Unsupervised Anomaly Detection in Multivariate Time Series through Transformer-based Variational Autoencoder

Hongwei Zhang[1], Yuanqing Xia[1,*], Tijin Yan[1], Guiyang Liu[2].
1. Key Laboratory of Intelligent Control and Decision of Complex Systems,
School of Automation, Beijing Institute of Technology, Beijing 100081, P. R. China
2. Alibaba Group, Hangzhou 311121, P. R. China
*Corresponding author: xia_yuanqing@bit.edu.cn

Abstract: Modern industrial devices often use multiple sensors to detect the status of system, which produce a large amount of multivariate time series. Due to the complex temporal dependency of intra-channel and inter-correlations among different channels, few of proposed algorithms have addressed these challenges for anomaly detection in multivariate time series. Besides, previous work does not consider future dependency, which has been shown to be critical for sequential data modeling. In this paper, we develop an unsupervised anomaly detection algorithm *TransAnomaly*, which integrates Transformer, variational autoencoder (VAE) and nonlinear state space model. *TransAnomaly* not only reduces the computational complexity and allows for more parallelization but also provides explainable insights. To the best of our knowledge, it is the first model that combines VAE and Transformer for multivariate time series anomaly detection. Extensive experiments on several public real-world datasets show that *TransAnomaly* outperforms state-of-the-art baseline methods while training cost is reduced by nearly 80%.

Key Words: Multivariate Time Series, Anomaly Detection, Transformer, Parallelization

1 INTRODUCTION

Anomaly detection, a popular topic in industrial applications and research area, aims to discover or detect anomalous data from normal data. Establishing an accurate and reliable monitor service can help operators to know the status of system in time. The status of the system is always complex as there are many metrics. For example, a server in the cluster has various KPIs (Key Performance Indicators), such as the utilization of CPU, memory and disk, CPU load and so on. A robot system has to monitor the position and velocity when moving arms. Spacecrafts need to record thousands of telemetry signals such as temperature, pressure, radiation and others. In this situation, it is often not comprehensive to use a single index to evaluate system status. Therefore, it is necessary to develop a multivariate time series anomaly detection algorithm.

Currently, the collected data of the cyber-physical system has the characteristics of highly nonlinear dynamics, high dimension and high uncertainty, which makes it a challenge to identify the abnormal status. It requires the model to have powerful expression ability and can deal with the stochasticity of data.

Anomalies usually deviate from normal pattern and distribution. Great efforts have been made to detect and discover anomalies. Previous traditional studies [4, 11] have made progress on public datasets but is not satisfactory in real application as these methods will not work without labels. As a generative model, variational autoencoder (VAE) [8] can enhance the model to learn the stochastic factors. Besides, VAE is a reconstruction-based model, which can get better results in anomaly detection. Recurrent neural networks (RNNs) have been widely used in sequence modeling as they can capture temporal dependency. Long short-term memory (LSTM) [5] and gated recurrent unit (GRU) [1] have been firmly established as the state-of-the-art approaches in sequential data modeling. Various multivariate time series anomaly detection algorithms have been proposed [12, 15] to solve the problem. However, due to the RNN's structure limitation (*i.e.* RNN is an autoregressive model) , it is difficult to be parallelized and only learns a local representation in terms of the past input, which makes the RNN-based methods is time-consuming when training and inferencing. Besides, a single channel data has different shapes at different stages, and there are also correlations among different channels in multivariate time series.

Recently, Transformer [16] has shown strong capacity on machine translations and other tasks. Transformer replaces the recurrent layers with multi-head self-attention, which speeds up the training process and is a better feature extractor [16]. In Transformer, different heads focus on different temporal patterns while attention provides an interpretable result.

To address the above challenges, we develop an unsupervised anomaly detection algorithm *TransAnomaly*. Specifically, we introduce Transformer, a brand new architecture which replaces the recurrent layers with multi-head self-attention in natural language processing (NLP) domain, to our anomaly detection application.

Compared to RNNs, our model allows for efficient paral-

This work is supported by the National Key Research and Development Program of China under Grant 2018YFB1003700.

2021 33rd Chinese Control and Decision Conference (CCDC) | 978-1-6654-4089-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/CCDC52312.2021.9601669

lelization on GPUs. Attention across different time implemented by Transformer helps us understand the location and significance of different time in each channel. The reason for this design is that some channels provide little information in one segment, but is very critical in another segment. It helps us to comprehensively understand and analyze the data. In order to further improve the expression ability of the model, we make reasonable assumptions on latent variable's distribution in VAE. Specifically, we consider the nonlinear state space model. In addition, our model can learn a global representation by considering the future dependency.

In summary, our main contributions are as follows:

- We propose a novel multivariate time series anomaly detection algorithm *TransAnomaly*. To the best of our knowledge, it is the first model combining VAE and Transformer for time series analysis, which not only considers the future dependency explicitly but also runs in parallel.
- We introduce Gated Transition Function(GTF), which is a non-linear transition relation, to model the dependency of latent variables.
- We conduct extensive experiments on several public real-world datasets. Our results demonstrate the superior performance of *TransAnomaly* over state-of-the-art baseline methods.

2 PRELIMINARIES

In this section, we introduce the problem and describe some background of the key components of our model.

2.1 Problem Definition

Suppose X is a collection of M related univariate time series. Let $X = (x_1, x_2, ..., x_T) \in \mathbb{R}^{M\times T}$, where T is the length of X and $x_t = (x_t^1, x_t^2, ..., x_t^M) \in \mathbb{R}^M$ is an M-dimensional vector at time $t \in [1, T]$. We denote $(x_1, ..., x_T)$ as $x_{1:T}$ for brevity.

Given a time t, whether x_t is recognized as an anomalous value depends on the past input series. The model returns an anomaly score. We can set a threshold manually or by automatic threshold selection algorithm.

2.2 VAE

VAE is a deep generative model [8] which combines Bayesian inference with the autoencoder framework, and has been widely applied to anomaly detection for time series. The key idea of VAE is embedding high-dimensional data x to the latent space with a reduced dimension. From the perspective of variational inference, given a prior $p(z)$, x is sampled from $p_\theta(x|z)$, which is posterior distributions. In general $p(z)$ is set as Gaussian distribution. VAE approximates $p_\theta(x|z)$ using another network $q_\phi(x|z)$ as the true posterior is intractable to compute. We have the following:

$$\begin{aligned}\log\ p_\theta(x) &= \log \mathbb{E}_{q_\phi(z|x)}[\frac{p_\theta(x|z)p_\theta(z)}{q_\phi(z|x)}] \\ &\geq \mathbb{E}_{q_\phi(z|x)}[\log \frac{p_\theta(x|z)p_\theta(z)}{q_\phi(z|x)}] \\ &= \mathcal{L}(x),\end{aligned} \tag{1}$$

where $\mathcal{L}(x)$ is called variational lower bound. Eq.(1) can also be rewritten as

$$\mathcal{L}(x) = \mathbb{E}_{q_\phi(z|x)}[\log(p_\theta(x|z))] - D_{KL}[q_\phi(z|x) \parallel p_\theta(z)], \tag{2}$$

where the first term is reconstruction loss, which shows how close between the input data x and reconstruction data x', the second term is a regularization loss, which shows the difference between the posterior distribution $q_\phi(z|x)$ and the prior distribution $p_\theta(z)$.
Monte Carlo integration[13] is often used to estimate expectations of the function $f(z)$ w.r.t. $q_\phi(z|x)$ as follow:

$$\mathbb{E}_{q_\phi(z|x)}[f(z)] \simeq \frac{1}{L}\sum_{l=1}^{L} f(z^{(l)}), \tag{3}$$

where $z^{(l)}, l = 1, ..., L$ are samples from $q_\phi(z|x)$.
Using Eq.(3), the ELBO (Eq.(2)) can be written as Eq.(4):

$$\mathcal{L}(x)=\frac{1}{L}\sum_{l=1}^{L}[\log(p_\theta(x|z^{(l)}))+\log(p_\theta(z^{(l)}))-\log(q_\phi(z^{(l)}|x))]. \tag{4}$$

2.3 Transformer

Transformer is a model which is composed of a stack of N multi-head self-attention layers and point-wise, fully connected layers for both encoder and decoder. Transformer eschews recurrence and relies entirely on self-attention mechanism to draw global dependency between input and output, and allows for significantly more parallelization. It has been verified to perform well on various NLP tasks. Due to self-attention, the representation of each data point is related to others, which gives a global representation for each data point.
For simplicity, we denote queries, keys and values for attention as Q, K and V and multi-head self-attention as MultiHead(Q, K, V), point-wise feed-forward networks as FFN(x) and layer normalization as LayerNorm(x). For each head, we denote attention as Attention(Q, K, V) (Eq.(5))

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V, \tag{5}$$

where d_k is the dimension of keys.
Transformer uses multi-head attention to learn different representations at different positions.

$$\text{MultiHead}(Q, K, V) = \text{Concat}(head_1, ..., head_h), \tag{6}$$

where $head_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$. Since the model contains no recurrence and no convolution, Transformer adds positional encodings to the input embedding to make use of the order of the sequence.

2.4 Peaks-Over-Threshold Approach

The Peaks-Over-Threshold (POT) is based on extreme value theory, which describes the properties of extreme values, one of which is the distribution of extreme values. [14] adapts the generalized Pareto distribution (GPD) to fit the the excess over a threshold th and use POT to detect the raw stream data whereas we uses it to determine the threshold on the reconstruction probability. [15] modifies the GPD function to fit the low end of the distribution. We follow the setting of [15].

3 ANOMALY DETECTION FRAMEWORK

In this section, we first give the model a probabilistic explanation to show that adding future dependency is reasonable by factorizing the sequence's distribution over time.

Inspired by Transformer's performance in NLP tasks, we embed the TransformerBlock to VAE architecture. Unlike the RNN-based models, TransformerBlock can capture the relationship in any pairs of vectors in different location, which means the output contains all the information of input vector. Next we give an explanation that why introducing TransformerBlock to our model makes sense in terms of probability. We formulate the posterior distribution over time as:

$$\begin{aligned} p(z_{1:T}|x_{1:T}) &= p(z_1|x_{1:T}) \prod_{t=2}^{T} p(z_t|z_{t-1}, x_{1:T}) \\ &= p(z_1|x_{1:T}) \prod_{t=2}^{T} p(z_t|z_{t-1}, x_{t:T}), \end{aligned} \tag{7}$$

where the second equation holds on using $z_t \perp x_{1:t-1}|z_{t-1}$.

Eq.(7) implies that the current state z_t depends on the previous state z_{t-1} and the input $x_{t:T}$. The dependency of z_t on future input $x_{t+1:T}$ makes posterior inference intractable. Our insight is that we instead replace $x_{t:T}$ with e_t, which contains the information from past, current and future, rather than inferring posterior distribution using $x_{t+1:T}$ directly.

The joint probability is factorizes as

$$\begin{aligned} &p(x'_{1:T}, d_{1:T}, e_{1:T}, z_{1:T}|x_{1:T}, z_0) \\ &= p(x'_{1:T}|d_{1:T}) p(d_{1:T}|z_{1:T}) q(z_{1:T}|e_{1:T}, z_0) q(e_{1:T}|x_{1:T}) \\ &= \prod_{t=1}^{T} p(x'_t|d_t) \cdot p(d_{1:T}|z_{1:T}) \prod_{t=1}^{T} q(z_t|z_{t-1}, e_t) \cdot q(e_{1:T}|x_{1:T}) \\ &= \prod_{t=1}^{T} p(x'_t|d_t) p(d_t|z_{1:T}) q(z_t|z_{t-1}, e_t) q(e_t|x_{1:T}), \end{aligned} \tag{8}$$

where z_0 is the initial state of z, $q(e_t|x_{1:T})$ and $p(d_t|z_{1:T})$ are expressed with TransformerBlock, and e_t and d_t are intermediate states of TransformerBlock in variational and generative network respectively.

This factorization means that we use the TransformerBlock's intermediate state to address the future dependency. Eq.(8) is implied by the graphical structure of the *TransAnomaly*(Fig. 2(c)).

In this section, we focus on the network architecture of *TransAnomaly*. *TransAnomaly* contains variational network and generative network. The structure of generative network is similar with variational network with different input. The diagram of variational network is shown as Figure 1 and the overall graphic of our model can be seen in Figure 2(c). We then introduce the architecture of *TransAnomaly*.

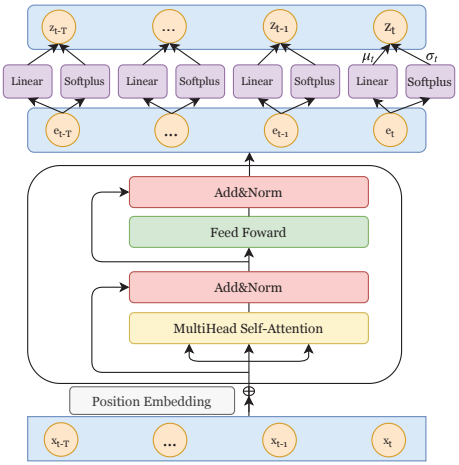


Figure 1: Variational Network in *TransAnomaly*

3.1 Generative Network

In the generative network, VAE is conditioned on the output d_t of the Transformer at every time step, which helps VAE to take the temporal structure of latent variable z into account. The output of Transformer acts as an internal memory of *TransAnomaly* which contains the global information from the input window $z_{1:T}$, *i.e.* all time steps are taken into account when we generate a sample x'_t.

The prior transition distribution of the latent random variable follows GTF, a nonlinear state space model, which is introduced in [9]. We use GTF to express the transition probability $p(z_t|z_{t-1})$.

$$z_t|z_{t-1} \sim \text{GTF}(z_{t-1}). \tag{9}$$

The form of GTF(z_{t-1}) can be formalized as follows:

$$\begin{aligned} z_t|z_{t-1} &\sim \text{GTF}(z_{t-1}) \\ g_t &= \text{MLP}(z_{t-1}, \text{ReLU}, \text{Sigmoid}) \\ h_t &= \text{MLP}(z_{t-1}, \text{ReLU}, I) \\ \mu(z_{t-1}) &= (1 - g_t) \odot (W z_{t-1} + b) + g_t \odot h_t \\ \sigma^2(z_{t-1}) &= \text{softplus}(W\text{ReLU}(h_t) + b) \end{aligned}$$

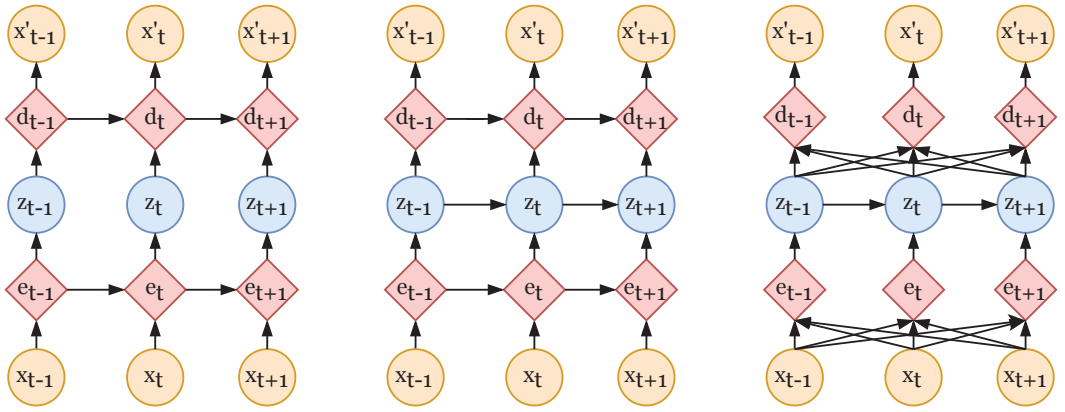
The generating distribution depends on z_t, formalized as

$$x_t|z_t \sim N(\mu_{\theta,t}(d_t), \sigma^2_{\theta,t}(d_t)), \tag{10}$$

where $\mu_{\theta,t}(d_t) = \text{Linear}(d_t)$ and $\sigma_{\theta,t}(d_t) = \text{softplus}(d_t)$, and Linear$(\cdot)$ and softplus$(\cdot)$ are parameterized by the neural network.

In generative network we have:

$$d_{1:T} = \text{Transformer}_\theta(z_{1:T}), \tag{11}$$



(a) LSTM-VAE (b) OmniAnomaly (c) *TransAnomaly*

Figure 2: Graphical models that are related to *TransAnomaly*. Diamond-shaped units are hidden state of temporal structure, including RNN and TransformerBlock, while circles are latent random variables of VAE. The arrows indicate the flow of information.

where $\text{Transformer}_\theta$ is a function stacked by N TransformerBlock, which capture temporal structure of $z_{1:T}$. d_t is determined by the information from the past as well as present and future. We can regard d_t as a high-level feature learned from a global view while autoregressive model can only look the past and current information. As we don't adopt recurrent structure, the model can handle the input of T time steps simultaneously Specifically, TransformerBlock contains the following part:

$$\begin{aligned}
\text{Input}^n_{Dec} &= \text{Output}^{n-1}_{Dec} \\
\text{A}^n_{Dec} &= \text{MultiHead}_{Dec}(\text{Input}^n_{Dec}, \text{Input}^n_{Dec}, \text{Input}^n_{Dec}) \\
\text{B}^n &= \text{LayerNorm}_{Dec}(\text{A}^n_{Dec} + \text{Input}^n_{Dec}) \\
\text{Output}^n_{Dec} &= \text{LayerNorm}_{Dec}(\text{FFN}_{Dec}(\text{B}^n_{Dec}) + \text{B}^n_{Dec})
\end{aligned}$$

It's necessary to make use of the order of sequential data. We adopt the same idea with original paper of Transformer [16] that the input of the model is constructed by summing the series and the position embedding which have the same dimension with z. The position embedding of x is set as follows:

$$\begin{aligned}
\text{PE}_{(pos,2i)} &= \text{Sin}(pos/10000^{2i/d_x}) \\
\text{PE}_{(pos,2i+1)} &= \text{Cos}(pos/10000^{2i/d_x}),
\end{aligned}$$

where $d_x = m$ is the dimension of x.

3.2 Variational Network

The variational distribution q_θ, or called approximate posterior, which conditioned on the output of Transformer follows the equation:

$$z_t|x_t \sim N(\mu_{\phi,t}(e_t), \sigma^2_{\phi,t}(e_t)), \tag{12}$$

where $\mu_{\phi,t}(d_t) = \text{Linear}(e_t)$ and $\sigma_{\phi,t}(e_t) = \text{softplus}(e_t)$, and $\text{Linear}(\cdot)$ and $\text{softplus}(\cdot)$ are expressed through a similar neural network.

$$\begin{aligned}
\text{Input}^n_{Enc} &= \text{Output}^{n-1}_{Enc} \\
\text{A}^n_{Enc} &= \text{MultiHead}_{Enc}(\text{Input}^n_{Enc}, \text{Input}^n_{Enc}, \text{Input}^n_{Enc}) \\
\text{B}^n &= \text{LayerNorm}_{Enc}(\text{A}^n_{Enc} + \text{Input}^n_{Enc}) \\
\text{Output}^n_{Enc} &= \text{LayerNorm}_{Enc}(\text{FFN}_{Enc}(\text{B}^n_{Enc}) + \text{B}^n_{Enc})
\end{aligned}$$

In variational network we have:

$$e_{1:T} = \text{Transformer}_\phi(x_{1:T}), \tag{13}$$

where Transformer_ϕ is a function which is similar with $\text{Transformer}_\theta$.

3.3 Optimization Objective and Anomaly Score

The overall architecture of the model follows VAE, which is trained by maximizing the ELBO. As mentioned above, we use Monte Carlo integration to estimate the ELBO as follows:

$$\begin{aligned}
\mathcal{L}(x_{1:T}) = \frac{1}{L}\sum_{l=1}^{L}\Big[&\log(p_\theta(x_{1:T}|z^{(l)}_{1:T}))+ \\
&\log(p_\theta(z^{(l)}_{1:T})) - \log(q_\phi(z^{(l)}_{1:T}|x_{1:T}))\Big],
\end{aligned} \tag{14}$$

where L is the sample size. We use Adam optimizer [7] to maximize the above loss (Eq.(14)).

Following the suggestion of [17], we use the reconstruction probability as our anomaly score, that is $score = \log(p_\theta(x_t|z_{1:T}))$. We determine a data point as an anomaly if the score is lower than a threshold. Here, we take the POT to choose the threshold automatically.

3.4 Online Detection

In the online detection stage, we determine whether a data point is anomalous using the trained TransAnomaly model. We consider the future dependency when training the model, whereas we do not know the future input when apply the model to online detection. Actually, we do not need to know the real future input. Considering future dependencies allows us to fit the data during the training, which helps us migrate the trained model to new data.

4 Experiments

In this section, we conduct extensive experiments to show the performance of our model.

Table 1: Statistics of The Public Benchmark Datasets

Dataset	Dimensions	Training set size	Testing set size	Anomaly ratio(%)
SMAP	25	135183	427617	13.13
MSL	55	58317	73729	10.72
SMD	38	708405	708420	4.16

4.1 Datasets and Performance Metrics

We employ three benchmark dataset: SMAP, MSL and KDDCUP99. We briefly describe the datasets below.

- **SMAP**: SMAP (Soil Moisture Active Passive satellite), a public datasets from NASA, is first introduced in [6].
- **MSL**: MSL (Mars Science Laboratory) is also a public datasets from NASA [6].
- **SMD**: SMD (Server Machine Dataset), a 5-week-long dataset collected from a large Internet company, is first introduced in [15]. SMD contains data from 28 machines, and for each of them, we need to conduct training and testing.

For these datasets, we set the window size as 100 for training and testing. Each segment is compressed and reconstructed by *TransAnomaly*, and the reconstruction probability is used as an indicator of anomaly. In our evaluation, we use Precision, Recall and F1-score to evaluate the performance of each model: $F1 = \frac{2\times \text{Precision}\times \text{Recall}}{\text{Precision}+\text{Recall}}$. As some methods have shown that how to choose anomaly threshold, we adopt the default strategy of the paper. Following by [15], we enumerate all possible thresholds to get $F1_{best}$, which indicates the theoretical capability of the model. This is also the focus of *TransAnomaly*.
Table 1 show the details of the datasets.

4.2 Baseline Methods

We compare *TransAnomaly* with 4 state-of-the-art methods for multivariate time series anomaly detection: DAGMM [18], LSTM-VAE [12] and OmniAnomaly [15] on three public real-world datasets. We also show the graphic of LSTM-VAE and OmniAnomaly in Figure 2.

- **DAGMM**: DAGMM does not consider the temporal information of the data points, which is crucial for multivariate time series.
- **LSTM-VAE**: LSTM-VAE combines LSTM and VAE but don't include the temporal dependence among stochastic variables.
- **OmniAnomaly**: OmniAnomaly adopts similar structure of SRNN [3], and the deterministic state only depends on the current input and don't consider future dependency.

All models provide specific methods for choosing anomaly thresholds and we try to use the original paper's settings for fair competition. The core model we want to compare is OmniAnomaly, a state-of-the-art model on these datasets, so we adopt the same automatic threshold selection strategy as OmniAnomaly.

Table 2: Training time per epoch of OmniAnomaly and Ours(minutes)

Methods	SMAP	MSL	SMD	Params $\times 10^4$
OmniAnomaly	48	11	87	160∼180
TransAnomaly	10	1.5	25	3∼4

Table 3: $F1_{best}$ of OmniAnomaly and Ours

Methods	SMAP	MSL	SMD	Total
OmniAnomaly	0.8535	0.9014	0.9620	0.9056
TransAnomaly	**0.9154**	**0.9218**	**0.9693**	**0.9355**

We conduct the experiment on the same platform. Table 2 shows the running time of OmniAnomaly and ours. Table 3 shows $F1_{best}$ of OmniAnomaly and ours. Experimental results demonstrate that our model not only improves the performance but also has lower training cost. Table 4 shows the precision, recall and F1 of models mentioned above and ours on three datasets, where the best score is shown in bold. Here F1 is calculated based on POT. We can see that LSTM-VAE performs poorly. This may be because it does not consider the temporal dependency in latent variables. DAGMM doesn't adopt temporal structure as the feature extractor, which is crucial for time series. Therefore, DAGMM does not perform well on SMAP and MSL. OmniAnomaly is a competitive method as it adds z-connection and makes posterior distribution more intractable. However, it doesn't consider the future dependency. In summary, *TransAnomaly* performs better than other baseline method on all datasets. In *TransAnomaly*, we introduce future dependency and GTF, which is beneficial for the improvement of the result. Besides, our model adopts Transformer as the feature extractor, which has been shown to have powerful ability for sequential data modeling.

4.3 Ablation Study

Here we look at the importance of each module what we introduced in this paper to demonstrate the importance of *TransAnomaly*'s modules. Moreover, we take the following *TransAnomaly* variants into account.
All the variants are listed as follows:

- **TransAnomaly**$_{\text{SSM}}$: We replace GTF with SSM.
- **TransAnomaly**$_{\text{IGD}}$: No z-connection, which means all the latent variables follow normal distribution.
- **TransAnomaly**$_{\text{w/o mask}}$: We add mask on TransformerBlock, which doesn't consider the future dependency, that means the current state only depends on the previous input.

We evaluate all variants on the datasets. Table 3 shows the performance of *TransAnomaly* and its variants.
From Figure 3 we can see that adding future dependency is critical for learning global representations. TransAnomaly is slightly better than TransAnomaly$_{\text{SSM}}$.

Table 4: Performance of *TransAnomaly* and 4 baseline approaches.

Methods	SMAP			MSL			SMD		
	P	R	F1	P	R	F1	P	R	F1
DAGMM [18]	0.5845	0.9058	0.7105	0.5412	0.9934	0.7007	0.5951	0.8782	0.7094
LSTM-VAE[12]	0.8551	0.6366	0.7298	0.5257	0.9546	0.6780	0.7922	0.7075	0.7842
OmniAnomaly[15]	0.7416	0.9776	0.8434	0.8867	0.9117	0.8989	0.8334	0.9449	0.8857
TransAnomaly(Ours)	0.7998	0.9752	**0.8788**	0.9090	0.9149	**0.9117**	0.8415	0.9502	**0.8931**

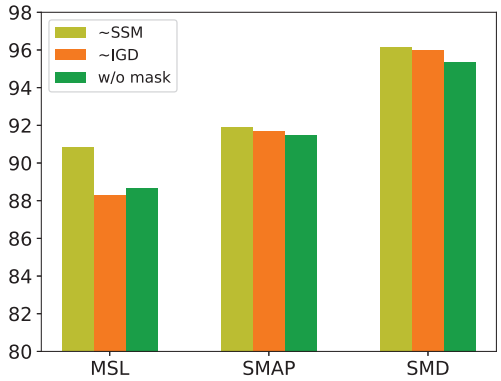


Figure 3: $F1_{best}$ of TransAnomaly and the variants(%).

It could be explained by the fact that GTF can capture more complex dependency on latent variables. Both TransAnomaly and $\text{TransAnomaly}_{\text{SSM}}$ are obviously better than $\text{TransAnomaly}_{\text{IGD}}$, which means that connecting latent variables by state space model is effective. This has also been proved in [3]. On different datasets, the performance of each module is slightly different. It can be explained by the different characteristics of different datasets. For example, in SMAP and MSL, there are substantial discrete variables. Even without adding future dependency, our model performs better than other baseline, which can be explained by the feature extractor's ability.

4.4 CONCLUSION

Anomaly detection for multivariate time series has great significance. We propose *TransAnomaly*, a novel unsupervised parallelizable model integrated multi-head self-attention and GTF. We give a probabilistic explanation of *TransAnomaly*, which provides reasonable result for later work. Extensive empirical studies on several real-world datasets demonstrate that *TransAnomaly* compares favorably to state of the art and reduces the training cost.

REFERENCES

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

[2] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. In *NIPS*, pages 2980–2988, 2015.

[3] M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther. Sequential neural models with stochastic layers. In *NIPS*, pages 2199–2207, 2016.

[4] V. Hautamaki, I. Karkkainen, and P. Franti. Outlier detection using k-nearest neighbour graph. In *ICPR*, volume 3, pages 430–433, 2004.

[5] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[6] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *SIGKDD*, pages 387–395, 2018.

[7] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv*, 2014.

[8] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv*, 2013.

[9] R. G. Krishnan, U. Shalit, and D. Sontag. Structured inference networks for nonlinear state space models. In *AAAI*, 2017.

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

[11] L. M. Manevitz and M. Yousef. One-class svms for document classification. *JMLR*, 2(Dec):139–154, 2001.

[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*, 3(3):1544–1551, 2018.

[13] C. Robert and G. Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

[14] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouet. Anomaly detection in streams with extreme value theory. In *SIGKDD*, pages 1067–1075. ACM, 2017.

[15] 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 *SIGKDD*, pages 2828–2837. ACM, 2019.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[17] 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 *WWW*, pages 187–196, 2018.

[18] 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 *ICLR*, 2018.