SDFVAE: Static and Dynamic Factorized VAE for Anomaly Detection of Multivariate CDN KPIs

Liang Dai
Institute of Information Engineering, Chinese Academy of Sciences
Beijing, China
dailiang@iie.ac.cn

Tao Lin*
State Key Laboratory of Media Convergence and Communication, Communication University of China
Beijing, China
lintao@cuc.edu.cn

Chang Liu
Institute of Information Engineering, Chinese Academy of Sciences
Beijing, China
liuchang2@iie.ac.cn

Bo Jiang
Shanghai Jiao Tong University
Shanghai, China
bjiang@sjtu.edu.cn

Yanwei Liu, Zhen Xu
Institute of Information Engineering, Chinese Academy of Sciences
Beijing, China
{liuyanwei,xuzhen}@iie.ac.cn

Zhi-Li Zhang
University of Minnesota
MN, USA
zhang089@umn.edu

ABSTRACT

Content Delivery Networks (CDNs) are critical for providing good user experience of cloud services. CDN providers typically collect various multivariate Key Performance Indicators (KPIs) time series to monitor and diagnose system performance. State-of-the-art anomaly detection methods mostly use deep learning to extract the normal patterns of data, due to its superior performance. However, KPI data usually exhibit non-additive Gaussian noise, which makes it difficult for deep learning models to learn the normal patterns, resulting in degraded performance in anomaly detection. In this paper, we propose a robust and noise-resilient anomaly detection mechanism using multivariate KPIs. Our key insight is that different KPIs are constrained by certain time-invariant characteristics of the underlying system, and that explicitly modelling such invariance may help resist noise in the data. We thus propose a novel anomaly detection method called SDFVAE, short for Static and Dynamic Factorized VAE, that learns the representations of KPIs by explicitly factorizing the latent variables into dynamic and static parts. Extensive experiments using real-world data show that SDFVAE achieves a F1-score ranging from 0.92 to 0.99 on both regular and noisy dataset, outperforming state-of-the-art methods by a large margin.

CCS CONCEPTS

• **Computing methodologies** → **Anomaly detection**.

KEYWORDS

Multivariate Anomaly Detection, Content Delivery Network, Static and Dynamic Factorization, Latent Variable Model

*Corresponding author.

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450013

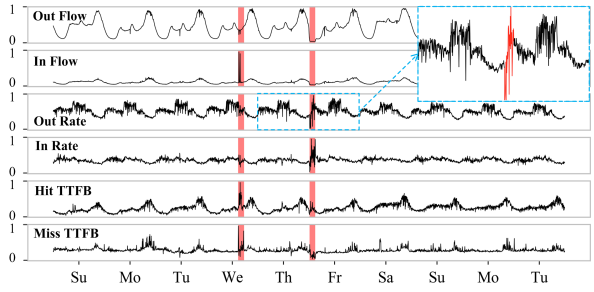
ACM Reference Format:
Liang Dai, Tao Lin, Chang Liu, Bo Jiang, Yanwei Liu, Zhen Xu, and Zhi-Li Zhang. 2021. SDFVAE: Static and Dynamic Factorized VAE for Anomaly Detection of Multivariate CDN KPIs. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3442381.3450013

1 INTRODUCTION

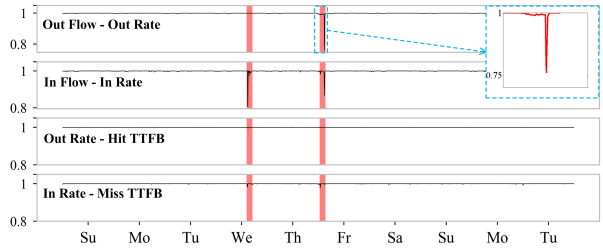
Content Delivery Networks (CDNs) play a critical role in today's content delivery ecosystem, providing good user experiences with reduced latency for various websites and cloud services. With growing complexity and scale, performance issues are inevitable. Therefore CDN operators measure and collect various Key Performance Indicators (KPIs) such as traffic volume, cache hit ratio and server response delay, to conduct service quality management. However, due to massive volume, noise in the data and lack of ground truth, mining such time series KPI data for effective anomaly detection is still a challenging task, especially to rapidly respond to performance issues before they cause critical performance degradation.

In general, unexpected instances are usually considered as anomalies if they show significant abnormal behaviors. Earlier studies [5, 6, 23, 35] started with univariate KPI anomaly detection, i.e., using only a single type of KPI data. More recent studies have shifted to multivariate KPIs anomaly detection. Utilizing multiple types of KPI data streams not only avoids training and maintaining a large number of individual models for each metric, but also helps increase the effectiveness of anomaly detection, as an incident typically tends to produce anomalies in multiple KPIs [2, 21, 31, 39]. Meanwhile, deep learning based anomaly detection method, or deep anomaly detection in short, has been widely concerned in recent years due to its huge advantages in learning expressive representations of complex and massive data. The basic idea of deep anomaly detection is to model the normal patterns of time series, considering an anomaly or outlier often behaves differently from the normal data. The larger an observation deviates from the normal patterns, the more likely it is considered as an anomaly.

However, the performance of deep anomaly detection is vulnerable to noise presented in multivariate KPIs since the models are also trained to learn the distribution of noise in addition to normal data,



(a) Time-varying characteristics



(b) Time-invariant characteristics exemplified by correlation analysis

Figure 1: 10-day real world multivariate CDN KPIs analysis

thus usually suffer from the problem of over-fitting [25]. Unfortunately, noise is not unusual in real multivariate data due to volatile system environment and fine-granularity data. As shown in Fig. 1(a), it is observed that the multivariate time series data of CDN often exhibit non-additive Gaussian noises, e.g., multiplicative Gaussian[1], and thus present a complex data distribution which makes it difficult to model. Although various anomaly detection methods on multivariate KPIs have been proposed [16, 21, 24, 31, 39, 42], however, there are few studies on anomaly detection of time series data with noise. As shown in Section 4, the state-of-the-art approaches [24, 31, 39, 42] do not perform well on noisy data.

In this paper, our goal is to design a robust and noise-resilient anomaly detection method for multivariate time series data. To this end, we start with a thorough analysis of a real-world multivariate CDN KPI dataset to gain a deeper insight of its characteristics. We underscore two domain-specific observations as follows.

- Observation 1: Except for obvious *time-varying* or dynamic characteristics such as periodicity on individual KPI, some KPI pairs are highly correlated with each other[2] , as shown in Fig. 1(b). More importantly, the correlations of these KPI pairs present *time-invariant* characteristic, namely, they remain unchanged for most of the time except for the occurrence of anomalies.
- Observation 2: the characteristic of time-invariant exists not only in regular KPI pairs, but also in noisy KPI pairs, e.g., the KPI pair of Out_Rate and Hit_ttfb in Fig. 1(b). It indicates that noise has little impact on the time-invariant characteristic (See Section 2.1 for more details).

To some extent, the observed time-invariant characteristic reflects the intrinsic stability of a real network system like CDN in which different KPIs are constrained by certain time-invariant characteristic of the underlying system. Such time-invariance, which has not been fully utilized by previous studies, can be considered as a hidden representation of the normal patterns in multivariate KPIs. Exploiting the time-invariant characteristic of multivariate data to build a noisy-resilient anomaly detection system can not only capture more expressive representations of normal data pattern, but also help resist noise in the data.

Motivated by above observations, we propose a novel anomaly detection method for multivariate KPIs, named **S**tatic and **D**ynamic **F**actorized **V**ariational **A**uto**E**ncoder (**SDFVAE**). The major challenge is how to *explicitly* learn the representations of both time-varying and time-invariant characteristics hidden in the multivariate KPIs. To this end, a novel representation model is proposed to factorize the latent space into two separate latent variables, namely *static* and *dynamic*, which corresponds to time-varying and time-invariant characteristics of multivariate KPIs respectively. Specifically, we utilize a Bi-directional Long Short-Term Memory (BiLSTM) based inference network to capture the static latent representations, and a recurrent Variational AutoEncoder (VAE) inference network to learn the dynamic latent representations. The main contributions of our work are summarized as follows:

- Through a careful analysis of a real-world CDN KPI dataset, we find that multivariate KPIs exhibit hidden time-invariant characteristics and modeling such time-invariance may help resist the noise in the data.
- We propose a noisy-resilient anomaly detection method based on static and dynamic factorized VAE named SDFVAE, which is capable of explicitly learning the representations of time-invariant characteristics of multivariate KPIs, in addition to the time-varying characteristics.
- We conduct extensive experiments employing both real-world dataset collected from a top CDN provider in China and a public dataset. The results demonstrate that SDFVAE achieves a F1-score ranging from 0.92 to 0.99, which significantly outperforms state-of-the-art baselines. For the convenience of reproduction, we have released our source codes at https://github.com/dlagul/SDFVAE.

2 PRELIMINARY

2.1 Understanding Time-invariance: an Example Analysis

To further understand the characteristic of time-invariance, we will first briefly introduce the infrastructure of CDN. Then we will illustrate the rationality behind time-invariant characteristic.

As shown in Fig. 2, a typical CDN works as follows. Internet users will first query the scheduling center for the most suitable CDN edge node. Then HTTP requests of the Internet user will be routed to the edge node via the front-haul network, such as cellular network and residential network. Once a cache hit occurs, the edge node will return the requested content object directly; else

[1]The noise degree is usually related with the KPI values, i.e., the larger the value the higher degree the noise and vice versa.

[2]We employ Local Correlation Score (LCS) [26] to examine how the correlations change over time. The scores of LCS range from 0 to 1 and the higher the score, the stronger the correlation.

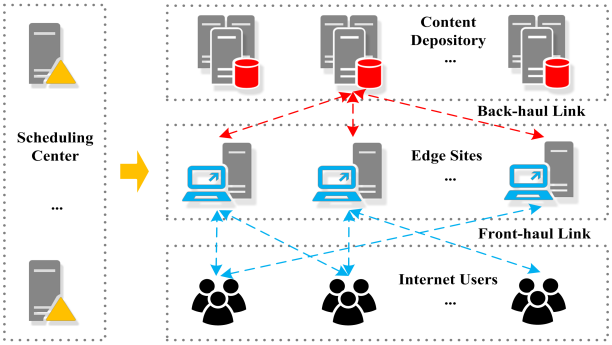


Figure 2: CDN Infrastructure

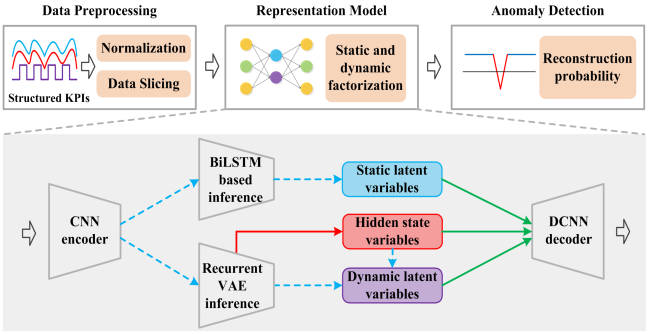


Figure 3: The framework of SDFVAE

it needs to retrieve the content from upper-layer content depository or neighboring nodes via back-haul network. In order to detect and diagnose the performance problem of CDN services, a common practice for CDN operators is to monitor multivariate KPIs data collected from CDN edge nodes.

As mentioned earlier, it is observed that CDN multivariate KPIs exhibit both time-varying and time-invariant characteristics. Obviously, time-varying characteristic comes from dynamic external environment, e.g., the variation of Internet user requests and volatile condition of network. However, time-invariant characteristic mainly attributes to intrinsic stability of a network system, which is determined by the limited capacity of servers (network bandwidth, I/O throughput, etc.) and the well-defined internal interfaces between different components. For instance, due to capacity limitation of CDN edge server, the increase of Out_Flow or out-bound traffic will lead to a higher load of the server, resulting in the increase of average TTFB (Time To First Byte)[3] of "Hit" requests, or Hit_TTFB. Thus it further reduces the average out-bound download bitrates of Http sessions (Out_Rate) and vice versa. As a result, as shown in Fig. 1(b), some KPI pairs such as (Out_Flow, Out_Rate) and (Out_Rate, Hit_ttfb) are highly correlated and such correlation remains unchanged for most of time. Meanwhile, thanks to the stable but strong correlation between different KPIs, when one KPI such as Hit_ttfb tends to be turbulent, it will lead to the same turbulence in its highly related KPI, e.g., Out_Rate. Therefore, although each KPI is noisy, the correlation of KPI pair (Out_Rate, Hit_ttfb) seems to be stable and smooth as shown in Fig. 1(b).

However, once a system anomaly occurs, the intrinsic stability of the system is violated and thus the time-invariant characteristic will not be respected. For instance, for the second anomaly highlighted in red in Fig. 1(b), it is observed that the correlation between the KPIs of Out_Flow and Out_Rate becomes weak.

In summary, this example illustrates that time-invariance reflects the intrinsic stability of a complex system, and thus is a critical factor to capture normal patterns of multivariate KPIs. Therefore, except for the well-known time-varying characteristic, leveraging the time-invariance has the potential to learn robust representations of multivariate noisy data and restrain the impact of noise.

[3]TTFB is a measure to indicate the processing delay between receiving an Http request and sending the first byte of the reply at a CDN server.

2.2 Variational AutoEncoder

Since SDFVAE works on VAE, here we give a brief introduction. VAE is a deep generative model aiming to model the relationship between latent variable $\boldsymbol{z}$ and observed variable $\boldsymbol{x}$ [20, 28]. Considering the joint probability distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$, it specifies a latent variable model parameterized by $\boldsymbol{\theta}$ over a set of observed variables $\boldsymbol{x}$ and latent variables $\boldsymbol{z}$, with the goal of maximizing the marginal log-likelihood of $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \log \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p_{\boldsymbol{\theta}}(\boldsymbol{z}) d\boldsymbol{z}$. However, it is often intractable for complex generative models. VAE provides a solution via introducing an inference model $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ parameterized by $\boldsymbol{\phi}$ to approximate the true posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$. Thus, the problem is transformed to an optimization problem with maximizing the **ELBO** (**E**vidence **L**ower **BO**und) as follows.

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

where KL denotes the Kullback-Leibler divergence. Both generative model $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ and inference model $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ are constructed by deep neural networks and trained jointly by applying backpropagation based on the reparameterization trick [20].

3 PROPOSED METHOD

In this section, we first present the problem statement and then the framework of SDFVAE, followed by the details of our design including data preprocessing, representation model and anomaly detection.

3.1 Problem Statement

We define the multivariate CDN KPIs as $\boldsymbol{k} = \{\boldsymbol{k}_1, \boldsymbol{k}_2, ..., \boldsymbol{k}_N\}$, where N is the durations of $\boldsymbol{k}$, and each observation $\boldsymbol{k}_\tau \in \boldsymbol{R}^n$ is a n-dimensional vector at time τ ($\tau \leq N$), and $\boldsymbol{k} \in \boldsymbol{R}^{n \times N}$, where n denotes the number of KPIs. The problem of anomaly detection on multivariate KPIs can be defined as deciding whether an observation at certain time step τ ($\boldsymbol{k}_\tau$) is anomalous or not.

3.2 Overview of the Framework

As shown in Fig. 3, here we briefly introduce the overall framework of SDFVAE. First, in order to obtain time-invariant and time-varying related information, the original multivariate KPIs data is normalized and then pre-processed via introducing sliding windows (Section 3.3). Second, a novel VAE based representation model is proposed to factorize the latent space into static latent variables and

dynamic latent variables in order to represent both time-varying and time-invariant characteristics of multivariate KPIs (Section 3.4). Finally, anomaly detection is conducted based on the reconstruction probability (Section 3.5).

The detailed neural network architecture of the representation model is illustrated in the bottom of Fig. 3. First, we use a CNN (Convolutional Neural Network) based encoder to capture complex information of correlations hidden in multivariate time series data. Second, a BiLSTM based inference network is employed to learn the static latent representations. Third, we propose a recurrent VAE inference network to learn the dynamic latent variables. Forth, the sampled latent variables (static and dynamic) as well as the hidden state variable of recurrent VAE are concatenated and fed into a DCNN (DeConvolutional Neural Network) based decoder to obtain the mean and standard deviation of generated variables.

3.3 Data Preprocessing

Motivated from previous studies on video processing [10, 22, 34] where the sequence of video frames contains not only time-varying (e.g., motion) but also time-invariant (e.g., object) information, we pre-process the normalized KPIs data $\boldsymbol{k}$ as shown in Fig. 4. Similar with a video frame, each $\boldsymbol{x}_t$ denotes an *observed variable*. A sequence containing T observed variables is denoted as $\boldsymbol{x}_{1:T}^{(\tau)} = [\boldsymbol{x}_1^{(\tau)}, \ldots, \boldsymbol{x}_t^{(\tau)}, ..., \boldsymbol{x}_T^{(\tau)}]$, where $\boldsymbol{x}_T^{(\tau)} = [\boldsymbol{k}_{\tau-w+1}, \ldots, \boldsymbol{k}_{\tau-1}, \boldsymbol{k}_\tau]$, τ corresponds to the time step of the observation $\boldsymbol{k}$, w is the length of observed variable, l denotes the strides between two consecutive observed variables, thus $\boldsymbol{x}_{1:T}^{(\tau)} \in \boldsymbol{R}^{n\times w\times T}$. Then we obtain a pre-processed multivariate KPIs dataset denoted as $\mathcal{D}(\boldsymbol{x}_{1:T}) = \{\boldsymbol{x}_{1:T}^{(\tau)}\}_{\tau=w+(T-1)\times l}^{N}$ which consists of $N - w + (T-1)\times l + 1$ pre-processed sequences of observed variables through sliding window. The data distribution of these sequences can be denoted as $p_{\mathcal{D}}(\boldsymbol{x}_{1:T})$. For simplicity, we drop the index τ, and the input of our model is a sequence of observed variables which can be denoted as $\boldsymbol{x}_{1:T}$ or $\boldsymbol{x}_{\leq T}$. In the following of this paper, the term of sequence of observed variables is interchangeable with *observed sequence*.

3.4 Representation Model

In order to explicitly model the observed sequence $\boldsymbol{x}_{1:T}$ with both time-invariant and time-varying features, we present a latent variable model with two separated latent spaces: static and dynamic. In this section, we mainly introduce the generative model and the inference model involved, followed by the objective function used for learning.

3.4.1 Generation. We formulate a generative process [15] for the observed sequence by assuming it can be generated from both the static and dynamic latent variables $\boldsymbol{s}$ and $\boldsymbol{d}$ as follows. First, the static latent variables $\boldsymbol{s}$ representing the factors remaining unchanged over time are sampled from a time-independent prior distribution $p_{\boldsymbol{\theta}}(\boldsymbol{s})$. Then the dynamic latent variables $\boldsymbol{d}_t$ at each time step are sampled from the time-dependent prior distribution $p_{\boldsymbol{\theta}}(\boldsymbol{d}_t|\boldsymbol{d}_{<t})$ as illustrated in Fig. 5(a) and represent the factors indicating how the current observed variable changes over the previous observed variables $\boldsymbol{x}_{<t}$; Last, $\boldsymbol{x}_t$ are generated from the conditional distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{d}_{\leq t}, \boldsymbol{s})$.

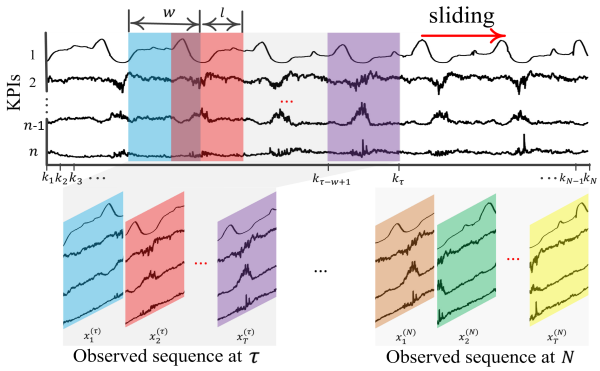


Figure 4: Multivariate-KPI data preprocessing

We implement the above generative process in the following way. We first place time-dependent prior on dynamic latent variables by employing a RNN (Recurrent Neural Network). As illustrated in Fig. 5(a), $\boldsymbol{d}_t$ is conditioned on the hidden state variable $\boldsymbol{h}_{t-1}^p$, which is updated using the recurrence equation

$$\boldsymbol{h}_t^p = r_p(\boldsymbol{h}_{t-1}^p, \boldsymbol{d}_t) \tag{2}$$

where $r_p(.)$ is the deterministic non-linear transition function of RNN. Different from [31] which utilizes time-independent prior of dynamic latent variables, in our design each $\boldsymbol{d}_t$ is conditioned on $\boldsymbol{d}_{<t}$ to introduce time-dependent prior of dynamic latent variables , which helps improve the performance of representation model [8]. Next, inspired by [8, 12], we introduce the hidden state variables of an RNN. A series of VAEs are stacked at each time step and linked by these state variables to construct the generative model of the sequence. In this way, each VAE at time step t is conditioned on the state variable $\boldsymbol{h}_{t-1}$. This kind of structure is denoted as *recurrent VAE*. As shown in Fig. 5(b), except for latent variables $\boldsymbol{d}_t$ and $\boldsymbol{s}$, $\boldsymbol{x}_t$ is also conditioned on $\boldsymbol{h}_{t-1}$. Besides, the recurrence equation to update state variables is illustrated in Fig. 5(c) and formulated by

$$\boldsymbol{h}_t = r(\boldsymbol{h}_{t-1}, \boldsymbol{d}_t, \boldsymbol{x}_t) \tag{3}$$

where $r(.)$ is the deterministic non-linear transition function. Therefore, each generated variable $\boldsymbol{x}_t$ is conditioned on $\boldsymbol{x}_{<t}$, $\boldsymbol{d}_{\leq t}$ and $\boldsymbol{s}$. In this way, our generative model results in the factorization:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}, \boldsymbol{s}, \boldsymbol{d}_{1:T}) = p_{\boldsymbol{\theta}}(\boldsymbol{s}) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{d}_{\leq t}, \boldsymbol{s}) p_{\boldsymbol{\theta}}(\boldsymbol{d}_t|\boldsymbol{d}_{<t}) \tag{4}$$

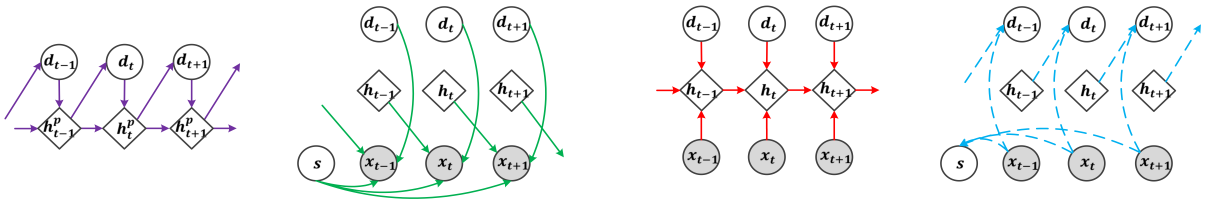
Specifically, each of the RHS (Right-Hand Side) term is formulated as follows:

$$p_{\boldsymbol{\theta}}(\boldsymbol{s}) = \mathcal{N}(\boldsymbol{s}|\boldsymbol{0}, \boldsymbol{I}) \tag{5}$$

$$p_{\boldsymbol{\theta}}(\boldsymbol{d}_t|\boldsymbol{d}_{<t}) = \mathcal{N}(\boldsymbol{d}_t|g_{\mu_d}(\boldsymbol{h}_{t-1}^p), diag(g_{\sigma_d^2}(\boldsymbol{h}_{t-1}^p)) \tag{6}$$

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{d}_{\leq t}, \boldsymbol{s}) = \mathcal{N}(\boldsymbol{x}_t|f_{\mu_x}(\boldsymbol{h}_{t-1}, \boldsymbol{d}_t, \boldsymbol{s}), diag(f_{\sigma_x^2}(\boldsymbol{h}_{t-1}, \boldsymbol{d}_t, \boldsymbol{s}))) \tag{7}$$

where the prior over the static latent variables $\boldsymbol{s}$ is a standard multivariate Gaussian distribution and the prior over the dynamic latent variables $\boldsymbol{d}_t$ is a diagonal multivariate Gaussian, whose mean and variance are parameterized by neural networks $g_{\mu_d}(.)$ and $g_{\sigma_d^2}(.)$



(a) Prior of latent variables $\boldsymbol{d}$ (b) Generative model (c) Recurrent structure (d) Inference model

Figure 5: Graphical models of each component of SDFVAE: Circles denote stochastic variables while diamond-shaped units are used for deterministic variables, shaded nodes denote observed variables. Solid arrows in purple represent the recurrence equation and time-dependent prior of dynamic latent variables while the red denote the recurrence equation of state variables in recurrent VAE. Green solid arrows represent generative model while blue dashed arrows denote the inference network.

with input $\boldsymbol{h}^p_{t-1}$. Moreover, the generative distribution of the observed variable $\boldsymbol{x}_t$ is also a diagonal multivariate Gaussian, whose mean and variance are parameterized by neural networks $f_{\mu_x}(.)$ and $f_{\sigma_x^2}(.)$ with input $\boldsymbol{h}_{t-1}, \boldsymbol{d}_t$ and $\boldsymbol{s}$. Specifically, considering LSTM (Long Short-term Memory) is a special gated RNN which is able to learn the long-term dependence in a sequence, we first employ a LSTM as $r_p(.)$, followed by two MLPs (Multi-Layer Perceptrons) to construct $g_{\mu_d}(.)$ and $g_{\sigma_d^2}(.)$ respectively. Besides, since DCNN performs well in spatial data restoration [36], e.g., image, here two DCNNs are used to approximate $f_{\mu_x}(.)$ and $f_{\sigma_x^2}(.)$ respectively as shown in Fig. 6. Note that all parameters in generative models are denoted as $\boldsymbol{\theta}$.

The key difference between SDFVAE and the models with only dynamic latent space [8, 31] is that these models only consider the factors that indicate how the current changes over the former while ignores the factors that remain unchanged over time. Accordingly, our separated latent variable model helps learn richer representations to capture the normal patterns of multivariate KPIs and provides more information to reconstruct them.

3.4.2 Inference. The goal of inference here is to get the full posterior over the static and dynamic latent variables $p_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{d}_{1:T}|\boldsymbol{x}_{1:T})$, however, this full posterior is always intractable. Thus, we apply variational inference method to get an approximate one. To this end, as shown in Fig. 5(d), the static latent variables $\boldsymbol{s}$ are time-independent and conditioned on the entire observed sequence which implies that its approximate posterior will be a function of $\boldsymbol{x}_{1:T}$. Further, as demonstrated in section 3.4.1, since each VAE at time step t is conditioned on the state variable $\boldsymbol{h}_{t-1}$, except for observed variable $\boldsymbol{x}_t$, each dynamic latent variable $\boldsymbol{d}_t$ is also conditioned on $\boldsymbol{h}_{t-1}$. Accordingly, $\boldsymbol{d}_t$ is conditioned on $\boldsymbol{d}_{<t}$ and $\boldsymbol{x}_{\leq t}$ due to $\boldsymbol{h}_t = r(\boldsymbol{h}_{t-1}, \boldsymbol{d}_t, \boldsymbol{x}_t)$. In this way, we observe that it results in a fully-factorized variational distribution:

$$q_{\boldsymbol{\phi}}(\boldsymbol{s}, \boldsymbol{d}_{1:T}|\boldsymbol{x}_{1:T}) = q_{\boldsymbol{\phi}}(\boldsymbol{s}|\boldsymbol{x}_{\leq T}) \prod_{t=1}^{T} q_{\boldsymbol{\phi}}(\boldsymbol{d}_t|\boldsymbol{d}_{<t}, \boldsymbol{x}_{\leq t}) \tag{8}$$

We formulate each of the RHS term as follows:

$$q_{\boldsymbol{\phi}}(\boldsymbol{s}|\boldsymbol{x}_{\leq T}) = \mathcal{N}(\boldsymbol{s}|\varphi_{\mu_s}(\boldsymbol{x}_{1:T}), diag(\varphi_{\sigma_s^2}(\boldsymbol{x}_{1:T}))) \tag{9}$$

$$q_{\boldsymbol{\phi}}(\boldsymbol{d}_t|\boldsymbol{d}_{<t}, \boldsymbol{x}_{\leq t}) = \mathcal{N}(\boldsymbol{d}_t|\psi_{\mu_d}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t), diag(\psi_{\sigma_d^2}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t))) \tag{10}$$

where the posteriors over $\boldsymbol{s}$ and each $\boldsymbol{d}_t$ are all diagonal multivariate Gaussian distributions. As shown in Fig. 6, since CNN is able to extract spatial features, e.g., correlations, from data with convolution structures, we employ a CNN to extract the information of correlations and get a summarized fixed-dimension vector which is denoted as $\hat{\boldsymbol{x}}_{1:T}$. Then, consider that the latent variables $\boldsymbol{s}$ is conditioned on the entire observed sequence $\boldsymbol{x}_{1:T}$ and that BiLSTM [13] helps capture context information of a sequence. Thus, similar with [22], we utilize a BiLSTM, followed by two MLPs taking the forward and backward hidden states of BiLSTM as input to approximate $\varphi_{\mu_s}(.)$, $\varphi_{\sigma_s^2}(.)$. Last, a LSTM is employed as $r(.)$ and we use another two MLPs to construct $\psi_{\mu_d}(.)$ and $\psi_{\sigma_d^2}(.)$ respectively. The parameters of these neural networks are denoted as $\boldsymbol{\phi}$.

3.4.3 Learning. As the usual strategy of variational inference, the optimization of our model can be achieved by maximizing the corresponding **ELBO**, which can be expressed as

$$\mathcal{L}(\boldsymbol{x}_{1:T}; \boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{s}, \boldsymbol{d}_{\leq T}|\boldsymbol{x}_{\leq T})}\Big[\sum_{t=1}^{T} \big[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{s}, \boldsymbol{d}_{\leq t}) - \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{d}_t|\boldsymbol{d}_{<t}, \boldsymbol{x}_{\leq t})||p_{\boldsymbol{\theta}}(\boldsymbol{d}_t|\boldsymbol{d}_{\leq t}))\big] - \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{s}|\boldsymbol{x}_{\leq T})||p_{\boldsymbol{\theta}}(\boldsymbol{s}))\Big] \tag{11}$$

where KL denotes the Kullback-Leibler divergence. Therefore, we train the generative and inference models jointly with

$$\arg\max_{\boldsymbol{\theta}, \boldsymbol{\phi}} \mathbb{E}_{p_{\mathcal{D}}(\boldsymbol{x}_{1:T})}[\mathcal{L}(\boldsymbol{x}_{1:T}; \boldsymbol{\phi}, \boldsymbol{\theta})] \tag{12}$$

We employ Adam optimizer [19] during the training and utilise the reparameterization trick [20] in our model. The complete procedure is given in Algorithm 1.

3.5 Anomaly Detection

We apply the likelihood to determine whether an observed variable is anomalous or not [1, 27, 35]. The log-likelihood $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{s}, \boldsymbol{d}_{\leq t})$ denotes the reconstruction probability of each observed variable $\boldsymbol{x}_t$, consequently, $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_T^{(\tau)}|\boldsymbol{x}_{<T}^{(\tau)}, \boldsymbol{s}, \boldsymbol{d}_{\leq T})$ is employed to evaluate the reconstruction probability of $\boldsymbol{x}_T^{(\tau)} = [\boldsymbol{k}_{\tau-w+1}, ..., \boldsymbol{k}_{\tau-1}, \boldsymbol{k}_\tau] \in \boldsymbol{R}^{n\times w}$. However, considering the real-time requirement for anomaly detection, we only focus on the reconstruction probability of $\boldsymbol{k}_\tau \in \boldsymbol{R}^n$. Since the generative distribution (or known as likelihood) is a diagonal Gaussian, the log-likelihood can be factorized as $\sum_{i=\tau-w+1}^{\tau} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{k}_i|\boldsymbol{x}_{<T}^{(\tau)}, \boldsymbol{s}, \boldsymbol{d}_{\leq T})\right]$. Therefore, anomaly score of $\boldsymbol{k}_\tau$ is denoted as $S_\tau = \log p_{\boldsymbol{\theta}}(\boldsymbol{k}_\tau|\boldsymbol{x}_{<T}^{(\tau)}, \boldsymbol{s}, \boldsymbol{d}_{\leq T})$. Note that the lower

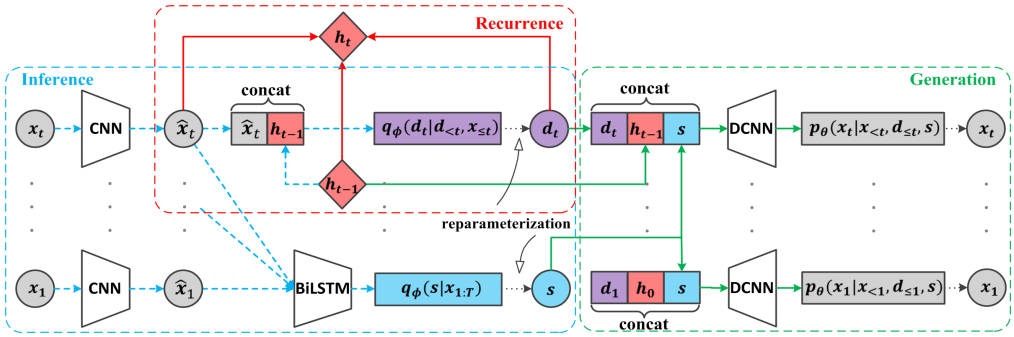


Figure 6: Detailed neural network architecture of SDFVAE. Blue dash arrows denote the inference network, solid arrows in green represent generative model while the red indicate the recurrence equation of state variables in recurrent VAE and black dotted arrows represent the sampling process using the reparameterization trick. Rectangles in purple and blue denote the fully connected layers predicting the mean and log variance of dynamic and static latent variables respectively.

Algorithm 1: SDFVAE training

input : The pre-processed KPIs training dataset $\mathcal{D}(\boldsymbol{x}_{1:T})$
output : The trained networks r_p, r, f, g, φ and ψ

Initialize the state variables $\boldsymbol{h}_0^p$, $\boldsymbol{h}_0$ and the network parameters $\boldsymbol{\theta}, \boldsymbol{\phi}$
repeat
  /* Sample minibatch samples from the training dataset */
  $\boldsymbol{x}_{1:T} \sim p_{\mathcal{D}}(\boldsymbol{x}_{1:T})$
  /* KL over $\boldsymbol{s}$ using Eq. 9 and Eq. 5 */
  $\mathcal{L}_{kl\_s} \leftarrow \mathrm{KL}(\mathcal{N}(\varphi_{\mu_s}(\boldsymbol{x}_{1:T}), diag(\varphi_{\sigma_s^2}(\boldsymbol{x}_{1:T}))) \,||\, \mathcal{N}(0, I))$
  /* Reparameterization trick based on Eq. 9 */
  Obtain $\epsilon \sim \mathcal{N}(0, I)$, set
  $\boldsymbol{s} \leftarrow \epsilon \odot diag(\varphi_{\sigma_s^2}(\boldsymbol{x}_{1:T})) + \varphi_{\mu_s}(\boldsymbol{x}_{1:T})$
  $\mathcal{L}_{kl\_d} \leftarrow 0, \mathcal{L}_{llh} \leftarrow 0$
  for $t = 1$ to T **do**
    /* KL over $\boldsymbol{d}_t$ using Eq. 10 and Eq. 6 */
    $\mathcal{L}_{kl\_d} \leftarrow \mathcal{L}_{kl\_d} +$
    $\mathrm{KL}(\mathcal{N}(\psi_{\mu_d}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t), diag(\psi_{\sigma_d^2}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t))) \,||$
    $\mathcal{N}(g_{\mu_d}(\boldsymbol{h}_{t-1}^p), diag(g_{\sigma_d^2}(\boldsymbol{h}_{t-1}^p))))$
    /* Reparameterization trick based on Eq. 10 */
    Obtain $\varepsilon \sim \mathcal{N}(0, I)$, set
    $\boldsymbol{d}_t \leftarrow \varepsilon \odot diag(\psi_{\sigma_d^2}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)) + \psi_{\mu_d}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$
    /* Obtain the log-likelihood using Eq. 7 */
    $\mathcal{L}_{llh} \leftarrow \mathcal{L}_{llh} +$
    $\log[\mathcal{N}(\boldsymbol{x}_t | f_{\mu_x}(\boldsymbol{h}_{t-1}, \boldsymbol{d}_t, \boldsymbol{s}), diag(f_{\sigma_x^2}(\boldsymbol{h}_{t-1}, \boldsymbol{d}_t, \boldsymbol{s})))]$
    /* Update state variables using Eq. 2 and Eq. 3 */
    Obtain $\iota \sim \mathcal{N}(0, I)$, set
    $\boldsymbol{h}_t^p \leftarrow r_p(\boldsymbol{h}_{t-1}^p, \iota \odot diag(g_{\sigma_d^2}(\boldsymbol{h}_{t-1}^p)) + g_{\mu_d}(\boldsymbol{h}_{t-1}^p))$
    $\boldsymbol{h}_t \leftarrow r(\boldsymbol{h}_{t-1}, \boldsymbol{d}_t, \boldsymbol{x}_t)$
  end
  $\mathcal{L}(\boldsymbol{x}_{1:T}; \boldsymbol{\theta}, \boldsymbol{\phi}) \leftarrow \mathcal{L}_{llh} - \mathcal{L}_{kl\_s} - \mathcal{L}_{kl\_d}$ // Eq. 11
  /* Update parameters according to gradients, Eq. 12 */
  $\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow \mathrm{Adam}(-\nabla_{\boldsymbol{\theta}, \boldsymbol{\phi}} \mathcal{L}(\boldsymbol{x}_{1:T}; \boldsymbol{\theta}, \boldsymbol{\phi}))$
until *convergence*

the score, the higher the degree of anomaly. There are various parametric [24] or nonparametric [16, 31] thresholding techniques to conduct anomaly detection based on anomaly score. In this paper, we do not take thresholding technique as our major work and leave it as a future work.

4 EVALUATION

4.1 Datasets

Extensive experiments are conducted on the basis of two categories of real-world datasets to evaluate the effectiveness of SDFVAE. The first consists of three CDN multivariate KPI datasets collected from a top ISP-operated CDN in China, while the other comes from a public dataset named SMD (Server Machine Dataset) [31].

For CDN multivariate KPI datasets, the three datesets are quite different from each other since they are collected from different provincial-level edge sites of the CDN provider. Besides, the first and the second datasets correspond to two popular VoD (Video on Demand) websites respectively, while the other is of a live streaming website. In addition, each dataset contains different levels of noise. The basic statistics of our datasets are summarized in Table 1. It should be noted that there are 7, 5 and 6 ground-truth anomaly segments in the test set of the three datasets, which have been confirmed by human operators.

For the public SMD dataset [4], it contains 28 entity-level datasets each of which was collected from a server machine to indicate the measures like CPU load and network usage, etc.. Among them, we observe that some datasets show a higher degree of noise, while the other exhibit a lower noise level[5]. In order to evaluate the performance of different algorithms on both regular data and noisy data, we manually divide them into two groups, namely, SMD-H with high-level noisy data and SMD-L with regular data. SMD-H consists of the datasets of 'machine-1-5', 'machine-3-5', 'machine-3-8' and 'machine-3-10', while others are included in SMD-L.

4.2 Evaluation Metrics

We employ four metrics including Precision, Recall, F1-score and PR_AUC (Area Under Curve) for performance evaluation. Since no specific threshold selecting method is provided in SDFVAE (we

[4] See https://github.com/NetManAIOps/OmniAnomaly for details.
[5] The noise degree can be quantified by SNR (Signal-to-Noise Ratio), however, we make a rough judgment for simplicity.

Table 1: Basic statistics and settings of all datasets

Statistics	KPIs of VoD1	KPIs of VoD2	KPIs of Live
# KPIs	24	16	48
durations (day)	78	64	54
granularity (min)	5	1	5
# points	22,356	91,507	15,617
# anomaly segments	7	5	6
anomaly ratio (%)	1.6	0.434	1.24
train period	1 ~ 10,656	1 ~ 51,336	1 ~ 7,808
test period	10,657 ~ 22,356	51,337 ~ 91,507	7,809 ~ 15,617

leave it as the future work), we obtain all F1-score by enumerating all thresholds and use the best F1-score as the final score which is also denoted as F1-best [31, 35]. In general, ROC_AUC can also be employed as the evaluation metrics especially when no thresholding technique is specified. However, considering the highly skewed datasets, e.g., 0.43% of anomaly ratio for the CDN VoD2 dataset, even a large change in the number of false positives may lead to an insignificant change of the false positive rate used in ROC (Receiver Operator Characteristic) analysis. In this case, the performance of algorithms tends to exhibit no significant difference in ROC space, however, such difference can be clearly captured in PR (Precision-Recall) space [9]. Therefore, we prefer to use F1-best and PR_AUC as our evaluation metrics.

From a practical point of view, we mainly focus on contiguous anomalies or anomaly segment, instead of single-point anomaly. However, considering the number of anomaly segments is too small in our datasets, we calculate aforementioned metrics based on point anomaly as follows. If any point in a ground-truth anomaly segment is correctly detected, all points in the ground-truth anomaly segment will be identified as true positive, while the points outside the ground-truth anomaly segment will be considered as normal [31].

4.3 Baseline Methods

We compare SDFVAE with four state-of-the-art unsupervised methods for multivariate data as follows.

- **MSCRED [39].** The signature matrix is introduced to capture the correlations of different KPI pairs and resist noise. Then a hierarchical AutoEncoder is proposed to model the spatial and temporal information hidden in the signature matrices. Anomaly score can be achieved based on the reconstruction error of signature matrix.
- **LSTMED [24].** A Seq2seq based deterministic model to capture the patterns of multivariate time series by learning the temporal dependency.
- **DAGMM [42].** A deep Autoencoder based stochastic model for multivariate data. However, no temporal information is taken into account.
- **OmniAnomaly [31].** A stochastic recurrent neural network based model to learn robust representation of multivariate data, and perform anomaly detection based reconstruction probability.

These baselines are carefully selected with respect to different properties as summarized in Table 2. It is worth noting that all baseline methods do not consider time-invariant characteristic hidden in multivariate time series data.

In our experiments, we implement SDFVAE based on Pytorch. Both CNN encoder and DCNN decoder are with 3 convolutional

Table 2: Comparison of baselines and SDFVAE

Methods	Deterministic / Stochastic	Consider Temporal Dependency (time-variant)	Consider time-invariance
MSCRED	deterministic	✓	×
LSTMED	deterministic	✓	×
DAGMM	stochastic	×	×
OmniAnomaly	stochastic	✓	×
SDFVAE	stochastic	✓	✓

layers, whose filters and strides are set according to the number of KPIs. For instance, for the VoD1 dataset with 24 KPIs, the filters of CNN encoder and the corresponding strides are (2,2), (2,2), (2,3) successively. The size of vector $\hat{\boldsymbol{x}}_t$ is fixed to 100, and the dimensions of the hidden states of LSTMCell and BiLSTM are 40. The parameters w, l and T of data preprocessing is set to 36, 10 and 20, respectively. Besides, we set s-space and d-space dimensions to 8 and 10 empirically. Adam optimizer is employed with learning rate of 0.0002, the batch size is set to 64. The other four baselines are reproduced based on open-source codes [6]. For each baseline, we adjust its hype-parameters, for instance, training epochs and dimensions of latent space, to obtain optimized performance. Our experiments are conducted on a server with Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz accelerated by a NVIDIA TITAN Xp graphics card with 12GB VRAM.

4.4 Results and Analysis

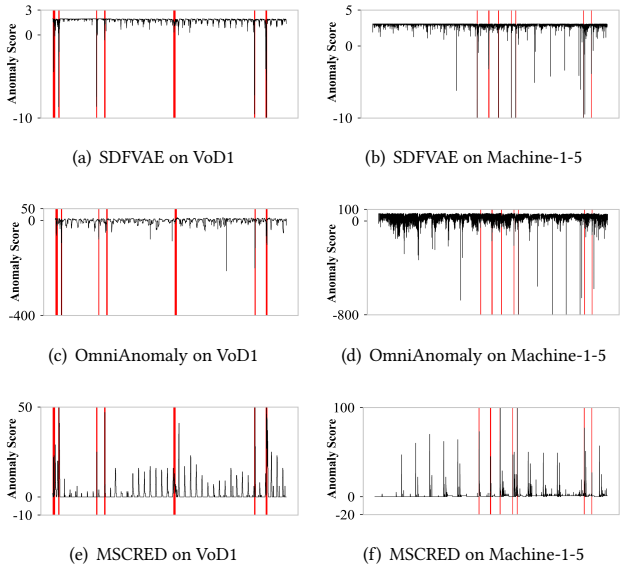
4.4.1 SDFVAE vs. Baselines. Table 3 reports the precision, recall and F1-best of the four state-of-the-art baselines and SDFVAE on both the CDN datasets and the SMD datasets, where the best F1-scores for all methods are highlighted in bold-face and the second best are indicated by underline. As for SMD-L and SMD-H datasets, it illustrates the overall performance of each method via unionizing the datasets belonging to them. Compared with other baselines, SDFVAE consistently achieves the best performance on all datasets. Specifically, SDFVAE achieves the F1-best of 0.965, 0.919 and 0.992 on datasets of VoD1, VoD2 and Live, as well as 0.984 and 0.973 on the lower and higher degree noise of the public datasets, respectively. The case studies of SDFVAE are shown in Fig. 7(a) and 7(b), where it is observed that the anomaly score of SDFVAE is relatively stable at most of the time, while presents serious spikes in the anomaly regions. It further demonstrates the capability of SDFVAE to learn normal patterns of noisy KPIs.

OmniAnomaly builds on a stochastic model being capable of learning robust representation of multivariate time series data and achieves the second best overall performance. However, OmniAnomaly introduces only the dynamic latent variables and can be trained to learn the posterior distribution of both normal data and noises, so it suffers from the problem of over-fitting. In this case, some anomalies may not be detected, since the distribution of them may be similar to that of noise. The case studies shown in Fig. 7 validate our conjecture. As shown in Fig. 7(c), we notice the anomaly score of the fifth anomaly on VoD1 dataset is much lower than that shown in Fig. 7(a), thus this anomaly has the risk

[6] https://github.com/7fantasysz/MSCRED,
https://github.com/KDD-OpenSource/DeepADoTS,
https://github.com/NetManAIOps/OmniAnomaly

Table 3: Comparison of Anomaly Detection Performance based on Precision, Recall and F1-best

Methods	Noisy Data												Regular Data		
	KPIs of VoD1			KPIs of VoD2			KPIs of Live			SMD-H			SMD-L		
	Pre	Rec	F_{best}	Pre	Rec	F_{best}	Pre	Rec	F_{best}	Pre	Rec	F_{best}	Pre	Rec	F_{best}
MSCRED	0.772	0.777	0.775	0.849	0.783	0.815	0.893	0.861	0.877	0.636	0.851	0.727	0.916	0.860	0.887
LSTMED	0.881	0.67	0.758	0.792	0.854	0.821	0.819	0.861	0.839	0.593	0.832	0.692	0.812	0.896	0.852
DAGMM	0.785	0.67	0.723	0.925	0.627	0.747	0.972	0.706	0.818	0.513	0.731	0.603	0.694	0.864	0.770
OmniAnomaly	0.97	0.67	0.793	0.753	0.854	0.80	0.928	0.861	0.893	0.540	0.787	0.641	0.976	0.975	0.976
SDFVAE	0.933	1.0	**0.965**	0.994	0.854	**0.919**	0.985	1.0	**0.992**	0.984	0.963	**0.973**	0.982	0.987	**0.984**



(a) SDFVAE on VoD1 (b) SDFVAE on Machine-1-5

(c) OmniAnomaly on VoD1 (d) OmniAnomaly on Machine-1-5

(e) MSCRED on VoD1 (f) MSCRED on Machine-1-5

Figure 7: Case study of anomaly score on VoD1 and SMD machine-1-5 datasets. Regions highlighted in red represent the ground-truth anomaly segments

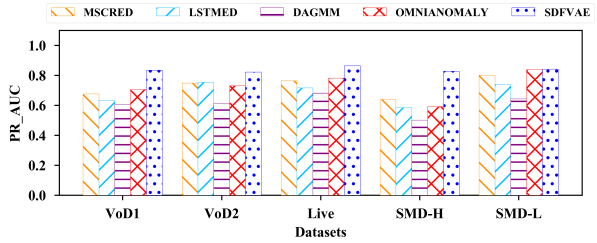


Figure 8: PR_AUC over different datasets

to be missed. The reason behind it is that OmniAnomaly assigns a higher log-likelihood to it due to the over-fitting. We also observe similar cases when we compare Fig. 7(d) with Fig. 7(b). Besides, the more turbulent anomaly score further demonstrates that OmniAnomaly is inferior to SDFVAE in capturing the normal patterns of noisy KPIs, especially as shown in 7(d). Consequently, it achieves an inferior performance compared with SDFVAE.

MSCRED is designed to be robust to noisy data via introducing the signature matrix which utilizes the correlations between different KPI pairs. As shown in our experiments, it presents a decent performance, especially on noisy data like SMD-H. However, it performs much less compared with SDFVAE. Specifically, the F1-best of MSCRED on SMD-H is 0.727 while SDFVAE reaches to 0.973. One reason behind that is the signature matrix is not sensitive to some real anomalies, especially for anomalies with small degree or short duration. Thus, it fails to detect these anomalies. Another possible reason lies in that MSCRED is a deterministic model and is not capable of learning the robust representations of varying characteristics in multivariate time series data. Therefore, there are some characteristics that MSCRED may have never seen before, which often leads to higher reconstruction errors. The case study shown in Fig. 7 also verifies our conjectures. Compared with SDFVAE and OmniAnomaly, MSCRED tends to achieve a higher anomaly score in some normal regions since it fails to learn the robust representations. In addition, MSCRED derives a lower anomaly score with respect to some anomaly regions, e.g., the fifth anomaly region in Fig. 7(e) and the sixth region in Fig. 7(f) , because the signature matrix is insensitive to some anomalies with small degree or short duration. LSTMED belongs to a deterministic model whose hidden layer is composed of LSTM units. Therefore, it fails to handle stochastic information and learn robust representations, which lead to a lower performance. Since DAGMM is designed for multivariate data rather than time series, it shows a plain performance, however, it illustrates that employing the spatial characteristics of multivariate data also works for anomaly detection. Meanwhile, it also validates the importance of temporal characteristics. Accordingly, just as presented by the performance of SDFVAE, considering both temporal and spatial characteristics of multivariate time series is critical for anomaly detection.

We also illustrate the corresponding PR_AUC in Fig. 8. It is observed that PR_AUC is mostly consistent with F1-best except for slight differences. Besides, since it is more important to have an excellent F-score at a certain threshold than to have just high but not so excellent F-scores at most thresholds [35], here we prefer to use F1-best to demonstrate the performance.

The above results validate that SDFVAE has significant advantages of performance on both regular and noisy data. The reason behind the performance is that SDFVAE is elaborately designed to explicitly take both time-varying and time-invariant characteristics into account. Thanks to the time-invariance which is not sensitive to noise (even the none-additive Gaussian noise), SDFVAE is capable of alleviating the over-fitting to some extent and learning robust and expressive representations of multivariate KPIs.

4.4.2 SDFVAE vs. Variants. Except for the four baselines, we also compare the performance of SDFVAE with its two variants, namely SDFVAE-v1 and SDFVAE-v2. Among them, SDFVAE-v1 leaves the dynamic latent space alone, while removing the static latent space $\boldsymbol{s}$.

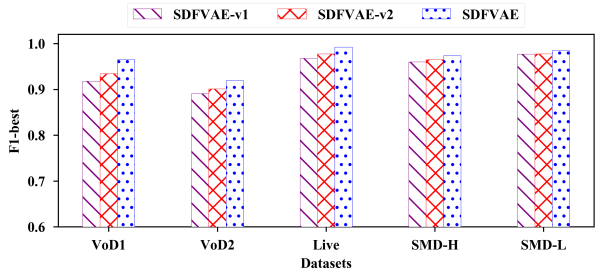
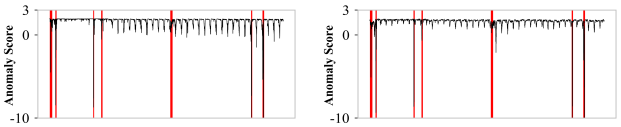


Figure 9: Comparison of SDFVAE with its variants



(a) SDFVAE-v1 (b) SDFVAE-v2

Figure 10: Case study of SDFVAE variants on VoD1 dataset

The other variant SDFVAE-v2 is to replace the conditional prior distribution of dynamic latent variables $\boldsymbol{d}$ at each time step $p_{\theta}(\boldsymbol{d}_t|\boldsymbol{d}_{<t})$ with time-independent standard diagonal multivariate Gaussian $p_{\theta}(\boldsymbol{d}_t) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

As shown in Fig. 9, SDFVAE-v1 achieves the lowest F1-best on all datasets due to removing the static latent space $\boldsymbol{s}$. The reason probably lies in that, without static latent variables, SDFVAE-v1 cannot learn the representations of time-invariant characteristics of multivariate KPIs, thus is not capable of resisting noise. Meanwhile, it is observed that the F1-best of SDFVAE-v2 on all datasets is also lower than that of standard SDFVAE. The reason for this is that SDFVAE-v2 replaces the time-dependent (conditional) prior of dynamic latent variables $\boldsymbol{d}$ with time-independent prior, thus fails to capture the feature of temporal dependency effectively. Compared with SDFVAE and SDFVAE-v2, the lowest performance of SDFVAE-v1 further emphasizes the importance of time-invariant characteristics for anomaly detection of multivariate KPIs. In addition, we also show the case studies of SDFVAE-v1 and SDFVAE-v2 in Fig. 10. Compared the anomaly score shown in Fig. 10(a) with that in Fig. 7(a), we notice that the anomaly score of SDFVAE-v1 is more turbulent than that of SDFVAE. Thus there is relatively higher log-likehilood of the fifth anomaly due to the over-fitting of SDFVAE-v1 which results in its inferior perfromance.

To sum up, our experimental results validate that the key designs of SDFVAE, including factorizing the latent space into two separate parts and time-dependent prior of dynamic latent variables, are sensitive to the performance and thus are beneficial for multivariate anomaly detection.

4.5 Parameter Sensitivity

We first study the sensitivity of the hyper-parameters w, l and T set used for data preprocessing. For simplicity, we pick the representative dataset VoD1 to conduct our experiments. We show the results in Fig. 11. To study the impact of w, we set $l = 10$ and $T = 20$, and then we increase w from 1 to 144. It is observed that lower values of w tend to result in poor performance, since the

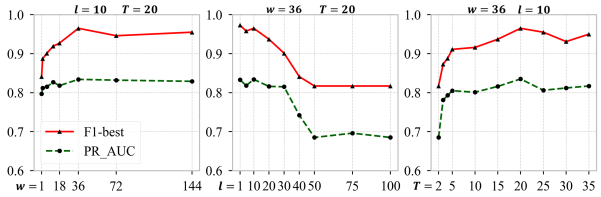


Figure 11: The sensitivity of w, l and T on VoD1 dataset

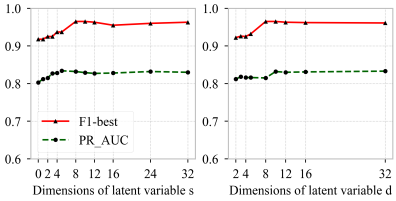


Figure 12: The sensitivity of dimensions of latent variables on VoD1 dataset

shorter the length of the observed variable, the less correlation information it contains in each pair of KPIs. Thus SDFVAE may not be able to capture the normal patterns effectively via time-invariant characteristics, especially when $w = 1$. Meanwhile, we observe that the performance tends to be relatively stable when w reaches to 18.

Similarly, we conduct additional experiments to analyze the effect of l and T. We increase l from 1 to 100 and notice that a smaller value of l makes higher performance. It is because the longer strides between two consecutive variables l may increase the difficulty of learning the temporal dependency. Thus it leads to inferior performance. After that, we increase T from 2 to 35 and observe that SDFVAE exhibits inferior performance before T reaches to 5. The reason is that one sequence with less observed variables may contains less time-varying and time-invariant information, which is not enough to capture the normal patterns. It validates that SDFVAE can achieve a relatively stable performance in a wide range of w, T and l, except for some very small values of w, T and large values of l.

Next we analysis the impact of the dimensions of latent variables s and d. This experiment is also conducted based on VoD1 dataset. Fig. 12 shows F-best and PR_AUC of SDFVAE by varying dimensions of static and dynamic latent variables. It is observed that the F-best do not change significantly when the dimensions vary between 8 and 32. This demonstrates there is a large room for us to choose the latent variable dimension.

4.6 Algorithm Efficiency

In order to examine the feasibility of SDFVAE in real system, we study its efficiency in terms of training time and testing time, with the same parameter setting and hardware configuration as described in subsection 4.3. Since SDFVAE tends to converge within 30 epochs on all datasets we use, we record the training time of SDFVAE by running 30 epochs. As shown in Table 4, the training time of SDFVAE increases linearly with the number of training samples and ranges from 10 minutes to 61 minutes, based on our experiment server equipped with a single TITAN GPU. The maximum training time reaches to 61 minutes when training the dataset of VoD2 which

Table 4: Training and testing time of SDFVAE

Datasets	# Training samples	Training times (min)	Testing times per sample (sec)
VoD1	10,430	14.25	0.045
VoD2	51,112	61	0.045
Live	7,582	10.25	0.044

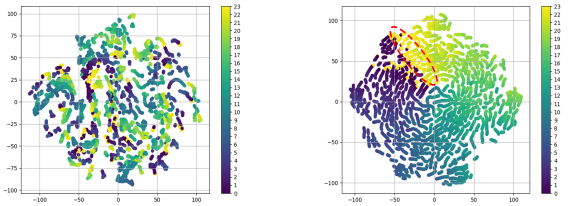
has around 51000 samples and spans over 35 days. Given a testing sample, the average time to obtain the anomaly score is around 0.045 seconds on these three CDN datasets. These results further demonstrate that SDFVAE can be easily deployed in real-world CDN in the manner of offline training and online detection.

4.7 Visualization of Latent Variables

To further demonstrate the capability of our model to extract static and dynamic representations, we conduct an additional experiment to visualize the learned latent variables.

Specifically, we start with selecting some labeled normal observed sequences $(\boldsymbol{x}_{1:T}^{(\tau)}, y)$ from the dataset of VoD1, where y denotes the certain hour that an observed sequence belongs to. Then we obtain the static variables $\boldsymbol{s}$ and dynamic variables $\boldsymbol{d}_{1:T}$ of these observed sequences with the trained SDFVAE, and employ t-Distributed t-SNE (Stochastic Neighbor Embedding) [33] to project them separately to a two-dimensional space. We color-code both projected $\boldsymbol{s}$ and $\boldsymbol{d}$ of each observed sequence according to the hour it belongs to, that is, the hour of time step $\boldsymbol{k}_\tau$ shown in Fig. 4. Intuitively, due to the multivariate KPIs show periodic properties, the observe sequences belong to the same hour tend to exhibit the similar time-varying characteristic, even though some of them do not belong to the same day. Hence, the dynamic latent variables tend to form clusters in the projected $\boldsymbol{d}$ space according to the hours they belong to. Meanwhile, since the time-invariant characteristic remains unchanged over time, the static latent variables should scatter randomly in the projected $\boldsymbol{s}$ space, despite some of them belong to the same hour.

As shown in Fig. 13(a) and 13(b), each point indicates the latent variables of an observed sequence and the corresponding color represents the certain hour it belongs to. We notice that in the projected $\boldsymbol{d}$ space, the dynamic latent variables of observed sequences in the same hour tend to form a cluster, e.g., as shown in Fig. 13(b), the cluster in yellow highlighted by a red circle is formed by samples belong to 23:00. It should be noted that we only show the dynamic latent variables $\boldsymbol{d}_T$ which represent the factors indicating how $\boldsymbol{x}_T^{(\tau)}$ changes over the previous observed variables $\boldsymbol{x}_{1:T-1}^{(\tau)}$. In contrast, there is no such clustering phenomenon in the projected $\boldsymbol{s}$ space, which implies that $\boldsymbol{s}$ tends to contain information about time-invariant factors instead of time-varying factors. As a result, this experiment demonstrates that SDFVAE can explicitly learn the representations of both the time-invariant and time-varying characteristics.



(a) Latent variables $\boldsymbol{s}$ by hour (b) Latent variables $\boldsymbol{d}_T$ by hour

Figure 13: Latent variables visualized via t-SNE

5 RELATED WORK

5.1 Multivariate Anomaly Detection

There are a growing number of literature on multivariate anomaly detection. Since supervised methods [23, 29, 37] usually suffer from labor-intensive data labelling and thus become impractical in most scenarios of anomaly detection, here we mainly summarize deep learning based unsupervised methods [16, 21, 24, 31, 39, 42].

Telemanom [16] is a prediction based method to detect anomaly for telemetry channels of spacecrafts via modeling the temporal dependency of time series data through LSTM. It determines an anomaly depending on the residual error between the predicted and the observed value. LSTMED [24] is a seq2seq [7, 32] based anomaly detection method where a LSTM based encoder-decoder is employed to learn the temporal dependency of multivariate time series and anomaly detection is determined based on reconstruction errors. It belongs to a deterministic anomaly detection method due to its design of deterministic latent space. Compared with the above deterministic models, some recent studies [3, 21, 31, 42] show stochastic approach has the potential to learn robust representations of multivariate time series since it helps capture the probability distributions of them. DAGMM [42] utilises an Autoencoder to learn representations and a Gaussian Mixture Model to perform density estimation, however, it ignores the temporal information. [21] proposes a Generative Adversarial Network (GAN) based multivariate anomaly detection method, employing the Long-Short-Term-Memory Recurrent Neural Networks (LSTM-RNN) as the base model of the GAN framework to capture the temporal and spatial information of time series. OmniAnomaly [31] is also a stochastic recurrent neural network based method for multivariate time series. It aims to learn the robust normal patterns of multivariate data.

5.2 Anomaly Detection on Noisy Data

Since deep learning-based models tend to suffer from the problem of over-fitting due to noisy data [25, 38], various noise-robust models have been studied in image classification[11, 40] and generation [4, 18]. [41] proposes a robust detection model on image by applying the idea of RPCA (Robust Principal Components Analysis) into AutoEncoders. However, there are few studies on anomaly detection of noisy time series data. [5] utilizes the adversarial training technique to capture the complex patterns in univariate KPI with non-Gaussian noises and complex data distribution. As suggested in previous study [14, 17, 30], the correlations between different pairs of multivariate time series are critical to characterize the system, thus may contribute to the performance of anomaly detection. MSCRED [39] introduces a signature matrix and utilizes the correlation of KPI pairs to resist noise, where a hierarchical encoder-decoder based deterministic models is used to capture the spatial and temporal patterns. However, the signature matrix is

not sensitive to some anomalies, especially for these with lower degree and short duration anomalous. Additionally, MSCRED belongs to a deterministic model, thus may fail to learn the robust representations of multivariate with different characteristics.

Similar with some of existing studies, SDFVAE also targets to learn both temporal and spatial representations of multivariate data. However, instead of resorting directly to the correlations between KPIs, we take a step forward and reveal a more general time-invariant characteristics in multivariate data. Further we utilize the time-invariance to build a robust and noise-resilient anomaly detection approach.

6 CONCLUSION

Through an in-depth analysis of real-world multivariate CDN KPI dataset, for the first time we reveal that multivariate KPIs exhibit time-invariant characteristics and that explicitly modelling such invariance may help resist noise in the data. Further, we propose a novel multivariate anomaly detection method called SDFVAE to learn the representation of KPIs via explicitly factorizing the latent variables into dynamic and static parts. Our experiments based on real-world data show that SDFVAE significantly outperforms state-of-the-art baselines.

ACKNOWLEDGMENTS

This work was supported in part by National Natural Science Foundation of China under Grant 61572497, 61771469, 62072302 and 61960206002.

REFERENCES

[1] Jinwon An and Sungzoon Cho. 2015. Variational Autoencoder based Anomaly Detection using Reconstruction Probability. *Technical Report. SNU Data Mining Center* (2015), 1–8.
[2] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A. Zuluaga. 2020. USAD: UnSupervised Anomaly Detection on Multivariate Time Series. In *ACM SIGKDD 2020*. 3395–3404.
[3] Philip Becker-Ehmck, Jan Peters, and Patrick van der Smagt. 2019. Switching Linear Dynamics for Variational Bayes Filtering. In *ICML 2019*. 553–562.
[4] Ashish Bora, Eric Price, and Alexandros G. Dimakis. 2018. AmbientGAN: Generative models from lossy measurements. In *ICLR 2018*.
[5] Wenxiao Chen, Haowen Xu, Zeyan Li, Dan Pei, Jie Chen, Honglin Qiao, Yang Feng, and Zhaogang Wang. 2019. Unsupervised Anomaly Detection for Intricate KPIs via Adversarial Training of VAE. In *IEEE INFOCOM 2019*. 1891–1899.
[6] Yingying Chen, Ratul Mahajan, Baskar Sridharan, and Zhi-Li Zhang. 2013. A provider-side view of web search response time. In *ACM SIGCOMM 2013*.
[7] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP 2014*. 1724–1734.
[8] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. 2015. A Recurrent Latent Variable Model for Sequential Data. In *NIPS 2015*. 2980–2988.
[9] Jesse Davis and Mark Goadrich. 2006. The relationship between Precision-Recall and ROC curves. In *ICML 2006*, Vol. 148. 233–240.
[10] Emily L. Denton and Vighnesh Birodkar. 2017. Unsupervised Learning of Disentangled Representations from Video. In *NIPS 2017*.
[11] Steven Diamond, Vincent Sitzmann, Stephen P. Boyd, Gordon Wetzstein, and Felix Heide. 2017. Dirty Pixels: Optimizing Image Classification Architectures for Raw Sensor Data. arXiv:1701.06487
[12] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. 2016. Sequential Neural Models with Stochastic Layers. In *NIPS 2016*. 2199–2207.
[13] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18 (2005), 602–610.
[14] David Hallac, Sagar Vare, Stephen P. Boyd, and Jure Leskovec. 2017. Toeplitz Inverse Covariance-Based Clustering of Multivariate Time Series Data. In *ACM SIGKDD 2017*. 215–223.
[15] Wei-Ning Hsu, Yu Zhang, and James R. Glass. 2017. Unsupervised Learning of Disentangled and Interpretable Representations from Sequential Data. In *NIPS 2017*. 1878–1889.
[16] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Söderström. 2018. Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding. In *ACM SIGKDD 2018*. 387–395.
[17] Guofei Jiang, Haifeng Chen, and Kenji Yoshihira. 2007. Efficient and Scalable Algorithms for Inferring Likely Invariants in Distributed Systems. *IEEE Trans. Knowl. Data Eng.* 19 (2007), 1508–1523.
[18] Takuhiro Kaneko and Tatsuya Harada. 2020. Noise Robust Generative Adversarial Networks. In *IEEE/CVF CVPR 2020*. 8401–8411.
[19] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR 2015*.
[20] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR 2014*.
[21] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. 2019. MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks. In *Artificial Neural Networks and Machine Learning - ICANN*. 703–716.
[22] Yingzhen Li and Stephan Mandt. 2018. Disentangled Sequential Autoencoder. In *ICML 2018*. 5656–5665.
[23] Dapeng Liu, Youjian Zhao, Haowen Xu, Yongqian Sun, Dan Pei, Jiao Luo, Xiaowei Jing, and Mei Feng. 2015. Opprentice: Towards Practical and Automatic Anomaly Detection Through Machine Learning. In *ACM IMC 2015*. 211–224.
[24] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. In *ICML 2016 Anomaly Detection Workshop*.
[25] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. 2020. Deep Learning for Anomaly Detection: A Review. arXiv:2007.02500
[26] Spiros Papadimitriou, Jimeng Sun, and Philip S. Yu. 2006. Local Correlation Tracking in Time Series. In *IEEE ICDM 2006*. 456–465.
[27] Shaogang Ren, Dingcheng Li, Zhixin Zhou, and Ping Li. 2020. Estimate the Implicit Likelihoods of GANs with Application to Anomaly Detection. In *WWW '20: The Web Conference 2020*. 2287–2297.
[28] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *ICML 2014*. 1278–1286.
[29] Taeshik Shon and Jongsub Moon. 2007. A hybrid machine learning approach to network anomaly detection. *Inf. Sci.* 177, 18 (2007), 3799–3821.
[30] Dongjin Song, Ning Xia, Wei Cheng, Haifeng Chen, and Dacheng Tao. 2018. Deep r -th Root of Rank Supervised Joint Binary Embedding for Multivariate Time Series Retrieval. In *ACM SIGKDD 2018*. 2229–2238.
[31] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. 2019. Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network. In *ACM SIGKDD 2019*. 2828–2837.
[32] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *NIPS 2014*. 3104–3112.
[33] Laurens van der Maaten. 2014. Accelerating t-SNE using tree-based algorithms. *J. Mach. Learn. Res.* 15, 1 (2014), 3221–3245.
[34] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. 2017. Decomposing Motion and Content for Natural Video Sequence Prediction. In *ICLR 2017*.
[35] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, Jie Chen, Zhaogang Wang, and Honglin Qiao. 2018. Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications. In *WWW 2018*. 187–196.
[36] Li Xu, Jimmy S. J. Ren, Ce Liu, and Jiaya Jia. 2014. Deep Convolutional Neural Network for Image Deconvolution. In *NIPS 2014*. 1790–1798.
[37] Makoto Yamada, Akisato Kimura, Futoshi Naya, and Hiroshi Sawada. 2013. Change-Point Detection with Feature Selection in High-Dimensional Time-Series Data. In *IJCAI 2013*. 1827–1833.
[38] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *ICLR 2017*.
[39] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V. Chawla. 2019. A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data. In *AAAI 2019*. 1409–1416.
[40] Stephan Zheng, Yang Song, Thomas Leung, and Ian J. Goodfellow. 2016. Improving the Robustness of Deep Neural Networks via Stability Training. In *IEEE CVPR 2016*. 4480–4488.
[41] Chong Zhou and Randy C. Paffenroth. 2017. Anomaly Detection with Robust Deep Autoencoders. In *ACM SIGKDD 2017*. 665–674.
[42] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Dae-ki Cho, and Haifeng Chen. 2018. Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection. In *ICLR 2018*.