2020 IEEE International Conference on Data Mining (ICDM)

Multivariate Time-series Anomaly Detection via Graph Attention Network

Hang Zhao*[§], Yujing Wang*[†][§], Juanyong Duan*, Congrui Huang*, Defu Cao[†]
Yunhai Tong[†], Bixiong Xu*, Jing Bai*, Jie Tong*, Qi Zhang*
*Microsoft, [†]Key Laboratory of Machine Perception, MOE, School of EECS, Peking University
{hang.zhao, yujwang, juanyong.duan, conhua, bix, jbai, jietong, qizhang}@microsoft.com, {cdf, yhtong}@pku.edu.cn

***Abstract*—Anomaly detection on multivariate time-series is of great importance in both data mining research and industrial applications. Recent approaches have achieved significant progress in this topic, but there is remaining limitations. One major limitation is that they do not capture the relationships between different time-series explicitly, resulting in inevitable false alarms. In this paper, we propose a novel self-supervised framework for multivariate time-series anomaly detection to address this issue. Our framework considers each univariate time-series as an individual feature and includes two graph attention layers in parallel to learn the complex dependencies of multivariate time-series in both temporal and feature dimensions. In addition, our approach jointly optimizes a forecasting-based model and a reconstruction-based model, obtaining better time-series representations through a combination of single-timestamp prediction and reconstruction of the entire time-series. We demonstrate the efficacy of our model through extensive experiments. The proposed method outperforms other state-of-the-art models on three real-world datasets. Further analysis shows that our method has good interpretability and is useful for anomaly diagnosis.**

***Index Terms*—multivariate time-series, anomaly detection, graph attention network**

I. Introduction

Time-series anomaly detection is an important research topic in data mining and has a wide range of applications in industry. Efficient and accurate anomaly detection helps companies to monitor their key metrics continuously and alert for potential incidents on time [1]. In real applications, multiple time-series metrics are collected to reflect the health status of a system [2]. Univariate time-series anomaly detection algorithms are able to find anomalies for a single metric. However, it could be problematic in deciding whether the whole system is running normally. For example, sudden changes of a certain metric do not necessarily mean failures of the system. As shown in Figure 1, there are obvious boosts in the volumes of *TIMESERIES_RECEIVED* and *DATA_RECEIVED_ON_FLINK* in the green segment, but the system is still in a healthy state as these two features share consistent tendency. However, in the red segment, *GC* shows inconsistent pattern with other metrics, indicating a problem in garbage collection. Consequently, *it is essential to take the correlations between different time-series into consideration* in a multivariate time-series anomaly detection system.

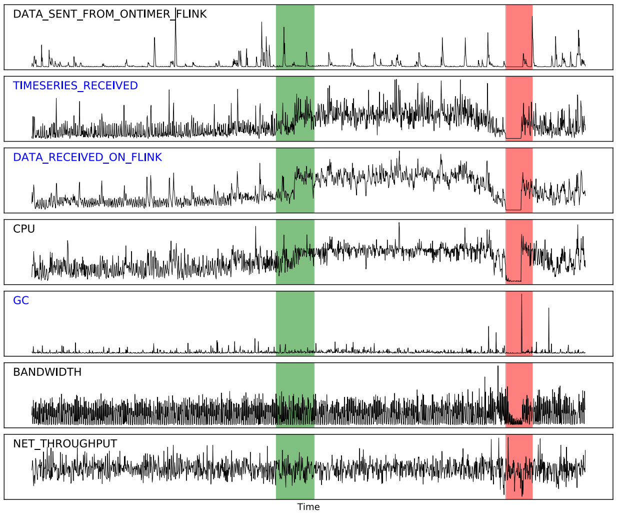


Fig. 1. An example of multivariate time-series input. Green indicates normal values and red indicates anomalies.

Previous studies on multivariate time-series anomaly detection have made fruitful progresses. For instance, Malhotra et. al [3] proposes a LSTM-based encoder-decoder network that models reconstruction probabilities of the "normal" time-series, and the reconstruction errors are utilized to detect anomalies from multiple sensors. Hundman et. al [2] leverages LSTM to detect anomalies in multivariate time-series metrics of spacecraft based on prediction errors. OmniAnomaly [4] proposes a stochastic recurrent neural network, which captures the normal patterns of multivariate time-series by modeling data distribution through stochastic latent variables. However, to the best of our knowledge, none of previous works in the literature have addressed the problem of capturing multivariate correlations explicitly. *We argue that there is still room for improvement if the relationships between different time-series can be modeled appropriately.*

Our Solution: In this paper, we propose a novel framework — MTAD-GAT (Multivariate Time-series Anomaly Detection via Graph Attention Network), to tackle the limitations of previous solutions. Our method considers each univariate time-series as an individual *feature* and tries to model the correlations between different features explicitly, while the temporal dependencies within each time-series are modeled at the same time. The key ingredients in our model are

[§]Equal contribution

2374-8486/20/$31.00 ©2020 IEEE
DOI 10.1109/ICDM50108.2020.00093

two graph attention layers [5], namely the *feature-oriented graph attention* layer and the *time-oriented graph attention* layer. The *feature-oriented graph attention* layer captures the causal relationships between multiple features, and the *time-oriented graph attention* layer underlines the dependencies along the temporal dimension. In addition, we jointly train a *forecasting-based model* and a *reconstruction-based model* for better representations of time-series data. The two models can be optimized simultaneously by a joint objective function.

The **contributions** of our paper are summarized as follows:

- We propose a novel framework to solve the multivariate time-series anomaly detection problem in a self-supervised manner. Our model shows superior performances on two public datasets and establishes state-of-the-art scores in the literature. It also achieves 9% improvement for overall F1 score on our production data, bringing big impact on user satisfaction.
- For the first time, we leverage two parallel graph attention (GAT) layers to learn the relationships between different time-series and timestamps dynamically. Especially, our model captures the correlations between different time-series successfully without any prior knowledge.
- We integrate the advantages of both forecasting-based and reconstruction-based models by introducing a joint optimization target. The forecasting-based model focuses on single-timestamp prediction, while the reconstruction-based model learns a latent representation of the entire time-series.
- Our network has good interpretability. We analyze the attention scores of multiple time-series learned by the graph attention layers, and the results correspond reasonably well to human intuition. We also show its capability of anomaly diagnosis.
- Code and dataset would be open source on GitHub[1] for reproduction, we hope it may provide inspiration for other works.

II. Related Work

There is a plenty of literature for time-series anomaly detection, which can be classified into two categories. The first category of approaches analyzes each individual time-series by applying univariate models [1], [6]–[8], while the second one models multiple time-series as a unified entity [2], [4], [9]–[13]. From another perspective, existing anomaly detection models can also be categorized into two paradigms, namely *forecasting-based* models [2], [14], [15] and *reconstruction-based* models [4], [11]–[13]. In this section, we summarize important works about time-series anomaly detection and discuss these two paradigms in detail.

A. Univariate Anomaly Detection

Classic methods typically utilize handcrafted features to model normal/anomaly event patterns [16], such as hypothesis testing [17], wavelet analysis [18], SVD [19] and ARIMA [20]. Recently, Netflix has released a scalable anomaly detection solution based on robust principal component analysis [6], which has been proven successful in some real scenarios. Twitter has also published a seasonality-considered anomaly detection method using the Seasonal Hybrid Extreme Study Deviation test (S-H-ESD) [7]. Recent advances in neural networks also lay a strong foundation for time-series anomaly detection [1], [8], [21]. DONUT [21] is an unsupervised anomaly detection method based on Variational Auto-Encoder (VAE), and SR-CNN [1] combines the benefits of Spectral Residual (SR) and convolutional neural network to achieve state-of-the-art performance on univariate time-series anomaly detection.

B. Multivariate Anomaly Detection

1) Forecasting-based Models: A forecasting-based model detects anomalies based on prediction errors [22]. LSTM-NDT [2] proposes an unsupervised and non-parametric thresholding approach to interpret predictions generated by an LSTM network. It builds up an automatic anomaly detection system to monitor the telemetry data sent back by the spacecraft. Ding et. al [14] proposes a real-time anomaly detection algorithm based on Hierarchical Temporal Memory (HTM) and Bayesian Network (BN). Gugulothu et. al [15] combines non-temporal dimensional reduction techniques and recurrent auto-encoders for time-series modeling through an end-to-end learning framework. DAGMM [9] focuses on anomaly detection of multivariate data without temporal dependencies. The input of DAGMM is just single entity observation (with multiple feature dimensions) instead of a temporal sequence.

2) Reconstruction-based models: A reconstruction-based model learns the representation for the entire time-series by reconstructing the original input based on some latent variables. Pankaj et. al [3] proposes an LSTM-based Encoder-Decoder framework to learn representations over normal time-series for anomaly detection. Kitsune [13] is an unsupervised model, mapping the features of an instance to integrated visible neurons which are then used to reconstruct the features back by an autoencoder. Generative Adversarial Networks (GANs) have also been widely used in multivariate time-series anomaly detection. Instead of treating each time-series independently, MAD-GAN [11] considers the entire variable set concurrently to capture the latent interactions among variables. GAN-Li [10] proposes a novel GAN-based anomaly detection method which deploys the GAN-trained discriminator together with the residuals between generator-reconstructed data and the actual samples. LSTM-VAE [12] integrates LSTM with variational auto-encoder that fuses signals and reconstructs expected distribution. For encoding, it projects multivariate observations and their temporal dependencies at each time step into a latent space using an LSTM-based encoder. For decoding, it estimates the expected distribution of multivariate inputs from the latent representation. OmniAnomaly [4] argues that deterministic methods may be misled by unpredictable instances and proposes a stochastic model for multivariate time-series anomaly detection. It captures the normal patterns

[1] https://github.com/Azure/Multivariate-AD

TABLE I
NOTATIONS

x	an instance of multivariate time-series input
n	the length of x in a pre-defined sliding window
k	the number of features (variates) in x
$\widetilde{x}$	input after preprocessing
v_i	input node representation for a GAT layer
h_i	output node representation for a GAT layer
α_{ij}	attention score of node j to node i in a GAT layer
d_1	hidden dimension of a GRU layer
d_2	hidden dimension of fully-connected layers in the forecasting-based model
d_3	latent space dimension of the VAE model
γ	hyper-parameter to combine multiple inference scores

behind data by learning robust representations of multivariate time-series with stochastic variable connection and planar normalizing flow. Their model considers patterns with low reconstruction probability as anomalies.

As introduced above, both forecasting-based and reconstruction-based models have shown their superiority in some specific situations. The forecasting-based model is specialized for feature engineering of next timestamp prediction, and construction-based model is good at capturing the data distribution of entire time-series. In our paper, we demonstrate that they are complementary to each other empirically. Moreover, none of the existing solutions capture the correlations between multiple features explicitly, which is emphatically addressed in this paper to enhance the performance of multivariate time-series anomaly detection.

III. Methodology

Multiple univariate time-series from the same entity forms a multivariate time-series. Multivariate time-series anomaly detection aims to detect anomalies at entity-level [4]. The problem can be defined as follows.

Problem Definition 1: An input of multivariate time-series anomaly detection is denoted by $x \in R^{n \times k}$, where n is the maximum length of timestamps, and k is the number of features in the input. For a long time-series, we generate fixed-length inputs by a sliding window of length n. The task of multivariate time-series anomaly detection is to produce an output vector $y \in R^n$, where $y_i \in \{0,1\}$ denotes whether the i^{th} timestamp is an anomaly.

We address this problem by modeling the inter-feature correlations and temporal dependencies with two graph attention networks in parallel, followed by a Gated Recurrent Unit (GRU) network to capture long-term dependencies in the sequential data. We also leverage the power of both forecasting-based and reconstruction-based models by optimizing a joint objective function. The following of this section is organized as follows. First, we will have a brief overview of our network in section III-A. Then, the details of data prepossessing, graph attention layers, and joint optimization will be presented in Section III-B, III-C, and III-D respectively. As last, the procedure of model inference is described in Section III-E. Table I summarizes the notations used in our model.

A. Overview

The overall network architecture of MTAD-GAT is shown in Figure 2, which is composed of the following modules in order:

1) We apply a 1-D convolution with kernel size 7 at the first layer to extract high-level features of each time-series input. As demonstrated in previous work [23], convolution operations are good at local feature engineering within a sliding window.
2) The outputs of 1-D convolution layer are processed by two parallel graph attention (GAT) [5] layers, which underline the relationships between multiple features and timestamps.
3) We concatenate the output representations from the 1-D convolution layer and two GAT layers, feed them into a Gated Recurrent Unit (GRU) [24] layer with d_1 hidden dimension. This layer is used for capturing sequential patterns in time-series.
4) The outputs of the GRU layer are fed into a forecasting-based model and a reconstruction based model in parallel to obtain the final result. We implement the forecasting-based model as a fully-connected network, and adopt VAE [25] for the reconstruction-based model.

B. Data Preprocessing

To improve the robustness of our model, we perform data normalization and cleaning for each individual time-series. Data normalization is applied on both training and testing set, while cleaning is only applied on the training set.

1) Data normalization: We normalize the time-series with the maximum and minimum values from the training data:

$$\widetilde{x} = \frac{x - \min(X_{\text{train}})}{\max(X_{\text{train}}) - \min(X_{\text{train}})} \tag{1}$$

where $\max(X_{\text{train}})$ and $\min(X_{\text{train}})$ are the maximum value and the minimum value of the training set respectively.

2) Data cleaning: Prediction-based and reconstruction-based models are sensitive to irregular and abnormal instances in the training data. To alleviate this problem, we employ a state-of-the-art univariate anomaly detection method, Spectral Residual (SR) [1], to detect anomaly timestamps in each individual time-series in the training data. Following [1], we set the threshold as 3 to generate anomaly detection results. Those detected anomaly timestamps will be replaced with normal values around that timestamp. Note that SR is light-weighted and adds little overhead to the entire model.

C. Graph Attention

Here we introduce the graph attention (GAT) layers in detail, which are the core designs of MTAD-GAT Net. A GAT layer is able to model the relationships between nodes in an arbitrary graph. Generally, given a graph with n nodes, i.e., $\{v_1, v_2, \cdots, v_n\}$, where v_i is the feature vector of each node,

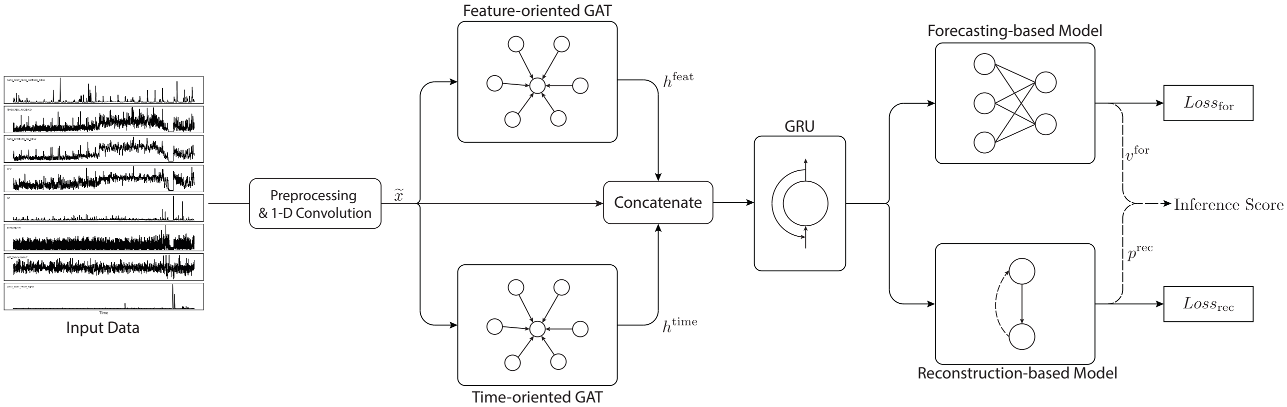


Fig. 2. The architecture of MTAD-GAT for multivariate time-series anomaly detection

a GAT layer computes the output representation for each node as follows:

$$h_i = \sigma(\sum_{j=1}^{L} \alpha_{ij} v_j), \tag{2}$$

where h_i denotes the output representation of node i, which has the same shape with input v_i; σ represents the sigmoid activation function; α_{ij} is the attention score which measures the contribution of node j to node i, where j is one of the adjacent nodes for node i; L denotes the number of adjacent nodes for node i.

The attention score α_{ij} can be computed by the following equations:

$$e_{ij} = \text{LeakyReLU}(w^{\top} \cdot (v_i \oplus v_j)) \tag{3}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{l=1}^{L} \exp(e_{il})}. \tag{4}$$

Here $\oplus$ represents concatenation of two node representations, $w \in R^{2m}$ is a column vector of learnable parameters where m is the dimension of the feature vector of each node, and LeakyReLU is a nonlinear activation function [26]. In the multivariate time-series anomaly detection scenario, we exploit two types of graph attention layers, namely *feature-oriented graph attention* and *time-oriented graph attention*.

1) Feature-oriented graph attention layer: On one hand, we need to detect multivariate correlations without any prior. Therefore, we treat the multivariate time-series as a complete graph, where each node represents a certain feature, and each edge denotes the relationship between two corresponding features. In this way, the relationships between adjacent nodes can be carefully captured through graph attention operations. Specifically, each node x_i is represented by a sequential vector $x_i = \{x_{i,t} | t \in [0, n)\}$ and there are totally k nodes, where n is the total number of timestamps and k is the total number of multivariate features. The layer is illustrated in Figure 3.

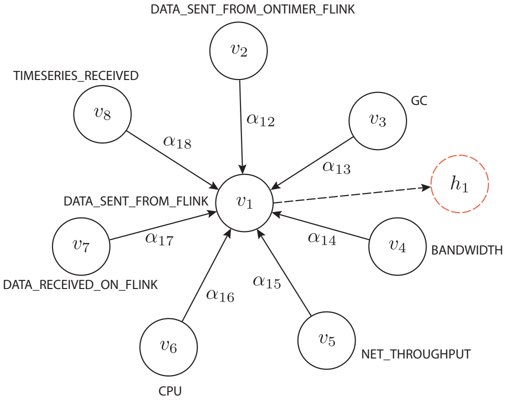


Fig. 3. Feature-oriented graph attention layer. Dashed circle is the final output.

2) Time-oriented graph attention layer: On the other hand, we leverage the power of graph attention network to capture temporal dependencies in time-series. We consider all the timestamps within a sliding window as a complete graph. Concretely, a node x_t represents the feature vector at timestamp t, and its adjacent nodes include all other timestamps in the current sliding window. This is much like a Transformer [27], where all words in a sequence are modeled by a fully-connected self-attention operation. The output of the feature-oriented graph attention layer is a matrix with shape $k \times n$, where each row is an n dimensional vector representing the output for each node and there are in total k nodes. Similarly, the output of the time-oriented graph attention layer has a shape of $n \times k$. We concatenate the outputs from the feature-oriented graph attention layer and the time-oriented graph attention layer as well as the preprocessed $\tilde{x}$ to a matrix with shape $n \times 3k$, where each row represents a $3k$ dimensional feature vector for each timestamp, to fuse the information from different sources.

D. Joint Optimization

As mentioned previously, the forecasting and the reconstruction models have distinct advantages respectively which are complementary with each other. Our model includes a *forecasting-based* model to predict the value at next timestamp and a *reconstruction-based* model to capture the data distribution of entire time-series. During the training process, the parameters from both models are updated simultaneously. The loss function is defined as the sum of two optimization targets, i.e., $Loss = Loss_{for} + Loss_{rec}$, where $Loss_{for}$ denotes the loss function of forecasting-based model and $Loss_{rec}$ denotes the loss function of reconstruction-based model.

1) Forecasting-based model: The forecasting-based model predicts the value at next timestamp. We stack three fully-connected layers with hidden dimensions d_2 after the GRU layer as the forecasting-based model. The loss function can be formulated as Root Mean Square Error (RMSE):

$$Loss_{for} = \sqrt{\sum_{i=1}^{k}(x_{n,i} - \hat{x}_{n,i})^2}. \tag{5}$$

where x_n denotes the next timestamp for the current input $x = (x_0, x_1, ..., x_{n-1})$; $x_{n,i}$ represents the value for the i^{th} feature in x_n; and $\hat{x}_{n,i}$ is the value predicted by the forecasting-based model.

2) Reconstruction-based model: The reconstruction-based model aims to learn a marginal distribution of data over a latent representation z. We employ Variational Auto-Encoder (VAE) [25], which provides a *probabilistic* manner for describing an observation in the latent space. By treating the values of time-series as variables, the VAE model is able to capture the data distribution of entire time-series. Given an input x, it is supposed to be reconstructed from a conditional distribution $p_\theta(x|z)$, where $z \in R^{d_3}$ is the vector representation in a latent space. The optimization target is to find the best model parameters that reconstruct x with the most close data distribution. The true posterior density can be given by:

$$p_\theta(z|x) = p_\theta(x|z)p_\theta(z)/p_\theta(x) \tag{6}$$

where the marginal density is formulated as

$$p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz \tag{7}$$

It is intractable to calculate the above equation, so we need to introduce a recognition model $q_\phi(z|x)$ to approximate the posterior distribution. Given the recognition model (encoder) $q_\phi(z|x)$ and the generative model (decoder) $p_\theta(\hat{x}|z)$, the reconstruction-based loss function can be computed as follows:

$$Loss_{rec} = -\mathrm{E}_{q_\phi(z|x)}[logp_\theta(x|z)] + D_{KL}(q_\phi(z|x)||p_\theta(z)) \tag{8}$$

where the first term is the expected negative log-likelihood of the given input. The second term is the Kullback-Leibler divergence between the encoder's distribution $q_\phi(z|x)$ and $p_\theta(z)$, which can be viewed as a regularizer. Theoretically, the negation of this loss function is a practical estimator of the lower bound for the intractable log likelihood, $\log p_\theta(x)$, so we can differentiate and optimize this loss function instead.

E. Model Inference

Corresponding to the joint optimization target, we also have two inference results for each timestamp. One is prediction value $\{\hat{x}_i|i = 1, 2, ..., k\}$ calculated by the forecasting-based model, and the other is reconstruction probability $\{p_i|i = 1, 2, .., k\}$ obtained from the reconstruction-based model. The final inference score balances their benefits to maximize the overall effectiveness of anomaly detection. We calculate an inference score s_i for each feature and take the summation of all features as the final inference score. We identify a timestamp as an anomaly if its corresponding inference score is larger than a threshold. We use Peak Over Threshold (POT) [28] to choose the threshold automatically. Specifically, the inference score can be calculated by:

$$score = \sum_{i=1}^{k} s_i = \sum_{i=1}^{k} \frac{(\hat{x}_i - x_i)^2 + \gamma \times (1 - p_i)}{1 + \gamma} \tag{9}$$

where $(\hat{x}_i - x_i)^2$ is the squared error between the forecasting value $\hat{x}_i$ and the actual value x_i, indicating how much the actual value of feature i is deviated from prediction; $(1 - p_i)$ is the probability of encountering an abnormal value for feature i according to the reconstruction model; k is the total number of features; and γ is a hyper-parameter to combine the forecasting-based error and the reconstruction-based probability. γ is chosen by grid search on the validation set, and a sensitivity study will be provided in the analysis section.

IV. Experiments

A. Datasets and Metrics

a) Datasets.: We use three datasets to verify the effectiveness of our model, namely SMAP (Soil Moisture Active Passive satellite), MSL (Mars Science Laboratory rover) and TSA (Time Series Anomaly detection system). SMAP and MSL are spacecraft datasets collected by NASA [29]. TSA is a dataset collected from our own time-series anomaly detection system that processes time-series by Flink[2]. We gathered two months metrics of services and hardwares from the Flink cluster of the system, and an example is shown in Figure 1. Anomaly labels used for evaluation in TSA dataset have been labeled based on incident reports from the system. The statistics of these three datasets are shown in Table II.

b) Metrics.: We use precision, recall and F1-score to indicate the performance of our model. Experience about AUC scores are also conducted. However we found it is hard to measure the performance by AUC for most state-of-the-art methods including ours have more than 0.97 AUC scores. In practice, anomalous observations usually form contiguous segment since they occur in a continuous manner. Following the evaluation strategy in [4], we treat the whole segment as correct if any observation in this segment is detected as anomaly correctly.

B. Setup

We compare MTAD-GAT with state-of-the-art models for multivariate time-series anomaly detection, including OmniAnomaly [4], LSTM-NDT [2], KitNet [13], DAGMM [9],

[2] https://flink.apache.org/

TABLE II
DATASET STATISTICS

	SMAP	MSL	TSA
Number of sequences	25	55	18
Training set size	135183	58317	39312
Testing set size	427617	73729	51408
Anomaly Rate(%)	13.13	10.27	10.58

GAN-Li [10], MAD-GAN [11] and LSTM-VAE [12]. We use the same sliding window size $n = 100$ for all models. In our method, we set $\gamma = 0.8$ through a grid search on the validation set. The hidden dimension sizes of the GRU layer (d_1), the fully-connected layers (d_2), and the VAE model (d_3) are set as 300 empirically. We use the Adam optimizer to train our model for 100 epochs with an initial learning rate 0.001. We compare the performance of state-of-the-art models in Section IV-B. To better understand our model, we also examine the effectiveness of different components through analysis in Section V.

C. Comparison with SOTAs

As shown in Table III, MTAD-GAT shows excellent generalization capability and achieves the best F1 scores consistently on three datasets. Specifically, we achieve 1%, 1%, and 9% improvement over the best state-of-the-art performance on SMAP, MSL and TSA datasets respectively. These performance lifts are significant as verified by a hypothesis testing.

The limitation of OmniAnomaly lies in not addressing the feature correlations explicitly in the model, which is essential to the success of multivariate time-series anomaly detection. As introduced earlier, our feature-oriented GAT layer is by design to tackle this problem. The superiority of this layer is verified in our experiments, as our model outperforms OmniAnomaly significantly and consistently on all three datasets.

The temporal information is also crucial for multivariate time-series anomaly detection. The performance of DAGMM is not ideal, because it dose not take the temporal information into consideration. In our model, GRU is used to capture long-term temporal dependencies, and a time-oriented GAT layer is applied to calculate attention scores between correlated timestamps. These designs are helpful to achieve a much better performance than DAGMM, and we also conduct additional experiments (shown in Section V) to compare different design variations.

Forecasting-based methods, such as LSTM-NDT, have good performances on SMAP, but perform poorly on MSL and TSA datasets. They are sensitive to different scenarios because it cannot model unpredictable cases. On the other hand, reconstruction-based methods (for example, OmniAnomaly) achieve much better results on the MSL and TSA datasets. This implies that both *forecasting-based* and *reconstruction-based* models have their own advantages, and the joint optimization strategy proposed in this paper is beneficial to the final performance.

D. Evaluation with Different Delays

In practice, anomalies usually occur in a continuous segment and we require a model to detect them as soon as possible to take quick actions. Therefore, we compare our model with the current best baseline, OmniAnomaly under different delay metrics. We follow the evaluation protocol described in [1], that is, treating the whole segment as true positives, if and only if there is an anomaly point detected correctly and its timestamp is at most δ steps after the first anomaly of the segment.

Figure 4 compares the F1-scores of our model and OmniAnomaly models for different delay metrics on three datasets. Notice that the F1 score becomes larger as the delay δ increases, and when δ is large enough, the number will match that reported in Table III. Overall, Our model achieves better performance consistently, especially when the acceptable delay is small. When $\delta = 10$, the relative enhancements on these datasets are 53.98%, 13.04%, and 19.93%. We can also observe a performance boost when the delay increases from 5 to 10 on all the three datasets. Therefore, the proposed anomaly detection model based on graph attention network is able to alert for potential incidents on time in real scenarios without excessive losses.

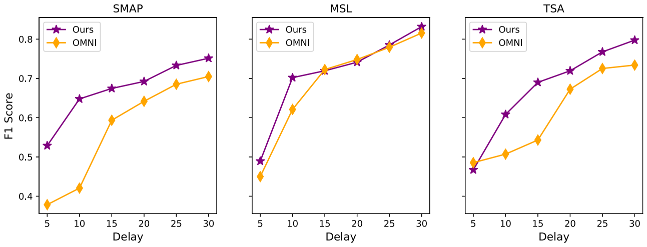


Fig. 4. Comparison with different delay constraints

V. ANALYSES

In this section, we analyze the effectiveness of graph attention and joint optimization through comprehensive experiments (summarized in Table IV), which will be discussed in detail with deeper insights. Then, we perform sensitivity study on the parameter γ and present experimental study on anomaly diagnosis.

A. Effectiveness of Graph Attention

We examine the influence of two graph attention layers in our model by disabling the feature-oriented GAT layer (denoted as *w/o feature*) and the time-oriented GAT layer (denoted as *w/o time*) one at a time. We adjust the number of parameters of both models to remove the impact of model complexity. From Table IV, we find that *w/o feature* causes 3.2% decline in average F1 score, while *w/o time* causes 2.5% decline on it. Specifically, on TSA dataset, F1 score of *w/o feature* drops by 4.8% relatively compared to the full implementation. In fact, features like *CPU* and *MEMORY* are highly correlated on this dataset, and the feature-oriented GAT

TABLE III
PERFORMANCE OF OUR MODELS AND BASELINES.

Method	SMAP			MSL			TSA		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Reconstruction based models									
OmniAnomaly	0.7416	0.9776	0.8434	0.8867	0.9117	0.8989	0.7028	0.8039	0.7499
KitNet	0.7725	0.8327	0.8014	0.6312	0.7936	0.7031	0.5579	0.8012	0.6577
GAN-Li	0.6710	0.8706	0.7579	0.7102	0.8706	0.7823	0.5302	0.7551	0.6229
MAD-GAN	0.8049	0.8214	0.8131	0.8517	0.8991	0.8747	0.5510	0. 8284	0.6620
LSTM-VAE	0.8551	0.6366	0.7298	0.5257	0.9546	0.6780	0.6970	0.7736	0.7333
Forecasting based models									
LSTM-NDT	0.8965	0.8846	0.8905	0.5934	0.5374	0.5640	0.5833	0.7232	0.6457
DAGMM	0.5845	0.9058	0.7105	0.5412	0.9934	0.7007	0.5351	0.8845	0.6668
MTAD-GAT	0.8906	0.9123	**0.9013**	0.8754	0.9440	**0.9084**	0.6951	0.9352	**0.7975**

TABLE IV
QUANTITATIVE RESULTS FOR ANALYSES. F1 SCORES ARE REPORTED.

Model	SMAP	MSL	TSA
MTAD-GAT	**0.9013**	**0.9084**	**0.7975**
w/o feature	0.8783	0.8851	0.7474
w/o time	0.8832	0.8897	0.7582
w/o prediction	0.8731	0.8857	0.7380
w/o reconstruction	0.8352	0.8058	0.7278

layer is useful to capture this correlation accurately for better anomaly detection. Moreover, the time-oriented GAT layer is also crucial to the final performance, although a GRU layer is already adopted for modeling the temporal dependencies. A potential explanation is that the time-oriented GAT layer can model the relationship between a pair of timestamps directly even if they are not adjacent. In this way, some long-term dependencies between timestamps can be modeled more explicitly.

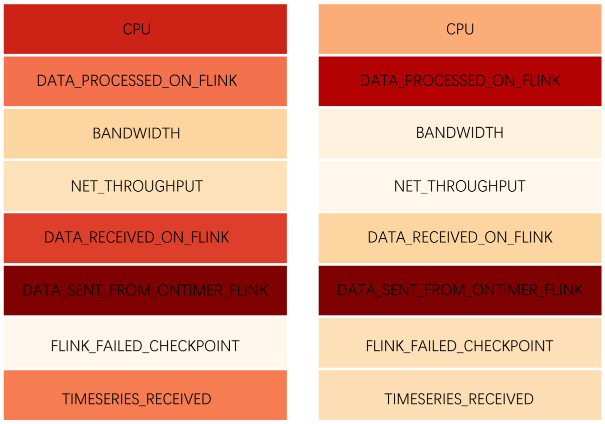


Fig. 5. Illustration of attention scores for feature *DATA_SENT_FROM_FLINK* in the case of Figure 1. The left part visualizes the average attention scores at normal timestamps while the right part visualizes the attention scores when anomaly occurs. Darker color indicates higher attention scores.

Here we leverage the example in Figure 1 to explain why the feature-oriented GAT layer is helpful to the performance of anomaly detection. As the data source of TSA dataset is processed by Flink, a couple of features have been collected to reflect the running state of the system. *TIMESERIES_RECEIVED* indicates the number of time-series monitored in the system, which are sent to Flink for calculation and delivered to down-streaming components. *DATA_RECEIVED_ON_FLINK*, *DATA_SENT_FROM_FLINK*, and *CPU* represent the data volume received on Flink, sent by Flink, and CPU utilization respectively. When the system works normally, other features should have strong positive correlation with *DATA_SENT_FROM_FLINK*. In Figure 5, we visualize the attention score α_{ij} calculated by the feature-oriented GAT layer based on Equation (4). As illustrated in the left part of this figure, our model correctly learns the most relevant features to *DATA_SENT_FROM_FLINK* under normal circumstances. When anomaly occurs (corresponding to the red segment in Figure 1), the attention scores are visualized in the right part of Figure 5. We observe that the features *CPU* and *DATA_RECEIVED_ON_FLINK* demonstrate much weaker correlations with *DATA_SENT_FROM_FLINK*. Actually, there is a traffic drop in the system, meanwhile, a garbage collection issue has been found on the Flink cluster. Thus, the job couldn't complete the checkpoint and keeps reprocessing the last batch of the input stream. The continuously re-processed stream causes spike in the output metric, so *DATA_SENT_FROM_FLINK* has shown evident inconsistency.

B. Effectiveness of Joint Optimization

In this section, we show the effectiveness of the joint optimization strategy by comparing F1 scores with controlled experiments. We compare our model with its simplified counterparts with single optimization target. Quantitative results in Table IV show that the reconstruction-based variant (denoted as *w/o prediction*) achieves better performance than the forecasting-based variant (denoted as *w/o reconstruction*), but both of them degrade the original performance of the model significantly.

Forecasting-based model predicts the actual value of next timestamp in a deterministic manner, which is sensitive to randomness of time-series. On the other hand, reconstruction

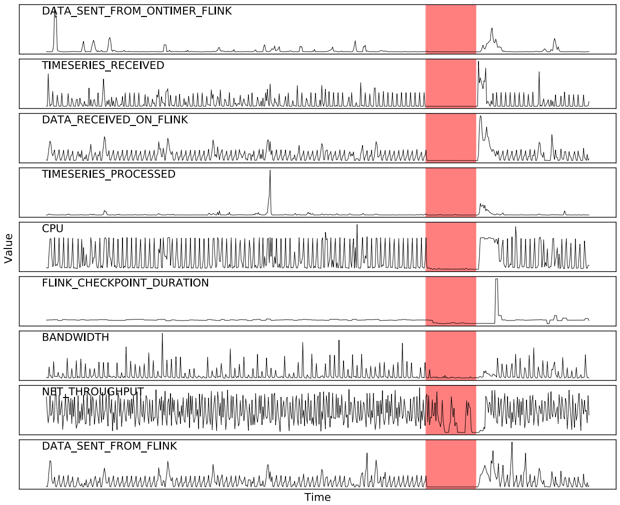


Fig. 6. A failure case for reconstruction-based model

model alleviates this problem by learning a distribution of stochastic variables, which is more robust to perturbations and noises. However, there are still some cases that the reconstruction-based variant is not able to handle. For example, Figure 6 shows a periodic time-series containing an anomaly segment that the reconstruction-based variant does not find out. Generally, a reconstruction-based model is good at capturing global data distribution, but it may neglect sudden perturbations to disrupt periodicity in a time-series, especially when the values still conform to normal distribution. As shown in Figure 6, the time-series has frequent periodicity in normal circumstances, so an anomaly should be detected when it is broken at the red segment. However, as the values still conform to normal distribution, the reconstruction-based variant fails to report the incident. Instead, this is successfully captured by the forecasting-based variant. Therefore, the joint optimization target is useful for achieving better anomaly detection results.

C. Analysis of γ

Originally, we made forecasting and reconstruction part have the same weight for the training loss and the anomaly score. Then we conduct experiments to optimize the ratio of them during training and inference. And we found there would not be significant improvement by adjusting the balance between forecasting and reconstruction during training. For it always take more epochs for training a reconstruction-based method than a forecasting-based method, and when the forecasting part has converged, its loss would be a constant to reconstruction part. So we perform additional experiments on analyzing the influence of γ which balances the forecasting-based error and the reconstruction-based probability.

We evaluate precision, recall, and F1 scores on three datasets with different values of γ. Results are summarized in Table V. We notice that different settings of γ achieve similar performance on all three evaluation metrics. Specifically, when $\gamma = 0.8$, we achieve the best performance. The result shows that our model is robust against γ. By setting γ between 0.4 and 1.0, we can always achieve better results consistently than other state-of-the-art solutions listed in Table III.

D. Anomaly Diagnosis

Besides detecting anomalies in multivariate time-series, our method also provides useful insights for anomaly diagnosis. In real applications, those insights help people find the root cause of an incident and save the efforts to resolve it. For an instance of multivariate time-series, $x \in R^{n\times k}$, we define a collection of features, $\{x_i\} \in R^{n\times m} \subseteq x$ as the root causes, where $x_i \in R^n$ represents for the univariate time-series corresponding to a certain feature.

An algorithm of anomaly diagnosis sorts these features by the possibility of being the actual root cause for an incident. Then, it selects the top m features to be the predicted root causes. In our method, we have a set of inference scores for each instance which can be calculated by Equation (9), that is, $S = \{s_1, s_2, ...s_k\}$, where k is the number of features and s_i is the inference score for feature i. In our experiments, we select top 8 features with largest inference scores as root cause candidates.

We demonstrate diagnostic performance on our dataset TSA. We leverage our incident records to label root causes for each anomaly event. Along with the system developers and operators, we label more than 700 qualified instances in the test data which have clear root causes in the incident records. We use two metrics to evaluate the performance of anomaly diagnosis, HitRate@P% [4] and NDCG [30]. HitRate is used to measure how many ground truths have been included in the top candidates. It can be calculated by $HitRate@P\% = \frac{Hit@\lfloor P\% \times |GT| \rfloor}{|GT|}$, where $|GT|$ is the number of ground truth for a single anomaly event. Normalized Discounted Cumulative Gain (NDCG) is a popular measure for relevance evaluation, and we adopt it here to quantify the ranking accuracy of root causes.

As shown in Table VI, our model demonstrates the capability of finding the top root cause features. Especially, 70% true root cause has been captured in the diagnose results and the ranking performance of top 5 results also indicate our approach has a high probability to find the root cause at the top 5 candidates learned by the algorithm. In real AI-ops scenarios, this ability help us find the real causes and solve the problem as soon as possible.

The ability of anomaly diagnosis is much owing to the graph attention layer leveraged in the model. In real scenarios, a feature could be the actual root cause if its correlations with others have changed. For example in Figure 1, when there is a dip in the input stream, we expect a lower value in the output steam. Thus, features *DATA_SENT_FROM_FLINK* and *DATA_RECEIVED_FROM_FLINK* should keep consistent tendency. When there is an abnormal correlation between them, we can speculate that an incident occurs in the system, and these two features may be the potential root causes. The feature-oriented graph attention layer in our model captures the correlation

TABLE V
QUANTITATIVE RESULTS FOR DIFFERENT γ.

γ	SMAP			MSL			TSA		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
1.0	0.8832	0.9003	0.8917	0.8772	0.9413	0.9081	0.6874	0.9147	0.7849
0.8	**0.8906**	**0.9123**	**0.9013**	**0.8754**	**0.9440**	**0.9084**	**0.6951**	**0.9352**	**0.7975**
0.6	0.8814	0.9037	0.8924	0.8802	0.9315	0.9051	0.6861	0.9174	0.7851
0.4	0.8849	0.8991	0.8900	0.8817	0.9196	0.9003	0.6839	0.9233	0.7858

TABLE VI
QUANTITATIVE RESULTS FOR ANOMALY DIAGNOSIS

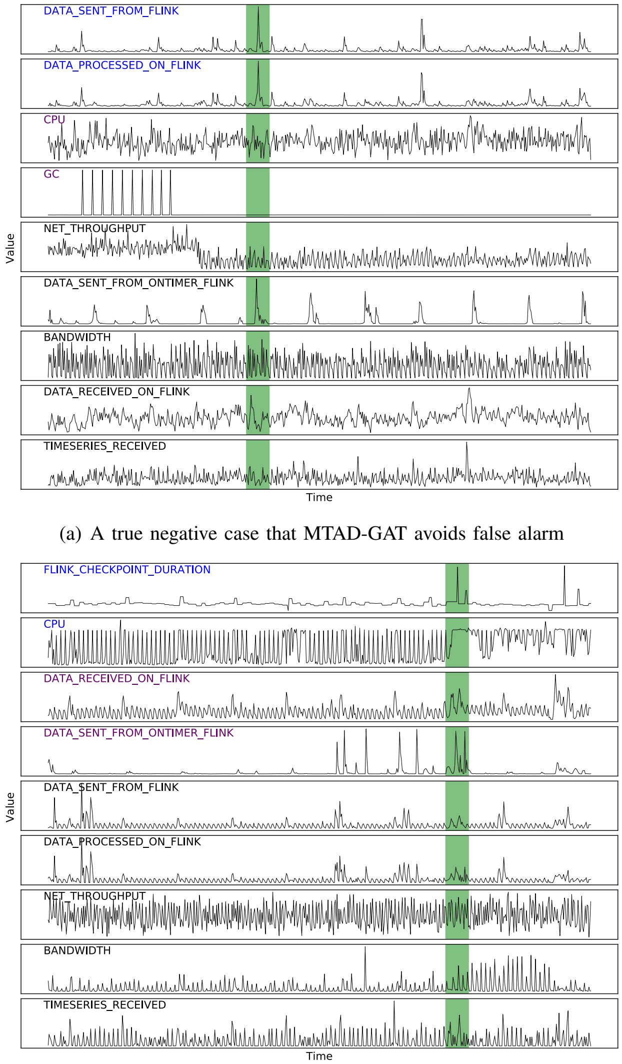
Model	NDCG@5	HitRate@100%	HitRate@150%
MTAD-GAT	0.8556	0.7428	0.8561

between features properly, so this complicated circumstance can be well-handled.

VI. CASE STUDY

In this section, we provide another successful case and a failure case to analyze the advantages and deficiencies of our model. In Figure 7(a), we show a true negative case of our model. In the green segment, spikes show up in several time-series. If we detect each univariate time-series separately, we may alert for an abnormal incident. But considering feature relationships, we can find that the correlations between those features remain unchanged, although their volumes have greatly boosted. Actually, these spikes indicate that more data has been processed in the Flink job, and it is normal to have spikes in the corresponding time-series of *DATA_SENT_FROM_FLINK* and *DATA_PROCESSED_ON_FLINK*. Meanwhile, the values of *CPU* and *GC* keep stable, indicating that the system is in a healthy state and the increased traffic can be processed smoothly. To conclude, our approach is advantageous in dealing these cases and thus decreases the number of false alerts largely in our monitoring system.

Next, we analyze a false positive case plotted in Figure 7(b). The green segment in the figure represents for a normal instance that is falsely detected as an anomaly by our model. Through detailed examination we find that our approach identifies abnormal events on *FLINK_CHECKPOINT_DURATION* and *CPU* because there are unusual spikes as shown in Figure 7(b). However, this is caused by an increase of input data volume and the related features such as *DATA_RECEIVED_ON_FLINK* and *DATA_SENT_FROM_ONTIMER_FLINK* also demonstrate the same pattern. In this case, although a Flink job takes more time to complete the checkpoint, the traffic peak does not last for a long time, so it is not considered as an anomaly in the system. However, as this spike appears seldom in the history, the unsupervised anomaly detection algorithm naturally treats it as an abnormal case. We may need more domain knowledge or user feedbacks to solve this problem, which is left for future work.



(a) A true negative case that MTAD-GAT avoids false alarm

(b) A false positive case that MTAD-GAT generates false alarm

Fig. 7. Case Study

VII. CONCLUSION

In this paper, we propose a novel framework based on graph attention network for multivariate time-series anomaly detection. By learning feature-wise and temporal relationships of multivariate time-series and leveraging a joint optimization strategy, our method outperforms other state-of-the-art models on three datasets consistently. In addition, our model demonstrates good capability of anomaly diagnosis, which helps customers to find the actual root causes for an anomaly event. Extensive analysis provides more insights into the model and verifies the effectiveness of the proposed architecture. Future works may come from two aspects. First, our model has no prior knowledge on the correlations between features. Using user feedback or domain prior knowledge may benefit the performance. Second, current anomaly diagnosis is studied on relatively simple scenarios. We may utilize our model to investigate more complicated cases.

References

[1] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, and Q. Zhang, "Time-series anomaly detection service at microsoft," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 3009–3017.

[2] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 387–395.

[3] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[4] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2828–2837.

[5] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[6] J. Wong, C. Colburn, E. Meeks, and S. Vedaraman, "Rad—outlier detection on big data," *Web blog post. The Netflix Tech Blog. Netflix*, vol. 19, 2015.

[7] A. Kejariwal, "Introducing practical and robust anomaly detection in a time series," *Twitter Engineering Blog. Web*, vol. 15, 2015.

[8] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *Proceedings*. Presses universitaires de Louvain, 2015, p. 89.

[9] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *International Conference on Learning Representations*, 2018.

[10] D. Li, D. Chen, J. Goh, and S.-K. Ng, "Anomaly detection with generative adversarial networks for multivariate time series," *arXiv preprint arXiv:1809.04758*, 2018.

[11] D. Li, D. Chen, L. Shi, B. Jin, J. Goh, and S.-K. Ng, "Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks," *arXiv preprint arXiv:1901.04997*, 2019.

[12] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.

[13] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," *arXiv preprint arXiv:1802.09089*, 2018.

[14] N. Ding, H. Gao, H. Bu, H. Ma, and H. Si, "Multivariate-time-series-driven real-time anomaly detection based on bayesian network," *Sensors*, vol. 18, no. 10, p. 3367, 2018.

[15] N. Gugulothu, P. Malhotra, L. Vig, and G. Shroff, "Sparse neural networks for anomaly detection in high-dimensional time series," in *AI4IOT workshop in conjunction with ICML, IJCAI and ECAI*, 2018.

[16] H. Mousavi, S. Mohammadi, A. Perina, R. Chellali, and V. Murino, "Analyzing tracklets for the detection of abnormal crowd behavior," in *2015 IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2015, pp. 148–155.

[17] B. Rosner, "Percentage points for a generalized esd many-outlier procedure," *Technometrics*, vol. 25, no. 2, pp. 165–172, 1983.

[18] W. Lu and A. A. Ghorbani, "Network anomaly detection based on wavelet analysis," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, pp. 1–16, 2008.

[19] A. Mahimkar, Z. Ge, J. Wang, J. Yates, Y. Zhang, J. Emmons, B. Huntley, and M. Stockert, "Rapid detection of maintenance induced changes in service performance," in *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*, 2011, pp. 1–12.

[20] Y. Zhang, Z. Ge, A. Greenberg, and M. Roughan, "Network anomography," in *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, 2005, pp. 30–30.

[21] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 187–196.

[22] N. Laptev, S. Amizadeh, and I. Flint, "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, 2015, pp. 1939–1947.

[23] C. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 69–78.

[24] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[25] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[26] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[28] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouet, "Anomaly detection in streams with extreme value theory," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1067–1075.

[29] P. O'Neill, D. Entekhabi, E. Njoku, and K. Kellogg, "The nasa soil moisture active passive (smap) mission: Overview," in *2010 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2010, pp. 3236–3239.

[30] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 422–446, 2002.