiliang Sun:** Conceptualization, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Project 61673179, Shanghai Knowledge Service Platform Project (No. ZF1213), and the Strategic Priority Research Program of ECNU.

References

Agarwal, D., 2005. An empirical Bayes approach to detect anomalies in dynamic multidimensional arrays. In: International Conference on Data Mining. pp. 8–16.
Agarwal, D., 2007. Detecting anomalies in cross-classified streams: a Bayesian approach. Knowl. Inf. Syst. 11 (1), 29–44.
Ahmad, S., Lavin, A., Purdy, S., Agha, Z., 2017. Unsupervised real-time anomaly detection for streaming data. Neurocomputing 262, 134–147.
Amarbayasgalan, T., Jargalsaikhan, B., Ryu, K., 2018. Unsupervised novelty detection using deep autoencoders with density based clustering. Appl. Sci. 8 (9), 1468.
Bahdanau, D., Brakel, P., Xu, K., Goyal, A., Lowe, R., Pineau, J., Courville, A., Bengio, Y., 2016. An actor-critic algorithm for sequence prediction. pp. 1–10, ArXiv Preprint arXiv:1607.07086.
Bianco, A.M., Garcia Ben, M., Martinez, E., Yohai, V.J., 2001. Outlier detection in regression models with ARIMA errors using robust estimates. J. Forecast. 20 (8), 565–579.
Borghesi, A., Bartolini, A., Lombardi, M., Milano, M., Benini, L., 2019. A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems. Eng. Appl. Artif. Intell. 85, 634–644.
Bourdonnaye, F., Teulière, C., Chateau, T., Triesch, J., 2017. Learning of binocular fixations using anomaly detection with deep reinforcement learning. In: International Joint Conference on Neural Networks. pp. 760–767.
Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J., 2000. LOF: identifying density-based local outliers. In: ACM Sigmod Record, Vol. 29. pp. 93–104.
Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: A survey. ACM Comput. Surv. 41 (3), 15:1–15:58.
Chauhan, S., Vig, L., 2015. Anomaly detection in ECG time signals via deep long short-term memory networks. In: IEEE International Conference on Data Science and Advanced Analytics. pp. 1–7.
Das, K., Schneider, J., 2007. Detecting anomalous records in categorical datasets. In: International Conference on Knowledge Discovery and Data Mining. pp. 220–229.
Dong, Y., 2019. Implementing deep learning for comprehensive aircraft icing and actuator/sensor fault detection/identification. Eng. Appl. Artif. Intell. 83, 28–44.
Esty, 2014. Skyline. https://github.com/etsy/skyline.
Fernández-Sanjurjo, M., Bosquet, B., Mucientes, M., Brea, V.M., 2019. Real-time visual detection and tracking system for traffic monitoring. Eng. Appl. Artif. Intell. 85, 410–420.
Ghasedi Dizaji, K., Wang, X., Huang, H., 2018. Semi-supervised generative adversarial network for gene expression inference. In: International Conference on Knowledge Discovery and Data Mining. pp. 1435–1444.
Gorokhov, O., Petrovskiy, M., Mashechkin, I., 2017. Convolutional neural networks for unsupervised anomaly detection in text data. In: International Conference on Intelligent Data Engineering and Automated Learning. pp. 500–507.
Hu, Y., Palmé, T., Fink, O., 2017. Fault detection based on signal reconstruction with auto-associative extreme learning machines. Eng. Appl. Artif. Intell. 57, 105–117.
Huang, C., Wu, Y., Zuo, Y., Pei, K., Min, G., 2018. Towards experienced anomaly detector through reinforcement learning. In: AAAI Conference on Artificial Intelligence. pp. 8087–8088.
Kumar, G.R., Mangathayaru, N., Narsimha, G., 2016. An approach for intrusion detection using novel Gaussian based kernel function. J. UCS 22 (4), 589–604.
Laptev, N., Amizadeh, S., Flint, I., 2015. Generic and scalable framework for automated time-series anomaly detection. In: International Conference on Knowledge Discovery and Data Mining. ACM, pp. 1939–1947.
Ma, J., Perkins, S., 2003. Time-series novelty detection using one-class support vector machines. In: International Joint Conference on Neural Networks, Vol. 3. pp. 1741–1745.
Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., Shroff, G., 2016. LSTM-based encoder-decoder for multi-sensor anomaly detection. pp. 1–5, Arxiv Preprint arxiv:1607.00148.
Malhotra, P., Vig, L., Shroff, G., Agarwal, P., 2015. Long short term memory networks for anomaly detection in time series. In: European Symposium on Artificial Neural Networks. pp. 89–94.
Mikhail, S., 2015. Contextual anomaly detection. https://github.com/smirmik/CAD.
Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., Kavukcuoglu, K., 2016. Asynchronous methods for deep reinforcement learning. In: International Conference on Machine Learning. pp. 1928–1937.
Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.
Mukkamala, S., Janoski, G., Sung, A., 2002. Intrusion detection using neural networks and support vector machines. In: International Joint Conference on Neural Networks, Vol. 2. pp. 1702–1707.
Oh, D., Yun, I., 2018. Residual error based anomaly detection using auto-encoder in SMD machine sound. Sensors 18 (5), 1308.
Pane, Y.P., Nageshrao, S.P., Babuška, R., 2016. Actor-critic reinforcement learning for tracking control in robotics. In: Conference on Decision and Control. pp. 5819–5826.
Ramaswamy, S., Rastogi, R., Shim, K., 2000. Efficient algorithms for mining outliers from large data sets. In: ACM Sigmod Record, Vol. 29. pp. 427–438.
Shekhar, S., Lu, C.-T., Zhang, P., 2001. Detecting graph-based spatial outliers: algorithms and applications (a summary of results). In: International Conference on Knowledge Discovery and Data Mining. pp. 371–376.
Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT Press.
Tandon, G., Chan, P.K., 2007. Weighting versus pruning in rule validation for detecting network and host anomalies. In: International Conference on Knowledge Discovery and Data Mining. pp. 697–706.
Tang, J., Chen, Z., Fu, A.W.-C., Cheung, D.W., 2002. Enhancing effectiveness of outlier detections for low density patterns. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 535–548.
Tuor, A.R., Baerwolf, R., Knowles, N., Hutchinson, B., Nichols, N., Jasper, R., 2018. Recurrent neural network language models for open vocabulary event-level cyber anomaly detection. In: Workshops at the AAAI Conference on Artificial Intelligence. pp. 285–293.
Twitter, 2015. Twitter anomaly detection. https://github.com/twitter/AnomalyDetection/releases.
Venkataraman, S., Caballero, J., Song, D., Blum, A., Yates, J., 2006. Black box anomaly detection: is it utopian? HotNets 127.
Wang, J., Ding, X., Lahijanian, M., Paschalidis, I.C., Belta, C.A., 2015. Temporal logic motion control using actor–critic methods. Int. J. Robot. Res. 34 (10), 1329–1344.
Wei, J., Zhao, J., Zhao, Y., Zhao, Z., 2018. Unsupervised anomaly detection for traffic surveillance based on background modeling. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 129–136.
Yamanishi, K., Takeuchi, J.-I., Williams, G., Milne, P., 2004. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. Data Min. Knowl. Discov. 8 (3), 275–300.
Zhu, L., Laptev, N., 2017. Deep and confident prediction for time series at Uber. In: International Conference on Data Mining Workshops. pp. 103–110.