RobustTAD: Robust Time Series Anomaly Detection via Decomposition and Convolutional Neural Networks

Jingkun Gao, Xiaomin Song, Qingsong Wen, Pichao Wang, Liang Sun, Huan Xu
{jingkun.g,xiaomin.song,qingsong.wen,pichao.wang,liang.sun,huan.xu}@alibaba-inc.com
Machine Intelligence Technology, DAMO Academy, Alibaba Group

ABSTRACT

The monitoring and management of numerous and diverse time series data in many real-world applications calls for an effective and scalable time series anomaly detection service. In this paper, we propose RobustTAD, a Robust Time series Anomaly Detection framework by integrating time series decomposition and convolutional neural network for time series data. The robust seasonal-trend decomposition can effectively handle complicated patterns in time series, and meanwhile significantly simplifies the architecture of the neural network. In order to effectively capture the multi-scale information from time series for the purpose of anomaly detection, we apply an encoder-decoder architecture with skip connections. Due to the limited labeled data in time series anomaly detection, we systematically investigate data augmentation methods in both time and frequency domains for the decomposed components. We also introduce label-based weight and value-based weight in the loss function by utilizing the underlying unbalanced nature of the problem where anomaly samples are rare. Compared with the widely used forecasting-based anomaly detection algorithms, decomposition-based algorithms, traditional statistical algorithms, as well as recent neural network based algorithms, our proposed RobustTAD algorithm performs significantly better on public benchmark datasets.

CCS CONCEPTS

• **Mathematics of computing → Time series analysis**; • **Computing methodologies → Anomaly detection**.

KEYWORDS

time series, anomaly detection, seasonal-trend decomposition, robustness

ACM Reference Format:
Jingkun Gao, Xiaomin Song, Qingsong Wen, Pichao Wang, Liang Sun, Huan Xu. 2020. RobustTAD: Robust Time Series Anomaly Detection via Decomposition and Convolutional Neural Networks. In *MileTS '20: 6th KDD Workshop on Mining and Learning from Time Series, August 24th, 2020, San Diego, California, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/1122445.1122456

MileTS '20, August 24th, 2020, San Diego, California, USA

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

As the rapid increase of time series data due to the developments of Internet of Things (IoT) and many other connected data sources, real-time time series anomaly detection is a required capability in many real-world applications, such as predictive maintenance, intrusion detection, fraud prevention, cloud platform monitoring and management, business data monitoring, etc. For example, in the monitoring of a data center, usually we need to monitor metrics at different levels, ranging from physical machine, docker, to service hosted on each docker. Another example is the tracking of sales volume as some unusual high amounts may be caused by some bogus transactions. In this paper we focus on the anomalies in univariate time series. Specifically, we define an anomaly in a time series as an observation which is significantly different from previous normal observations, which we call its "context". Formally, given a sequence $\cdots, x_{t-w}, x_{t-w+1}, \cdots, x_{t-1}, x_t$, our goal is to determine whether the newest observation x_t is an anomaly by considering the current and previous observations up to a window size w.

Time series anomaly detection has been researched for over a long time [2, 6]. However, challenges still remain in order to achieve the optimal performance. Firstly, the time series data in real-world scenarios is quite diverse. It does not only contain the temporal dependency, but may also exhibit more complicated patterns, such as abrupt change of trend, seasonality shift and fluctuation. Secondly, the labeling of anomalies is challenging and thus very limited labeled data is available. In time series anomaly labeling, we need to compare each sample with its previous context, thus it is quite time-consuming to label all samples in a time series manually. In addition, the definitions of anomaly may vary in different scenarios. Thirdly, almost all time series anomaly detection systems are required to respond in real time. Therefore, the anomaly detection algorithm should be efficient and can handle a large number of time series in parallel with low latency.

Recent years witness the advances in deep learning [5]. Compared with the wide success of deep learning in computer vision and natural language processing, it only has limited applications in time series. When processing time series data, a key challenge is how to consider temporal dependency and extract features from the original time series effectively. In particular, in time series anomaly detection the multi-scale information is very important as it effectively helps to define the context so anomaly can be compared and determined. When more complex patterns in seasonality and trend are present in time series, it is even more challenging to build a deep learning model. In addition, deep learning models generally require a lot of labeled data for training. Thus, data augmentation is a crucial step in model training given the limited labeled data. Unlike data augmentation for image [17] and speech [11], few work has been done on data augmentation for time series [23]. Specifically,

as recurrent neural network (RNN) and long short-term memory (LSTM) network are ideal tools to model temporal dependency, several works [10, 14] have been proposed for time series anomaly detection. However, a challenge for these networks is how to deal with the seasonality especially long seasonality in a general manner. For example, [12] proposes to create a shortcut connection to learn seasonal pattern directly by assuming that there is only single seasonality with known seasonal length, however, its generality is harmed by time series data with multiple seasonalities without prior seasonal lengths. Meanwhile, convolutional neural networks (CNN) only have limited applications in time series classification , clustering [27], and anomaly detection [25]. How to encode the temporal dependency and complicated time series patterns in CNN still remains an open problem.

In this paper, we propose RobustTAD, a **Robust T**ime series **A**nomaly **D**etection framework integrating seasonal-trend decomposition with convolutional neural network. In the seasonal-trend decomposition, we first apply RobustPeriod [22] algorithm to detect if the time series is periodic and estimate its period lengths for periodic time series, and then apply RobustSTL [21, 24] to decompose the input time series into different components. Next, we build a convolutional neural network based on the decomposed components to detect anomaly. Our network is based on U-Net [16], an encoder-decoder architecture with skip connection to extract multi-scale features from time series. Performing robust seasonal-trend decomposition for time series before training the network has several benefits: 1) as the seasonal component is explicitly extracted, there is no need to build complex structure in the neural networks to deal with it, which significantly simplifies the network structure and meanwhile improves model performance; 2) it leads to a more general anomaly detection framework for time series with different characteristics, such as different seasonal lengths, different types of trend, etc. As limited labeled data is available in typical time series anomaly detection tasks, data augmentation is crucial to the successful training of the neural network. In this paper, we investigate different data augmentation methods for time series, in both time domain and frequency domain for decomposed components. By utilizing the unbalanced nature of time series anomaly detection problem, we adjusted the loss function by introducing the label-based weight and value-based weight. To the best of our knowledge, our work is the first one which integrates the robust seasonal-trend decomposition with deep neural networks for time series anomaly detection. Our experiments on public benchmark datasets show that our integrated framework RobustTAD outperforms other popular algorithms, including forecasting-based methods, decomposition-based methods, and methods based on deep learning but without time series decomposition.

2 METHODOLOGIES

In this section, we introduce the proposed time series anomaly detection framework taking advantage of both seasonal-trend decomposition and deep convolutional neural network.

2.1 Decomposition

Usually the time series data contains different components. In this paper, we assume a time series can be decomposed as the sum of trend, seasonality (if the time series is periodic), and remainder components:

$$x_t = \tau_t + s_t + r_t, t = 1, 2, \cdots, N$$

where x_t denotes the original signal at time t, τ_t denotes the trend, s_t denotes the seasonality, and r_t is the remainder component. In this paper, we apply RobustPeriod [22] to detect periodicity and the corresponding period length. If the time series is periodic, we next apply RobustSTL [21, 24] to decompose the time series; otherwise we apply RobustTrend [20] to perform the decomposition. After the decomposition, we feed the remainder component r_t as the input to the neural network.

2.2 Encoder-Decoder Network

2.2.1 Architecture. Time series anomaly detection is a point-wise dense prediction problem. In other words, for a time series $\mathbf{x} = \{x_t\}_{t=1,2,\cdots,N}$, the goal is to produce a sequence of the same length $\mathbf{y} = \{y_t\}_{t=1,2,\cdots,N}$ where $y_i \in \{0, 1\}$ denotes whether x_i is an anomaly or not. Notice that this dense classification problem shares many similarities with the image segmentation problem in computer vision where a pixel-wise inference is required. As reviewed in [18], many state-of-the-art image segmentation approaches have adopted an encoder-decoder network structure, which helps to extract hidden features from the input. Meanwhile, to encode the complex patterns of the time series, it is necessary to consider both the local and global information (or multi-scale feature), which leads to the use of a skip connection that preserves the local information from the encoder layer by concatenating to the decoder input.

As a result, we adopt an encoder-decoder network architecture with skip connections, as known as U-Net structure [16] shown in Figure 1. The network is trained by feeding multiple samples of time series segment along with the corresponding labels. It is worth mentioning that utilizing U-Net structure for time series anomaly detection has shown promising results in [25]. However, in this paper, we would like to emphasize that applying a U-Net structure directly without modification would still lead to a sub-optimal performance. Besides the robust seasonal-trend decomposition, many adjustments are required to achieve the optimal results, as described in the following subsections.

2.2.2 Weight Adjusted Loss. We cast the time series anomaly detection problem as a supervised classification problem, which is highly unbalanced. The unbalanced nature lies in the fact that typically only a few anomalies exist in a long time series. Applying the most commonly used pixel-wise cross entropy loss from U-Net would lead to unsatisfying performance as equal weights are assigned to normal samples and anomalous samples, yet the few anomalous samples that contribute most to training the model have been overshadowed by the vast majority of normal samples. As a result, we assign more weights to anomalous samples, and refer this as **label-based weight**. On the other hand, we notice that a point in the time series with value different from its neighbors is more likely to be anomalous. By considering the difference between each point and its neighbours, we define the so-called **value-based weight**. More specifically, for a time series $\mathbf{x} = \{x_t\}$, with ground truth labels $\mathbf{y} = \{y_t\}$, we denote the predictions from the network as $\hat{\mathbf{y}} = \{\hat{y}_t\}$,

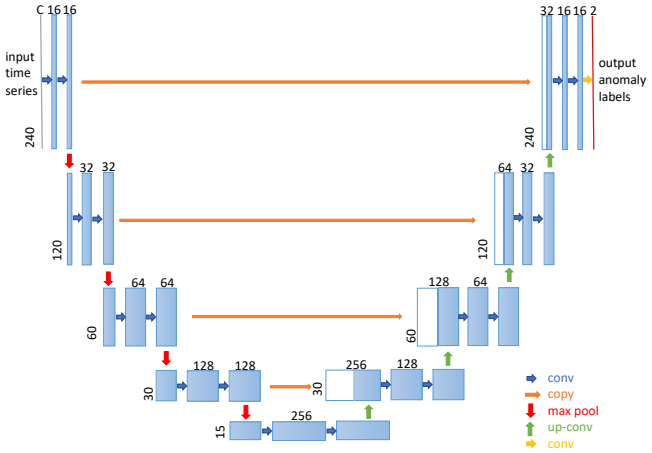


Figure 1: Architecture of the Encoder-Decoder network for time series anomaly detection.

which are essentially the probabilities of x_t being anomalies after the soft-max layer. The original cross-entropy loss is defined as

$$Loss(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_t y_t \log(\hat{y}_t) + (1 - y_t) \log(1 - \hat{y}_t).$$

By applying the label-based and value-based weights, we define the new weight adjusted loss as

$$WALoss(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_t w_t(\beta \cdot y_t \log(\hat{y}_t) + (1 - y_t) \log(1 - \hat{y}_t)),$$

where β represents the **label-based weight** which typically has a value greater than 1, and $w_t = \frac{1}{Z_t} \exp(\frac{\sum_{j=1}^{H}(x_t - x_{t-j})^2}{2\sigma_{t,H}^2})$ is the **value-based weight**, H is the size of the neighborhood we considered in value-based weight, $\sigma_{t,H}^2$ is the variance of a window size H before point t, Z_t is a normalization term of weights over all points of the current window.

The adjustment of weight is essential to ensure the network is able to learn the patterns from anomalies and converge faster given the unbalanced nature of the problem.

2.3 Data Augmentation

Data augmentation [5], which generates artificial data for training, is an effective way to improve model performance in deep learning, especially when the amount of the training data is limited. Currently, very few work has been done on data augmentation for time series data [4, 19, 23]. Note that the labeled data in time series anomaly detection is generally very limited. In this subsection, we present several practical and effective data augmentation techniques specifically designed for time series after decomposition, in both time domain and frequency domain.

2.3.1 Time Domain. We summarize several effective transforms for time series anomaly detection in the time domain, including flipping, downsampling, cropping, and label expansion. Assume we have an input time series $x_1, \cdots, x_N$:

Flipping. We generate a new sequence $x_1^{'}, \cdots, x_N^{'}$ where $x_t^{'} = -x_t$ with the same anomaly labels. It can be used when we care anomalies in both directions. In the scenarios where we are interested in only one direction, this transform should not be applied.

Downsampling. We downsample the original time series of length N with a specific downsampling rate k, to get a shorter time series with length $\lfloor N/k \rfloor$. The label series are also dow sampled, or diluted, in the same rate k as values series.

Cropping. Similar to random crop in computer vision, we crop samples with replacement from the original time series of length N to get shorter time series with length N'. The label series are also cropped, with the same time stamp as values series.

Label Expansion. In time series anomaly detection, the anomalies generally occur sequentially. As a result, a data point which is close to a labeled anomaly in terms of both time distance and value distance is very likely to be an anomaly. We select those data points and label as anomalies in our training dataset.

2.3.2 Frequency Domain. To further increase the labeled data and utilize the periodical properties of time series data, we explore the data augmentation methods in the frequency domain. Specifically, we have developed several different policies, i.e., magnitude and phase augmentation, to generate more artificial labeled data.

To perform data augmentation in the frequency domain, we first transform the data into the frequency domain by applying the fast Fourier transform, where we can get real and imaginary parts at corresponding frequency. In the frequency domain, our intuitive idea is to make perturbations in magnitude and phase in selected segments in the frequency domain, after which the perturbed signal is mapped back to the original time domain.

In magnitude augmentation, we make perturbations in the magnitude spectrum. Specifically, we replace the values of all points in the selected segment with v, where v has Gaussian distribution as $v \sim N(\mu_A, q_A\bar{\delta}_A^2)$, where μ_A can be set as zero or $\bar{\mu}_A$ based on configuration, and $\bar{\mu_A}, \bar{\delta}_A^2$ is the sample mean and variance of the time series in the segment, respectively, and q_A is adopted to control the degree of perturbation.

Similarly, we can make perturbation in the phase spectrum in phase augmentation. Specifically, the values of all points in the selected segment are increased by a small perturbation θ, which is sampled from Gaussian distribution as $\theta \sim N(0, \delta_\theta^2)$ where δ_θ^2 represents the variance.

As illustration, the proposed frequency-domain time series augmentation methods on a sample data is plotted in Figure 2.

2.4 Online Inference for Streaming Time Series

In the online inference stage, we need to perform both decomposition and the inference of the deep network.

For the decomposition algorithms RobustSTL, an online version can be adopted to process the streaming data efficiently. We present some details of the trend extraction in the online setting, the most computationally expensive step here. Firstly, we assume that the change of trend is slow most of the time. Hence, we do not need to re-estimate trend every time a new point streams. We pick a small step size q and only solve the optimization problem every q data points arrive. When the trend has an abrupt change, there will be a delay at most q points to estimate it in detecting such an anomaly,

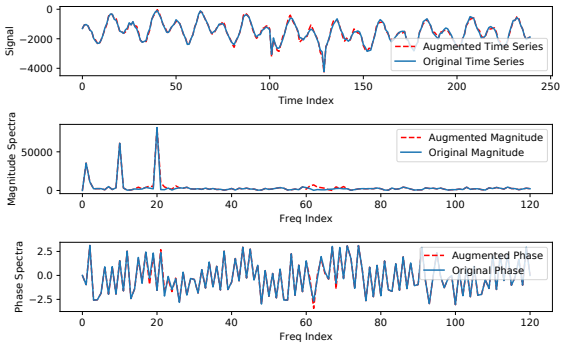


Figure 2: Plots of time series using magnitude augmentation and phase augmentation.

which is typically not a concern for most users especially when q is quite small (e.g., $q = 5$). Secondly, when we need to solve the optimization problem to estimate trend difference $\nabla\tau$, we can use the time series with the most recent $\hat{w}$ points instead of the whole time series. Additionally, we use the "warm start" technique where the solution at time $t-1$ is treated as the initial solution at time t, leading to a faster convergence speed.

Although training the network with a large amount of augmented samples is time-consuming, the online inference of the neural network is quite efficient as only matrix multiplications are needed. We are aware that the new patterns of anomaly might show up as time series stream and keep on coming. Hence, we update the trained model regularly by re-training with the new data to encode the latest information.

3 EMPIRICAL RESULTS

3.1 Datasets

We use 367 time series from the public Yahoo benchmark datasets [13]. These time series are hourly sampled data with a duration of two months, which are collected from the real production traffic to some of the Yahoo website properties and represent the complex patterns such as seasonality and trends in real-world time series data quite well. Yahoo data set has a good coverage of different varieties of anomalies in time series, such as seasonality, level change, variance change and their combinations.

3.2 Evaluation Metrics

We compute the following evaluation metrics:

- Precision: $\frac{TP}{TP+FP}$, represents the ratio of TP in all the data points which are predicted/detected as anomaly.
- Recall: $\frac{TP}{TP+FN}$, represents the ratio of TP in the all the data points which are truly anomaly.
- F1 score: $2 \times \frac{Precision \cdot Recall}{Precision + Recall}$, an overall measure of accuracy that combines both precision and recall. Perfect accuracy is achieved when F1 is 1, and 0 otherwise.

In addition to the metrics mentioned above, we also report a relaxed version of F1 score [3], which could serve better on the test datasets where pattern anomalies exist. Pattern anomalies happen when a segment of consecutive points are labeled as anomalies. Instead of requiring an exactly point-wise match between a true anomaly and a detected anomaly for a point to be counted towards TP, it is reasonable to give some leeway by allowing a lag up to a window size m. This corresponds to the scenario where the detected anomaly points might be earlier or delayed by a few points compared with the labeled anomaly points. We would still count these detected points as TP. We choose $m = 3$ to allow a mismatch up to 3 adjacent points. During the experiment, TP, FP, FN will be aggregated over multiple time series for one dataset and be used to produce a single micro-precision/recall/F1 score.

3.3 Experimental Setup

The following state-of-the-art methods are used for comparison:

- *ARIMA*[1]: ARIMA models [1] are well known for its versatility to forecast time series. We label a prediction point as anomaly if its true value lies outside the prediction intervals;
- *SHESD*[2]: SHESD [8] is an extension of both ESD and S-ESD. It decomposes and extracts the seasonal component, and then applies robust statistics including median and median absolute deviation (MAD) to calculate the anomaly score;
- *Donut*[3]: Donut [26] is an unsupervised, variational auto-encoder (VAE) based algorithm for detecting anomalies in seasonal KPIs in DevOps domain. It has competitive experimental results and a solid theoretical explanation.

In additional to these methods, we also investigate the performance of our framework in different settings:

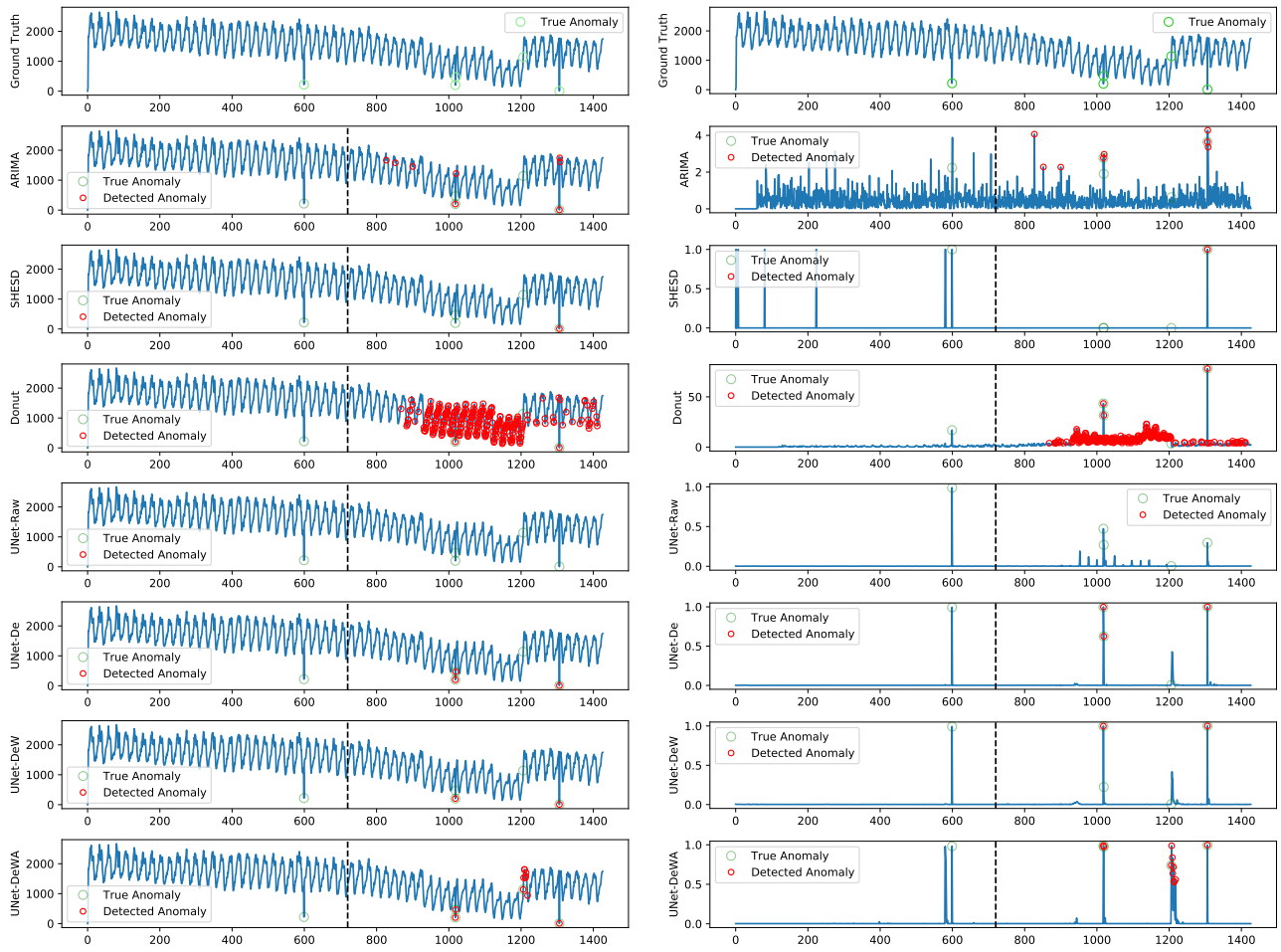
- *U-Net-Raw*: Apply U-Net on raw time series data directly.
- *U-Net-De*: Apply U-Net on decomposed remainder.
- *U-Net-DeW*: Adopt weight adjusted loss for U-Net-De.
- *U-Net-DeWA*: Adopt data augmentation for U-Net-DeW.

Throughout our experiments, we split each time series into left half and right half, corresponding to the train part and test part, respectively. We feed the training parts from all samples to train the network/model together. Then we predict and test the results using all the test parts. During the training and testing, we use the online mode, i.e., the sliding window mode, with window size at 240 points. In each sliding window, the model predicts the label of last point or right most point, then progresses one point at a time to the right. To make a relative fair comparison, for unsupervised algorithms, labels in the training part are used to tune hyper parameters, such as probability threshold, which is chosen by matching predicting scores with the true labels to maximize F1 within training data.

3.4 Results

Table 1 summarizes the performance of different methods, including both F1 score and the relaxed version. SHESD achieves acceptable result. Donut, as it was designed for DevOps data, has good recall but low precision. For the architecture based on U-Net, we can see that a naive version of the U-Net without any adjustments only achieves a 0.403 F1 score and 0.533 relaxed F1 score. However, once we include the decomposition, we can see a significant 0.22 increase in F1. By adopting the aforementioned loss adjustment, as well as

[1]https://www.rdocumentation.org/packages/forecast/versions/8.5/topics/auto.arima
[2]https://github.com/twitter/AnomalyDetection
[3]https://github.com/haowen-xu/donut



(a) Plot of raw values with detected anomaly.

(b) Plot of anomaly scores with detected anomaly.

Figure 3: Performance of each method on a specific time series. The values before the vertical black dashed line represent the training data, and those after represent the testing data.

data augmentation, we are able to further increase F1 up to 0.693, and relaxed F1 up to 0.812, which is significantly better than the other state-of-the-art methods.

Table 1: Performance comparison of different methods. Acronyms in U-Net section: De - decomposition, W - weight adjustment, A - augmentation

	Precision	Recall	F1 Score	Relax F1 Score
ARIMA	0.513	0.144	0.225	0.533
SHESD	0.501	0.488	0.494	0.557
Donut	0.015	0.829	0.029	0.030
U-Net-Raw	0.473	0.351	0.403	0.533
U-Net-De	0.651	0.594	0.621	0.710
U-Net-DeW	0.793	0.569	0.662	0.795
U-Net-DeWA	0.859	0.581	**0.693**	**0.812**

To further compare how these methods perform, we demonstrate their performance on a real-world dataset from Yahoo A1 in Figure 3, where it includes seasonality combined with level changes and spikes. The original U-Net-Raw fails to detect anomalies in this time series. However, as we include decomposition, loss adjustment, and data augmentation in our framework, the network has demonstrated significant learning capabilities, being able to identify those anomalies correctly in Figure 3a. We further examined the anomaly scores generated by each method in Figure 3b. As we can see, for ARIMA and Donut, an appropriate threshold needs to be picked in order to identify anomalies correctly. Note that the U-Net-Raw generates many false negatives. As we apply the adjustments we discussed in Section 2, more anomalies are being identified correctly. It is worth mentioning that for the anomalies showing at index position 1200 where there is a level change combined with seasonality, U-Net-De and U-Net-DeW all fail to identify correctly despite the probabilities of being anomaly is very close to 0.5. On

the other hand, as we incorporate data augmentation from both time domain and frequency domain, such anomalies could be easily identified with U-Net-DeWA.

Additionally, we report the time cost of the online inference for RobustTAD. For each streaming sample with a window size of 240, it takes $6ms$ for the online decomposition and $1ms$ for the network inference on a MacBook Pro with a 2.3GHz Intel i5 CPU and 8GB RAM. When we test on a NVIDIA Tesla V100 GPU card, the network inference time can be further reduced to $3\mu s$ per sample. The efficient inference of RobustTAD sheds light into the feasibility of handling a large number of streaming time series with low latency.

4 DISCUSSION AND CONCLUSION

The empirical comparison on the Yahoo datasets clearly demonstrates the promising performance of decomposition and encoder-decoder structure for time series anomaly detection. In particular, robust seasonal-trend decomposition significantly improves deep model's performance; while U-Net encoder-decoder structure can effectively extract multi-scale features. Intuitively, in anomaly detection we need to compare each observation with its context to determine its label. The multi-scaled features learned actually describe the context from different levels quite well, thus leading to good anomaly detection performance. Also note that the general framework integrating both time series decomposition and deep network with some modifications can be applicable for many other time series tasks such as forecasting.

In summary, in this paper we propose RobustTAD, a Robust Time series Anomaly Detection framework based on time series decomposition and convolutional neural network. Combined with both of their advantages, this algorithm does not only achieve high performance and high efficiency, but also can handle complicated patterns and lack of sufficient labels, which makes it practical and effective approach to serve cloud and IoT monitoring.

More future work can be done to extend RoubstTAD, including: 1) explore more network architectures for better learning of multi-scale features. For example, we can impose a loss function on each decoder layer [9] for a better representation learning; 2) multi-channel study by feeding three decomposed components directly to the network (in other words, $C = 3$ in Figure 1); 3) a joint learning framework to learn the time series decomposition and anomaly labels at the same time. Similar work has been done on object detection and image segmentation, such as Faster R-CNN [15] and Mask R-CNN [7]. It is also worth exploring more effective data augmentation methods for different machine learning tasks involving time series.

REFERENCES

[1] George E.P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, and Greta M. Ljung. 2015. *Time Series Analysis: Forecasting and Control.* John Wiley & Sons.

[2] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM Comput. Surv* 41 (2009), 1–72.

[3] Dhruv Choudhary. 2017. On the runtime-efficacy trade-off of anomaly detection techniques for real-time streaming data. (2017). arXiv:arXiv:1710.04735v1

[4] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2018. Data augmentation using synthetic data for time series classification with deep residual networks. *CoRR* abs/1808.02455 (2018). arXiv:1808.02455 http://arxiv.org/abs/1808.02455

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning.* MIT Press, Cambridge, MA. http://www.deeplearningbook.org.

[6] Manish Gupta, Jing Gao, Charu C. Aggarwal, and Jiawei Han. 2014. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and Data Engineering* 26, 9 (2014), 2250–2267.

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision.* 2961–2969.

[8] Jordan Hochenbaum, Owen S. Vallis, and Arun Kejariwal. 2017. Automatic anomaly detection in the cloud via statistical learning. *CoRR* abs/1704.07706 (2017). arXiv:1704.07706 http://arxiv.org/abs/1704.07706

[9] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip HS Torr. 2017. Deeply Supervised Salient Object Detection with Short Connections. *CVPR* (2017). https://doi.org/10.1109/CVPR.2017.563 arXiv:1611.04849

[10] J. Kim, J. Kim, H. L. T. Thu, and H. Kim. 2016. Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection. In *2016 International Conference on Platform Technology and Service (PlatCon).* 1–5. https://doi.org/10.1109/PlatCon.2016.7456805

[11] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. Audio augmentation for speech recognition. In *Sixteenth Annual Conference of the International Speech Communication Association.*

[12] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18).* Association for Computing Machinery, New York, NY, USA, 95–104. https://doi.org/10.1145/3209978.3210006

[13] Nikolay Laptev, Saeed Amizadeh, and Ian Flint. 2015. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 1939–1947.

[14] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2015. Long Short Term Memory Networks for Anomaly Detection in Time Series. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.*

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *NIPS* (2017), 241–294. https://doi.org/10.2307/j.ctt1d98bxx.10

[16] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. (2015), 234–241.

[17] Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data* 6, 1 (2019), 60.

[18] Saed Asgari Taghanaki, Kumar Abhishek, Joseph Paul Cohen, Julien Cohen-Adad, and Ghassan Hamarneh. 2019. Deep Semantic Segmentation of Natural and Medical Images: A Review. *arXiv preprint arXiv:1910.07655* (2019).

[19] Terry T. Um, Franz M. J. Pfister, Daniel Pichler, Satoshi Endo, Muriel Lang, Sandra Hirche, Urban Fietzek, and Dana Kuliundefined. 2017. Data Augmentation of Wearable Sensor Data for Parkinson's Disease Monitoring Using Convolutional Neural Networks. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction (ICMI '17).* Association for Computing Machinery, New York, NY, USA, 216–220. https://doi.org/10.1145/3136755.3136817

[20] Qingsong Wen, Jingkun Gao, Xiaomin Song, Liang Sun, and Jian Tan. 2019. RobustTrend: A Huber Loss with a Combined First and Second Order Difference Regularization for Time Series Trend Filtering. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI).* 3856–3862.

[21] Qingsong Wen, Jingkun Gao, Xiaomin Song, Liang Sun, Huan Xu, and Shenghuo Zhu. 2019. RobustSTL: A robust seasonal-trend decomposition algorithm for long time series. In *AAAI.* 1501–1509.

[22] Qingsong Wen, Kai He, Liang Sun, Yingying Zhang, Min Ke, and Huan Xu. 2020. RobustPeriod: Time-Frequency Mining for Robust Multiple Periodicities Detection. *arXiv preprint arXiv:3056069* (2020).

[23] Qingsong Wen, Liang Sun, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. 2020. Time Series Data Augmentation for Deep Learning: A Survey. *arXiv preprint arXiv:2002.12478* (2020).

[24] Qingsong Wen, Zhe Zhang, Yan Li, and Liang Sun. 2020. Fast RobustSTL: Efficient and Robust Seasonal-TrendDecomposition for Time Series with Complex Patterns. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD'20).*

[25] Tailai Wen and Roy Keyes. 2019. Time Series Anomaly Detection Using Convolutional Neural Networks and Transfer Learning. (2019). arXiv:1905.13628 http://arxiv.org/abs/1905.13628

[26] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. 2018. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference.* 187–196.

[27] Jian Bo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. 2015. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15).* AAAI Press, 3995–4001.