Australasian Transport Research Forum 2022 Proceedings
28-30 September, Adelaide, Australia
Publication website: http://www.atrf.info

Threshold-free Anomaly Detection on Traffic Flow Data with Reinforcement Learning

Dan He[1], Boyu Ruan[2], Jiwon Kim[1], Hua Shi[1]
[1]{d.he, jiwon.kim, hua.shi}@uq.edu.au, The University of Queensland, Australia
[2]ruanboyu@sics.ac.cn, Shenzhen Institute of Computing Sciences, China

1. Introduction

When the traffic network is unexpectedly congested, there is usually a direct and far-reaching impact on the downstream links, which requires instant reaction from the operators for alleviating traffic pressure. The detection of the unexpected events or rare patterns on traffic data plays a significant role in intelligent traffic management, which provides crucial alerts of the potential events or incidents. Accurate anomaly detection on traffic data can bring prompt troubleshooting and is conducive to timely decision-making for the operators. Generally, an anomaly is an observation or a sequence of observations which deviate significantly from the general distribution of the given data, and the amount of deviation is regarded as the probability of being an anomaly. In this paper, we study the anomaly detection for traffic flow data captured by loop detectors in Brisbane, Australia, where our flow data can be categorized as univariate time series.

In the literature, various methods have been proposed for anomaly detection in univariate time series. A recent survey (Braei and Wagner, 2020) summarizes a comprehensive set of methods, divided into three categories: statistics-based, classical machine learning, and deep learning methods. In particular, the statistics-based methods, such as auto-regressive integrated moving average (ARIMA) (Zhang et al., 2005), usually assume that the data could be generated by specific statistical models, where such a generative model is constructed based on the given data and new data are fit to the model to determine anomaly. Machine learning algorithms such as K-Means Clustering (Rebbapragada et al., 2009) and One-Class SVM (Eskin et al., 2002) try to detect the anomaly based on the proximity of data without assuming a specific generative model. More recently, deep learning techniques have been applied in anomaly detection, trying to improve the existing methods by capturing complex, nonlinear correlations in data. A general approach to these solutions (Buda et al., 2018; Hundman et al., 2018; Munir et al., 2019) is to predict the next value in a time series based on the data points at the previous time steps and determine an anomaly based on certain rules, e.g., whether the difference between the observed value and predicted value is greater than a pre-defined threshold value.

Challenges: However, none of the existing solutions can be directly applicable to our flow anomaly detection problem due to the following challenges. (1) Most of the existing approaches rely on location-specific threshold setting to determine whether some data points are anomalies, thereby requiring separate anomaly detectors for different roads. However, we aim to develop a single, unified anomaly detection model that can be generalised and applied to any road with different scale of flow volumes, flow patterns, and anomaly patterns. (2) Some of the existing solutions are supervised models, which require the ground-truth labels of anomaly for model training. Unfortunately, our data contain no label, and it is difficult to manually label the anomaly from the flow data as the size of the required training set is usually large. (3) We are interested in detecting a sequence of abnormal data points in a flow time series, rather than a single abnormal data point, where the length of the anomaly sequences is not known and varies from one to another. However, most of the existing sequence-based anomaly detection methods

require a predefined length of anomalies, which is used as an abnormal/normal data unit. Thus, we need a more flexible model that can detect flow anomaly sequences with different lengths.

Contributions: To address the above-mentioned research gaps, we introduce a Reinforcement Learning (RL) approach to train an intelligent AI agent that can automatically detect the anomaly sequences with varying lengths from traffic flow data, without relying on predefined rules or requiring ground-truth labels. Specifically, there are three major contributions of our work. (1) *Generalisation:* The detecting mechanism of our model is completely data-driven in that it takes as input the flow data with different characteristics and can output the detected results with different anomaly patterns and lengths. (2) *Threshold-free:* Generally, most of the existing anomaly detection models require a predefined or self-adjusted threshold to examine whether the data unit is an anomaly after obtaining the corresponding anomaly score. In contrast, our model is threshold-free, which makes the model more flexible and adaptable. (3) *Unsupervised reward learning:* Some RL-based models (Huang et al., 2018; Yu and Sun, 2020) for anomaly detection define the rewards based on the known anomaly information for flow data, which requires ground-truth labels. Alternatively, we propose a novel unsupervised reward learning method to automatically learn the reward based on the distribution of data. As a result, our model requires no label of anomaly for training, while achieving high performance with around 90% precision and 80% recall in our experimental study on synthetic data.

2. Problem Statement

In our work, we aim to detect anomaly patterns from a sequence of traffic flow data within a certain period on any link of the road network. We define the traffic flow sequence as a univariate time series, denoted by $X = \{f_{t_1}, f_{t_2}, \dots, f_{t_n}\}$ where $f_{t_i} (1 \leq i \leq n)$ represents a traffic flow or volume (vehicles per unit time) at time step t_i. Usually, each flow value is measured by aggregating raw observations over a fixed time interval. Generally, the most significant characteristic of flow data is the periodicity that the daily fluctuation of flow follows similar pattern. However, the fluctuation pattern and the amount of peak flow might be different on different links. We aim to develop an anomaly detector that can detect the anomaly in the flow sequence from any link which might reflect different fluctuation patterns, and the length of flow sequence can various. Given a sequence of traffic flow data $X = \{f_{t_1}, f_{t_2}, \dots, f_{t_n}\}$, the anomaly detector generates an output sequence $Y = \{y_{t_1}, y_{t_2}, \dots, y_{t_n}\}$, where $y_{t_i} \in \{0,1\}$ is a binary indicator with $y_{t_i} = 1$ indicating f_{t_i} is abnormal and $y_{t_i} = 0$ indicating f_{t_i} is normal.

3. Methodology

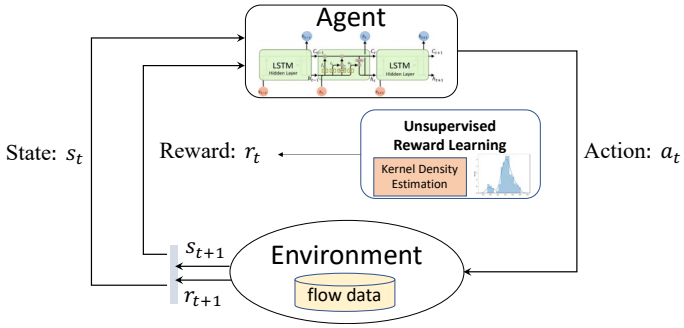
In general, a reinforcement learning (RL) problem can be represented as a Markov Decision Process (MDP), where there is an agent interacting with the environment. This process is formulated by a tuple with five elements: (A, S, R, P, γ). At each time step t, the agent takes an action $a_t \in \mathcal{A}$ according to the current state $s_t \in \mathcal{S}$ in the environment $\mathcal{E}$ based on a policy π. Then, the environment $\mathcal{E}$ produces a reward $r_t = \mathcal{R}(s_t, a_t)$ corresponding to such action, and obtain the next state s_{t+1} according to the state transition probability function $\mathcal{P}(s_{t+1}|s_t, a_t)$. The overall objective of the RL model is to learn an optimal policy π^* that can receive a maximum accumulative reward $\mathcal{R}_t = \sum_{i=0}^{\infty} \gamma^i r_{i+1}$, with the discount factor $\gamma \in (0,1]$. To achieve this, a well-known value-based learning algorithm, Q-learning algorithm (Watkins and Dayan, 1992), is introduced to learn the state and action value function $Q(s, a)$ for policy improvement, with the following update rule, where α is the learning rate.

$$Q(s, a) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

Recent years, Deep Q-Network (Mnih et al., 2015) is proposed to leverage the advantages of deep learning techniques in RL models. Consider our anomaly detection problem on traffic flow data. It can be regarded as an MDP that the determination of anomaly at the current time

step might influence the next decision due to the change of environment. Thus, we employ an RL model for our anomaly detection problem. Figure 1 shows the general framework of our model. Firstly, we introduce the formulation of state, action, and reward.

Figure 1: Framework of anomaly detection model



State. The environment of our RL model is derived from the traffic flow data. At each time step, the anomaly detector agent takes from the environment a piece of data and makes decision accordingly based on the correlations captured from this piece of data. We define the state at time t as the observed flows and the chosen actions over the past n time steps as follows:

$s_t = \{(f_{t-n+1}, \bar{f}_{t-n+1}, a_{t-n+1}), (f_{t-n+2}, \bar{f}_{t-n+2}, a_{t-n+2}), \dots, (f_t, \bar{f}_t, a_t)\}$,

where f_i is the current flow at time i, $\bar{f}_i$ is the historical mean of the flows for the same time-of-day period as i, and a_i is the chosen action at time i. The actions $a_{t-n+1}, \cdots, a_{t-1}$ are added to the state definition to incorporate the agent's past decisions into the current decision making. Since the action at time t has not yet been determined, a_t is set to -1.

Action. The action space is defined as $\mathcal{A} = \{0,1\}$. Given a state at time step t, $s_t = \{(f_{t-n+1}, \bar{f}_{t-n+1}, a_{t-n+1}), (f_{t-n+2}, \bar{f}_{t-n+2}, a_{t-n+2}), \dots, (f_t, \bar{f}_t, a_t)\}$, the anomaly detector agent chooses action $a_t = 1$ if f_t is regarded as an anomaly and $a_t = 0$ if f_t is regarded as normal data.

Reward. The design of reward function is significant for the agent to obtain a desirable policy. Basically, we give a positive reward when the agent detects the anomaly correctly and a negative reward for incorrect detection. However, as there is no ground-truth label available, it is challenging to judge whether the agent makes accurate decision or not. One way to judge is using the observation that f_t is more likely to be an anomaly if f_t significantly differs from its historical mean flow value ($\bar{f}_t$) for the same time-of-day or if f_t significantly differs from the flow value at the previous time step (f_{t-1}) in the current time series. To determine how significant a given difference is without relying on a manual, hard-coded threshold, we design a threshold-free unsupervised reward learning approach, which works as follows. 1) Firstly, given a flow value f_t at time step t, we retrieve all the current and historical flow values around the neighbouring time intervals, denoted by $N(f_t) = \{f_{t-i} | i = -2, -1, 0, 1, 2\}$. 2) Then we perform clustering on $N(f_t)$ based on kernel density estimation (KDE), which is a non-parametric way to estimate the probability density function of $N(f_t)$, to obtain k clusters. It is worth noting that k is not a predefined parameter and could be different according to different data distribution. 3) We denote the cluster containing f_t as C and calculate the ratio between the size of C and the average size of clusters, i.e., $\delta = \frac{|C|}{|N(f_t)|/k}$. 4) If $\delta \geq 1$, we have $r_t = \delta$ (or $r_t = -\delta$) with $a_t = 0$ (or $a_t = 1$). Alternatively, if $\delta < 1$, we have $r_t = 1/\delta$ (or $r_t = -1/\delta$) with $a_t = 1$ (or $a_t = 0$). The underlying intuition is that if f_t falls into a cluster with a small number of elements, it is more likely to be anomalous, and the amount of penalty regarding it as normal data is larger when the size of the corresponding cluster is smaller.

In terms of the model architecture, we employ the framework of Deep Q-Network and apply a Long-Short-Term-Memory (LSTM) as the agent for policy learning, as shown in Figure 1, which works as a binary classifier taking a state as input and output an action of either 1 or 0. It is an unsupervised model that the agent iteratively reads from the environment for the states, outputs the actions, and tunes the model based on the rewards. After several epochs, we terminate such interaction and the actions in the last epoch are regarded as the final output from this model. We observe from some case studies that the detected result from our model contains some false alarms, where some sequences with a single value or very few flow values are regarded as anomaly. This might be caused by the randomness of action taking in the RL model. To eliminate such false alarms, we conduct a post-processing to smooth the whole action sequence with a bidirectional sliding window, where the key idea is that the middle action is determined by the majority of actions within the sliding window.

4. Experimental Results

To evaluate the performance of our proposed approach, we conduct two sets of experiments, based on the real-world data and synthetic data, respectively. For both sets, we set the length of the state as 20 and the size of the sliding window to be 21. The hidden dimension of the LSTM model is 128 and we make use of 2 layers of LSTM.

4.1 Case study

The data used in the case study is collected from loop detectors in Brisbane, Australia in 2017, provided by STREAMS (n.d.). We select three month flow data from two different links, where each flow value is in 3-minute aggregation. Since flow patterns are usually different between weekday and weekend, here we only consider the flow data from weekdays. We plot a portion of detected results for each link, containing two weeks of flow data, as shown in Figure 2 and 3, where we mark the anomaly detected by our model with blue shaded areas. As we can see from these figures, our model can detect anomalies in two very different flow time series: the anomaly sequence in Case 1 shows a drop in flows over several hours which might be caused by temporary incidents like crashes; the one in Case 2 shows very low flows over two consecutive days which might be due to some special events or roadwork. Despite few points with false alarms (Case 1) and missed detections (Case 2), our model demonstrates the ability to detect the whole sequence of an abnormal period in a fully unsupervised manner.

Figure 2: Anomaly detection result for Case 1

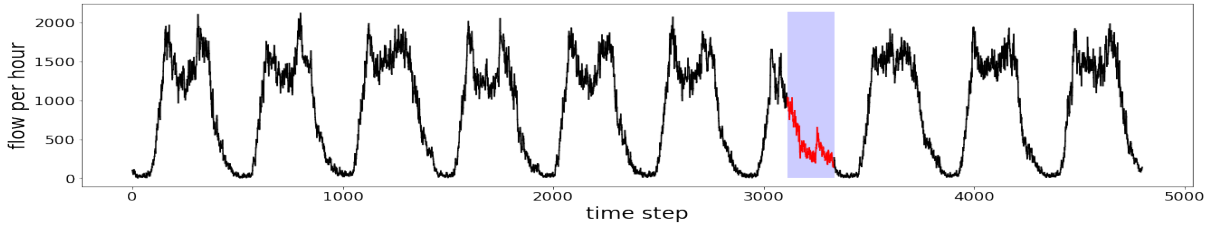
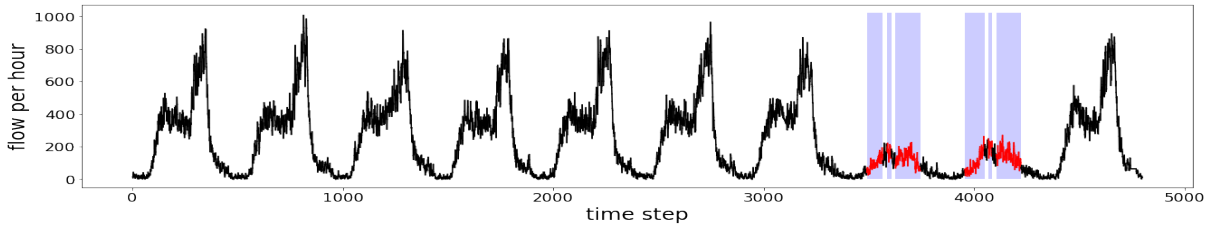


Figure 3: Anomaly detection result for Case 2



4.2 Performance Evaluation

Next, we evaluate the performance of our model on the synthetic data. Since there is no ground-truth in our real-world data set, it is difficult to evaluate the performance of our proposed model.

Thus we generate a comprehensive set of synthetic data that contains labels of anomalies. Firstly, we simulate the normal flow data from our real world data based on Gaussian distribution. Figure 4 shows an example of the synthetic data with no anomaly. Then, we randomly add some anomaly sequences into the normal data according to different parameter settings, as shown in Table 1.

Figure 4: Example of synthetic data

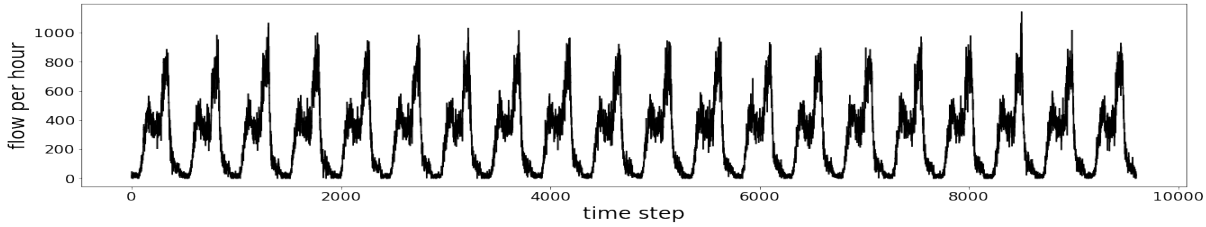


Table 1: Parameter settings for synthetic data generation

Parameters	Values	Unit	Default
Dataset size	1, 3, 6, 9, 12	month	3
Anomaly density	10%, 20%, 30%, 40%, 50%	percentage	20%
Sequence length	[50, 100], [100, 150], [150, 200], [200, 250], [250, 300]	time step	[50, 300]
Error rate	[40, 50], [50, 60], [60, 70], [70, 80], [80, 90]	percentage	[40, 90]

Specifically, we consider 4 different parameters: *size of dataset, anomaly density, length of anomaly sequence, and error rate*. The 'anomaly density' indicates the percentage of the days containing anomaly sequence. The 'error rate' defines the extent to which an abnormal flow deviates from the normal value, formulated by $(f - \tilde{f})/\tilde{f}$, where $\tilde{f}$ is the anomalous flow value. We use the default settings for the other three parameters when varying the values of one studied parameter. The synthetic data is generated as follows. We add one or more anomaly sub-sequences to the original traffic time series by changing the flow values over certain time windows. For each anomaly sub-sequence generation, we select the start and end times of the anomaly time window based on the 'sequence length' parameter and assign flow values based on 'error rate' parameter. Different anomaly sub-sequences within the same traffic time series might have different lengths as we randomly sample the 'sequence length' according to a given range shown in Table 1. Similarly, for each anomaly sub-sequence, the error rate to assign the flow value at each time step is picked from the defined range randomly. We simulate a group of data containing 5 sets of synthetic traffic time series based on each parameter setting and measure the average detected performance according to the following metrics (TP: True Positive, FP: False Positive, FN: False Negative).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad Recall = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Table 2: Performance results for synthetic data set with different parameter settings

Data Set	Precision	Recall	F1 Score	Data Set	Precision	Recall	F1 Score
density (10%)	93.27%	82.69%	87.60%	**length (50-100)**	74.36%	74.59%	74.45%
density (20%)	92.38%	81.75%	86.73%	**length (100-150)**	85.74%	82.93%	84.25%
density (30%)	92.31%	81.21%	86.39%	**length (150-200)**	92.11%	84.09%	87.31%
density (40%)	92.79%	80.16%	85.52%	**length (200-250)**	96.79%	83.84%	89.82%
density (50%)	91.38%	83.23%	87.10%	**length (250-300)**	99.87%	83.66%	91.04%
size (1m)	83.72%	80.58%	82.96%	**error (40-50%)**	89.77%	58.15%	69.67%
size (3m)	91.43%	84.19%	87.63%	**error (50-60%)**	92.22%	82.03%	86.81%
size (6m)	91.43%	84.19%	87.63%	**error (60-70%)**	92.66%	82.29%	87.14%
size (9m)	92.57%	82.09%	87.01%	**error (70-80%)**	93.39%	82.64%	87.65%
size (12m)	92.40%	83.53%	87.73%	**error (80-90%)**	91.29%	85.51%	86.77%

Table 2 shows the experimental results of our model. In general, with different parameter settings, the precision is mostly over 90%, recall is around 80%, and the F1 score is more than 85%. More specifically, varying the density of anomaly does not bring too much difference in performance, while the length of anomaly sequence has higher impact on the detected results with a longer sequence leading to a better performance, which shows that our model is more adaptable to data with longer anomaly sequences. In terms of the error rate, by intuition, it would be easier for a model to detect the anomaly with higher error rate, which is in line with the trend of the recall shown in Table 2. The recall is poor when the error rate is in the range between 40% to 50%, meaning that the model regards some of the anomalies as normal data. As for the size of data, it can be seen from the results that with more input data, the performance is slightly better. But the difference of performance for 3-month data is mild against the one for 12-month data, while the computational cost is much larger for the latter since the running time grows linearly with the size of data. Thus 3-month data is a good fit of input size for our model.

5. Conclusion

We introduce a reinforcement learning model to automatically detector the anomaly on traffic flow data. The experimental results demonstrate the effectiveness and practicability of our model in anomaly detection. We will further conduct a comprehensive experimental study and compare our model with the existing solutions to show the superiority of our approach.

Acknowledgement

The authors gratefully acknowledge the support of Australian Research Council (ARC), Queensland Department of Transport and Main Roads, and Transmax Pty Ltd under the ARC Linkage Project on Real-time Analytics for Traffic Management (LP180100018).

References

Braei, M., Wagner, S., 2020. Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art. ArXiv200400433 Cs Stat.

Buda, T.S., Caglayan, B., Assem, H., 2018. DeepAD: A Generic Framework Based on Deep Learning for Time Series Anomaly Detection, in: Phung, D., Tseng, V.S., Webb, G.I., Ho, B., Ganji, M., Rashidi, L. (Eds.), Advances in Knowledge Discovery and Data Mining, Lecture Notes in Computer Science. Springer International Publishing, Cham, pp. 577–588. https://doi.org/10.1007/978-3-319-93034-3_46

Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S., 2002. A Geometric Framework for Unsupervised Anomaly Detection, in: Barbará, D., Jajodia, S. (Eds.), Applications of Data Mining in Computer Security, Advances in Information Security. Springer US, Boston, MA, pp. 77–101. https://doi.org/10.1007/978-1-4615-0953-0_4

Huang, C., Wu, Y., Zuo, Y., Pei, K., Min, G., 2018. Towards Experienced Anomaly Detector Through Reinforcement Learning. Proc. AAAI Conf. Artif. Intell. 32.

Hundman, K., Constantinou, V., Laporte, C., Colwell, I., Soderstrom, T., 2018. Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18. Association for Computing Machinery, New York, NY, USA, pp. 387–395. https://doi.org/10.1145/3219819.3219845

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D., 2015. Human-level control through deep reinforcement learning. Nature 518, 529–533. https://doi.org/10.1038/nature14236

Munir, M., Siddiqui, S.A., Dengel, A., Ahmed, S., 2019. DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series. IEEE Access 7, 1991–2005. https://doi.org/10.1109/ACCESS.2018.2886457

Rebbapragada, U., Protopapas, P., Brodley, C.E., Alcock, C., 2009. Finding anomalous periodic time series. Mach. Learn. 74, 281–313. https://doi.org/10.1007/s10994-008-5093-3

Watkins, C.J.C.H., Dayan, P., 1992. Q-learning. Mach. Learn. 8, 279–292. https://doi.org/10.1007/BF00992698

Yu, M., Sun, S., 2020. Policy-based reinforcement learning for time series anomaly detection. Eng. Appl. Artif. Intell. 95, 103919. https://doi.org/10.1016/j.engappai.2020.103919

Zhang, Y., Ge, Z., Greenberg, A., Roughan, M., 2005. Network Anomography. pp. 317–330. https://doi.org/10.1145/1330107.1330146