Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network

Ya Su[*]
Tsinghua University; BNRist

Youjian Zhao
Tsinghua University; BNRist

Chenhao Niu
Tsinghua University; BNRist

Rong Liu
Stevens Institute of Technology

Wei Sun
Beijing University of Posts and Telecommunications

Dan Pei[†]
Tsinghua University; BNRist

ABSTRACT

Industry devices (*i.e.*, entities) such as server machines, spacecrafts, engines, *etc.*, are typically monitored with multivariate time series, whose anomaly detection is critical for an entity's service quality management. However, due to the complex temporal dependence and stochasticity of multivariate time series, their anomaly detection remains a big challenge. This paper proposes *OmniAnomaly*, a stochastic recurrent neural network for multivariate time series anomaly detection that works well robustly for various devices. Its core idea is to capture the normal patterns of multivariate time series by learning their robust representations with key techniques such as stochastic variable connection and planar normalizing flow, reconstruct input data by the representations, and use the reconstruction probabilities to determine anomalies. Moreover, for a detected entity anomaly, *OmniAnomaly* can provide interpretations based on the reconstruction probabilities of its constituent univariate time series. The evaluation experiments are conducted on two public datasets from aerospace and a new server machine dataset (collected and released by us) from an Internet company. *OmniAnomaly* achieves an overall F1-Score of 0.86 in three real-world datasets, significantly outperforming the best performing baseline method by 0.09. The interpretation accuracy for *OmniAnomaly* is up to 0.89.

CCS CONCEPTS

• **Computing methodologies → Anomaly detection**; **Neural networks**; **Bayesian network models**.

KEYWORDS

Anomaly Detection; Multivariate Time Series; Stochastic Model; Recurrent Neural Network

[*]BNRist: Beijing National Research Center for Information Science and Technology
[†]Dan Pei is the corresponding author.

ACM Reference Format:
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 *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3292500.3330672

1 INTRODUCTION

Anomaly detection has been an active research topic in SIGKDD community with applications in graph [3, 12], log messages [15, 23], time series [6, 9, 14, 22], *etc.* In this paper, we focus on anomaly detection for multivariate time series [6]. Industry devices, such as server machines [9, 14], spacecrafts [6], robot-assisted systems [16, 17], engines [11], are typically monitored with multiple time series metrics (also called telemetry data, sensor data, *etc.*) so that each device's behavioral anomalies can be timely detected and later resolved. Multiple univariate time series from the same device (or more generally, an entity) forms a multivariate time series. Table 1 shows some example datasets about entities, and Fig. 1 shows an example multivariate time series snippet with two anomalous regions from the server machine dataset.

Table 1: Example datasets about entities

Entity Type and dataset	NO. of entities	NO. of metrics per entity	Metric Name
Server machine [this paper]	28	38	CPU load, network usage, memory usage, *etc.*
Soil Moisture Active Passive satellite [6]	55	25	Telemetry data: radiation, temperature, power, computational activities, *etc.*
Mars Science Laboratory rover [6]	27	55	
Robot-assisted system [16, 17]	~39	17	Sensor data: kinematic, visual, haptic, auditory, *etc.*
Engine [11]	-	12	Sensor data: accelerator, torque, temperature, *etc.*

In general, it is preferred to detect entity anomalies at the entity-level directly using multivariate time series [6, 11, 16, 17], rather than at the metric-level using univariate time series, for a number of reasons. First, in practice, operation engineers are more concerned about the overall status of an entity than each constituent metric. Second, it is labor-intensive to train and maintain an individual anomaly detection model for each metric, given a large number of metrics (*e.g.*, 1485 (27*55) for Mars Science Laboratory rover in Table 1). Third, an incident (*e.g.*, overload) at an entity typically causes anomalies in multiple metrics. If we detect anomalies at the metric-level, we would need to define rules based on extensive domain knowledge to process anomaly results of all metrics to determine whether the entity is anomalous or not, which is challenging to do. Forth, intuitively, modeling the expected value of one univariate time series can benefit from the more information in the multivariate time series of the same entity. In summary, it is more intuitive, effective and efficient to detect anomalies at the entity-level than at the metric-level. Thus, in this paper, similar to [6, 11, 16, 17], we focus on detecting the overall anomalies of the multivariate time series of each monitored entity.

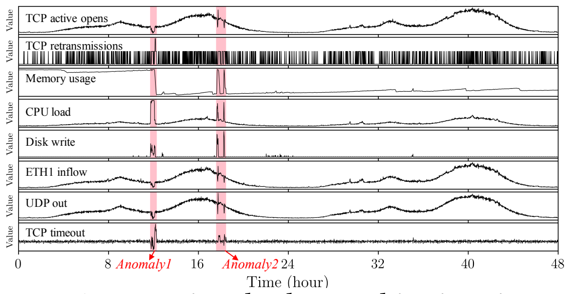


Figure 1: An 8-metric 2-day-long multivariate time series snippet from the server machine dataset, with two anomalous regions highlighted in pink.

Entities under our study (servers, spacecrafts, *etc.*) are engineering artifacts, which have software control logic and interact with environment, human operators, and other systems in a very complex way. As a result, their complex behaviors can exhibit both stochasticity and strong temporal dependence. A previous study [5] has shown that the stochasticity in speech sequence can be more precisely captured by stochastic variables with properly estimated probability distributions [5] than deterministic variables. [24] shows that univariate time series in an online shopping website can present complex temporal relationships. Therefore, ideally our study should take *a stochastic approach with temporal dependence modeling*. However, despite the rich literature in multivariate time series anomaly detection in different areas [4, 6, 11, 16, 17, 20, 26, 27], previous studies *either* take deterministic approaches [4, 6, 11] to model time series, *or* take stochastic approach but ignore the temporal dependence of observations in the time series [27].

Due to anomaly diversities and the lack of labels for training [6], our approach has to be an unsupervised one. Based on our observation and intuition, anomalies are usually unexpected instances significantly deviating from normal patterns formed by the majority of a dataset. Thus, our core idea is to *learn robust latent representations to capture normal patterns of multivariate time series, considering both the temporal dependence and stochasticity.* The more different an observation is from the normal patterns, the more likely it is considered as an anomaly. There are two major challenges for this idea.

The first challenge is how to learn robust latent representations, considering both the temporal dependence and stochasticity of multivariate time series. Previous work [5] has shown that a stochastic model *alone* is hard to capture the long-term dependence and complex probability distributions of multivariate time series. Intuitively, it is advantageous to let the deterministic hidden variable of RNN act as an internal memory for stochastic models [5]. [16] made an attempt along this direction by simply replacing the feed-forward network in a VAE [7] with LSTM [6], but its stochastic variables are very simple, without temporal dependence. Stochastic variables are latent representations of input data and their quality is the key to model performance. To learn robust representations of data, we propose a *stochastic recurrent neural network, with explicit temporal dependence among stochastic variables modeled.* Our approach novelly glues GRU [1] (a variant of RNN) and VAE with the following two key techniques. a) speech reconstruction literature [5] has shown that explicitly modeling the temporal dependence among stochastic variables can make these variables capture more information from historical stochastic variables and represent the input data better. Inspired by [5], to explicitly model temporal dependence among stochastic variables in the latent space, we propose the *stochastic variable connection* technique: Linear Gaussian State Space Model [8] connection between stochastic variables, and the concatenation of stochastic variable and GRU latent variable. b) To help stochastic variables capture complex distributions of input data, we adopt planar Normalizing Flows (planar NF) [18] which uses a series of invertible mappings to learn non-Gaussian posterior distributions in latent stochastic space. These techniques make our model capable of learning salient representations from datasets with different characteristics to achieve great robustness.

The second challenge is how to provide interpretation to the detected entity-level anomalies, given the stochastic deep learning approaches. The interpretation is to answer the question of why an observation is detected as an anomaly. Anomaly interpretation can help analyze the entity anomalies and speed up troubleshooting, and thus is often required in practice [6]. However, it is challenging to interpret anomalies of multivariate time series, and stochastic deep learning approaches make the interpretation even harder. Our solution to this problem is based on the following observation. In practice, when manually checking and troubleshooting entity anomalies, operators typically look for the top few individual metrics that deviate from historical patterns the most. For example, if a server machine is suffering from network slowdown, the metrics related to network would behave more abnormally than other metrics. Thus, in our approach, a detected entity anomaly can be interpreted by a few univariate time series with the lowest reconstruction probabilities.

The contributions of this paper are summarized as follows:

- We propose *OmniAnomaly*, a novel stochastic recurrent neural network for multivariate time series anomaly detection. To the best of our knowledge, *OmniAnomaly* is the first multivariate time series anomaly detection algorithm that can deal with explicit temporal dependence among stochastic variables to learn robust representations of input data, required by industry device monitoring.
- We propose the first anomaly interpretation approach for stochastic based multivariate time series anomaly detection algorithms that works with not only *OmniAnomaly*, but also other algorithms such as [16]. The interpretation accuracy for *OmniAnomaly* is up to 0.89.
- Our experiments show great effect of the four key techniques in *OmniAnomaly*: GRU, planar NF, stochastic variable connection, and an adjusted Peaks-Over-Threshold method for automatic anomaly threshold selection.

- Through extensive experiments, we show that *OmniAnomaly* achieves an overall F1-Score of 0.86 in three real-world datasets, significantly outperforming the best performing baseline model by 0.09, demonstrating the benefits of explicitly modeling the temporal dependence among stochastic variables in the latent space. *OmniAnomaly* exhibits great robustness in working with three datasets from different devices, with F1-Scores all higher than 0.84.
- We publicly publish our code and server machine dataset of experiments on GitHub[1] for better reproducibility of the results of this paper.

2 RELATED WORK

Multivariate time series anomaly detection is an active topic. Supervised learning methods [17, 20] need labeled data for model training and can only identify anomaly occurrences for known anomaly types [13]. As a result, supervised methods have limited usage and unsupervised approaches are desirable. The state-of-the-art unsupervised solutions to multivariate time series anomaly detection in literature can be categorized into the following types:

- Deterministic models [4, 6, 11]. To detect spacecraft anomalies, [6] applies LSTM for multivariate time series prediction and determines anomalies using prediction errors. Similar to seq2seq models, [11] proposes an LSTM-based Encoder-Decoder which aims at reconstructing "normal" time series behaviors, and uses reconstruction errors for multi-sensor anomaly detection. Although LSTM can deal with the temporal dependence of time series, it is deterministic without stochastic variables.
- Stochastic based models [16, 27]. [27] proposes a model DAGMM which joints Deep Autoencoder (AE) and Gaussian Mixture Model (GMM) simultaneously. It reduces the dimension of input observations to get latent representations by AE, and estimates the density of the representations using GMM. However, this method is designed for multivariate *variables* (not multivariate *time series*), and ignores the inherent temporal dependence of time series. Previous work suggests that, in general, stochastic variables can improve the performance of RNN, because they can capture the probability distributions of time series [5]. [16] simply combines LSTM and VAE by replacing the feed-forward network in a VAE to LSTM, but ignores the dependence of stochastic variables.

Compared with the above approaches, *OmniAnomaly* is a stochastic recurrent neural network which glues VAE and GRU such that the temporal dependence and stochasticity of time series can be explicitly modeled. Moreover, *OmniAnomaly* applies techniques such as stochastic variable connection to model the temporal dependence between stochastic variables. As a result, the stochastic variables can capture more information from historical stochastic variables and represent the input data better, as will be demonstrated in Section 5.

3 PRELIMINARIES

In this section, we present the problem statement of multivariate time series anomaly detection in detail and introduce the overall structure of our model. In addition, we provide preliminaries about GRU, VAE, planar NF, the key components of our model.

3.1 Problem Statement

A time series contains successive observations which are usually collected at equal-space timestamps [10]. In our study, we focus on multivariate time series, defined as $\mathbf{x} = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_N}\}$, where N is the length of $\mathbf{x}$, and an observation $\mathbf{x_t} \in R^M$ is an M-dimensional vector [6] at time t ($t \leq N$): $\mathbf{x_t} = [x_t^1, x_t^2, ..., x_t^M]$, and $\mathbf{x} \in R^{M \times N}$. In Fig. 1, the observations are equally spaced by 1 minute, the total number of observations is $N = 2 * 24 * 60$, and each observation has a dimension of $M = 8$. We use $\mathbf{x_{t-T:t}}$ ($\in R^{M \times (T+1)}$) to denote a sequence of observations $\{\mathbf{x_{t-T}}, \mathbf{x_{t-T+1}}, ..., \mathbf{x_t}\}$ from time $t - T$ to t.

For multivariate time series anomaly detection, the objective is to determine whether an observation $\mathbf{x_t}$ is anomalous or not. For time series modeling, historical values are beneficial for understanding current data. Therefore, a sequence of observations $\mathbf{x_{t-T:t}}$ instead of just $\mathbf{x_t}$ is used to calculate the anomaly result. Our anomaly detection approach returns an anomaly score for $\mathbf{x_t}$, and then the anomaly result can be obtained by comparing against a threshold.

3.2 Overall Structure

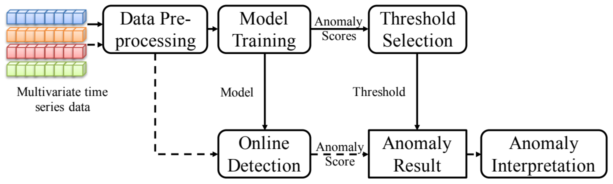


Figure 2: Overall Structure of *OmniAnomaly*. The solid lines denote offline training and the dash lines show the procedure of online detection.

As shown in Fig. 2, the overall structure of *OmniAnomaly* consists of two parts: offline training and online detection. Data Preprocessing is a module shared by both offline training and online detection. During data preprocessing, the dataset is transformed by data standardization, and then it is segmented into sequences through sliding windows [21] of length $T + 1$. After preprocessing, a training multivariate time series, usually spanning a period of time (*e.g.*, a couple of weeks), is sent to Model Training module to learn a model that captures the normal patterns of multivariate time series and outputs an anomaly score for each observation. These anomaly scores are used by the Threshold Selection module to choose an anomaly threshold automatically following the POT method (see later in Section 4.4). This offline training procedure can be conducted routinely, *e.g.*, once per week or month.

The Online Detection module stores the trained model. A new observation (*e.g.*, $\mathbf{x_t}$ at time t) after preprocessing can be fed into Online Detection module to get its anomaly score. If the anomaly score of $\mathbf{x_t}$ is below the anomaly threshold, $\mathbf{x_t}$ will be declared as anomalous, otherwise, it is normal. If $\mathbf{x_t}$ is detected as an anomaly, we interpret it by estimating and ranking the contribution (*i.e.*, reconstruction probability) of each dimension in $\mathbf{x_t}$.

3.3 Basics of GRU, VAE and Planar NF

RNNs [5] are able to represent the time dependence by adopting *deterministic* hidden variables. Simple RNN could fail to learn the long-term dependence in a sequence [1]. RNN variants, LSTM

[1] https://github.com/smallcowbaby/OmniAnomaly

and GRU [1], were invented to address this problem using gating mechanisms. In general, the performance of GRU is as good as LSTM (GRU performs even better than LSTM in some applications [1]), and GRU is more suitable for model training when the datasets are not very large because of its fewer parameters and simpler structure [1]. Thus we apply GRU in *OmniAnomaly* to capture the complex temporal dependence in time series.

VAE is a deep Bayesian model [7], and it has been successfully applied to anomaly detection for seasonal univariate time series [24]. VAE represents a high-dimensional input $\mathbf{x_t}$ to a latent representation $\mathbf{z_t}$ with a reduced dimension, and then reconstructs $\mathbf{x_t}$ by $\mathbf{z_t}$. With a prior $p_\theta(\mathbf{z_t})$ for $\mathbf{z_t}$, $\mathbf{x_t}$ is sampled from posterior distributions $p_\theta(\mathbf{x_t}|\mathbf{z_t})$. However, it is intractable to compute $p_\theta(\mathbf{x_t}|\mathbf{z_t})$, and VAE approximates $p_\theta(\mathbf{x_t}|\mathbf{z_t})$ using an inference network $q_\phi(\mathbf{z_t}|\mathbf{x_t})$, where ϕ and θ are parameters of the inference net (*i.e.*, *qnet*) and the generative net (*i.e.*, *pnet*), respectively.

Stochastic Gradient Variational Bayes (SGVB) [7] is a variational inference algorithm often used in VAE to train the parameters ϕ and θ by maximizing the evidence of lower bound (ELBO), $\mathcal{L}(\mathbf{x_t})$:

$$\begin{aligned}\mathcal{L}(\mathbf{x_t}) &= \mathbb{E}_{q_\phi(\mathbf{z_t}|\mathbf{x_t})}[\log(p_\theta(\mathbf{x_t}|\mathbf{z_t}))] - D_{KL}[q_\phi(\mathbf{z_t}|\mathbf{x_t})||p_\theta(\mathbf{z_t})] \qquad (1)\\ &= \mathbb{E}_{q_\phi(\mathbf{z_t}|\mathbf{x_t})}[\log(p_\theta(\mathbf{x_t}|\mathbf{z_t})) + \log(p_\theta(\mathbf{z_t})) - \log(q_\phi(\mathbf{z_t}|\mathbf{x_t}))]\end{aligned}$$

Monte Carlo integration [19] can be used to compute the above expectation, as shown in Eq. 2, where $\mathbf{z_t}^{(l)}, l = 1, 2...L$ is sampled from the $q_\phi(\mathbf{z_t}|\mathbf{x_t})$.

$$\mathcal{L}(\mathbf{x_t}) \approx \frac{1}{L}\sum_{l=1}^{L}[\log(p_\theta(\mathbf{x_t}|\mathbf{z_t}^{(l)})) + \log(p_\theta(\mathbf{z_t}^{(l)})) - \log(q_\phi(\mathbf{z_t}^{(l)}|\mathbf{x_t}))] \quad (2)$$

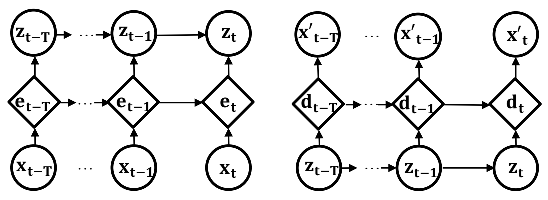
In the *qnet*, $q_\phi(\mathbf{z_t}|\mathbf{x_t})$ is often assumed to be diagonal Gaussian [7], but this simple assumption would make the network underfit because $q_\phi(\mathbf{z_t}|\mathbf{x_t})$ may not necessarily follow Gaussian. To learn the non-Gaussian posterior density $q_\phi(\mathbf{z_t}|\mathbf{x_t})$, [18] proposed a solution named planar NF which transforms $q_\phi(\mathbf{z_t}|\mathbf{x_t})$ using the invertible mappings. We first sample from $q_\phi(\mathbf{z_t}|\mathbf{x_t})$ to get $\mathbf{z_t^0}$. Then through a chain of invertible mappings, we get $\mathbf{z_t^K} = f^K(f^{K-1}(...f^1(\mathbf{z_t^0})))$, where $f^k(k = 1, 2...K)$ are invertible mapping functions ($f^k(\mathbf{z_t^{k-1}}) = \mathbf{z_t^{k-1}} + \mathbf{u}tanh(\mathbf{w}^T\mathbf{z_t^{k-1}} + \mathbf{b})$, $\mathbf{u}$, $\mathbf{w}$ and $\mathbf{b}$ are the parameters) [18]. In the *qnet*, we only take $\mathbf{z_t^K}$, the final output of planar NF, as our stochastic variable $\mathbf{z_t}$ (*i.e.*, $\mathbf{z_t} = \mathbf{z_t^K}$).

4 DESIGN OF OMNIANOMALY

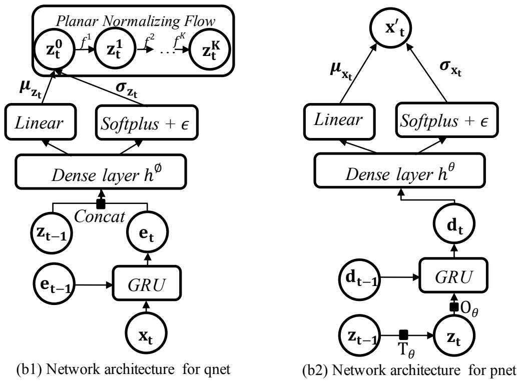
In this section, we first present the network architecture of *OmniAnomaly*, followed by offline model training, online anomaly detection, anomaly threshold selection, and anomaly interpretation.

4.1 Network Architecture

The basic idea of *OmniAnomaly* is the following. First, it uses GRU to capture complex temporal dependence between multivariate observations in $\mathbf{x}$-space. Second, we apply VAE, a popular variational algorithm for representation learning, to map observations (*i.e.*, input observations in $\mathbf{x}$-space) to stochastic (*i.e.*, $\mathbf{z}$-space) variables. Third, inspired by speech reconstruction literature [5], to explicitly model temporal dependence among stochastic variables in the latent space, we propose the stochastic variable connection technique (Linear Gaussian State Space Model (SSM) [8] connection between stochastic variables, and the concatenation of stochastic variable and GRU hidden variable). Forth, to help stochastic variables in



(a) Overall graphical model of *OmniAnomaly* which consists of two parts: (a1) *qnet* and (a2) *pnet*. Nodes correspond to different variables. At time t, $\mathbf{x_t}$ is the input observation and $\mathbf{x'_t}$ is the reconstruction of $\mathbf{x_t}$, $\mathbf{e_t}$ and $\mathbf{d_t}$ are memory variables in GRU cells which are deterministic, $\mathbf{z_t}$ is a z-space variable which is stochastic, and edges represent the dependence between variables.



(b) Detailed network architecture of *OmniAnomaly* at time t.

Figure 3: Overall graphical model and detailed network architecture of *OmniAnomaly*

the *qnet* capture complex distributions of input data, we adopt planar NF [18] which uses a series of invertible mappings to learn non-Gaussian posterior distributions in latent stochastic space.

The overall graphical model of *OmniAnomaly* is shown in Fig. 3(a), which is composed of a *qnet* and a *pnet*. In the *pnet*, it uses a latent representation $\mathbf{z_{t-T:t}}$ (a set of probability distributions) to reconstruct the input $\mathbf{x_{t-T:t}}$. An accurate representation can minimize the reconstruction loss. *qnet* is optimized to approximate the *pnet* and obtain good latent representations.

The details of the *qnet* are shown in Fig. 3(b1). At time t, an input observation $\mathbf{x_t}$ and $\mathbf{e_{t-1}}$ (the hidden variable in GRU at time $t-1$), are sent to a GRU cell to generate the hidden variable $\mathbf{e_t}$ (Eq. 3a). The deterministic $\mathbf{e_t}$ is critical for *OmniAnomaly* to capture long-term complex temporal information between $\mathbf{x_t}$ and its preceding $\mathbf{x}$-space observations. Then, $\mathbf{e_t}$, concatenated with $\mathbf{z_{t-1}}$, enters the dense layer to generate mean $\boldsymbol{\mu}_{\mathbf{z_t}}$ and standard deviation $\boldsymbol{\sigma}_{\mathbf{z_t}}$ for the stochastic variable $\mathbf{z_t}$ (Eq. 3b and 3c). As a result, $\mathbf{z}$-space variables are now temporally dependent, as shown in Fig. 3(a1). The *qnet* can be formulated as follows:

$$\mathbf{e_t} = (1 - \mathbf{c_t^e}) \circ tanh(\mathbf{w^e}\mathbf{x_t} + \mathbf{u^e}(\mathbf{r_t^e} \circ \mathbf{e_{t-1}}) + \mathbf{b^e}) + \mathbf{c_t^e} \circ \mathbf{e_{t-1}} \quad (3a)$$

$$\boldsymbol{\mu}_{\mathbf{z_t}} = \mathbf{w}^{\boldsymbol{\mu}_\mathbf{z}} h^\phi([\mathbf{z_{t-1}}, \mathbf{e_t}]) + \mathbf{b}^{\boldsymbol{\mu}_\mathbf{z}} \quad (3b)$$

$$\boldsymbol{\sigma}_{\mathbf{z_t}} = softplus(\mathbf{w}^{\boldsymbol{\sigma}_\mathbf{z}} h^\phi([\mathbf{z_{t-1}}, \mathbf{e_t}]) + \mathbf{b}^{\boldsymbol{\sigma}_\mathbf{z}}) + \boldsymbol{\epsilon}^{\boldsymbol{\sigma}_\mathbf{z}} \quad (3c)$$

In Eq. 3a-3c, $\mathbf{r_t^e} = sigmod(\mathbf{w}^{\mathbf{r^e}}\mathbf{x_t} + \mathbf{u}^{\mathbf{r^e}}\mathbf{e_{t-1}} + \mathbf{b}^{\mathbf{r^e}})$ is the reset gate determining how to combine a new input with the previous memory. $\mathbf{c_t^e} = sigmod(\mathbf{w}^{\mathbf{c^e}}\mathbf{x_t} + \mathbf{u}^{\mathbf{c^e}}\mathbf{e_{t-1}} + \mathbf{b}^{\mathbf{c^e}})$ is the update gate deciding how much previous memory need to keep.

As shown in Fig. 3(b1), $[\mathbf{z_{t-1}}, \mathbf{e_t}]$ is the concatenation of $\mathbf{z_{t-1}}$ and $\mathbf{e_t}$. h^{ϕ} denotes the dense layer with ReLU activation function. $\boldsymbol{\mu}_{\mathbf{z_t}}$ is derived from a linear layer and $\boldsymbol{\sigma}_{\mathbf{z_t}}$ is produced by the soft-plus activation function with small $\boldsymbol{\epsilon}$ to prevent numerical overflow. All the $\mathbf{u}^*$-s, $\mathbf{w}^*$-s, $\mathbf{b}^*$-s are the parameters of the corresponding layers. The output of the *qnet*, $\mathbf{z_t^0}$, is diagonal Gaussian and sampled from $\mathcal{N}(\boldsymbol{\mu}_{\mathbf{z_t}}, \boldsymbol{\sigma}^2_{\mathbf{z_t}}\mathbf{I})$. To learn a non-Gaussian posterior distribution of $q_{\phi}(\mathbf{z_t}|\mathbf{x_t})$, we use planar NF to approximate $\mathbf{z_t}$. As shown at the top of Fig. 3(b1), $\mathbf{z_t}$ (*i.e.*, $\mathbf{z_t^K}$) is obtained by passing the $\mathbf{z_t^0}$ through a chain of K transformations f^k which are planar mappings [18].

The *pnet*, as shown in Fig. 3(b2), attempts to reconstruct $\mathbf{x_t}$ with $\mathbf{z_t}$ using a structure similar to the *qnet*. We utilize linear Gaussian SSM [8] to "connect" $\mathbf{z}$-space variables in *qnet* and make them temporally dependent: $\mathbf{z_t} = \mathbf{O}_{\boldsymbol{\theta}}(\mathbf{T}_{\boldsymbol{\theta}}\mathbf{z_{t-1}} + \mathbf{v_t}) + \boldsymbol{\epsilon}_\mathbf{t}$, where $\mathbf{T}_{\boldsymbol{\theta}}$ and $\mathbf{O}_{\boldsymbol{\theta}}$ are transition and observation matrices [8], $\mathbf{v_t}$ and $\boldsymbol{\epsilon}_\mathbf{t}$ are transition and observation noises. At time t, $\mathbf{z_t}$, along with the variable $\mathbf{d_{t-1}}$ at time t-1, is passed through a GRU cell to produce the deterministic variable $\mathbf{d_t}$ (Eq. 4a). Then $\mathbf{d_t}$ is further processed through the dense layer to generate the mean $\boldsymbol{\mu}_{\mathbf{x_t}}$ and standard deviation $\boldsymbol{\sigma}_{\mathbf{x_t}}$ of variable $\mathbf{x_t'}$ (reconstruction of $\mathbf{x_t}$) (Eq. 4b and 4c).

Similar to the *qnet*, the *pnet* can be formulated as follows:

$$\mathbf{d_t} = (1 - \mathbf{c_t^d}) \circ tanh(\mathbf{w^d z_t} + \mathbf{u^d}(\mathbf{r_t^d} \circ \mathbf{d_{t-1}}) + \mathbf{b^d}) + \mathbf{c_t^d} \circ \mathbf{d_{t-1}} \tag{4a}$$

$$\boldsymbol{\mu}_{\mathbf{x_t}} = \mathbf{w}^{\boldsymbol{\mu}_\mathbf{x}} h^{\theta}(\mathbf{d_t}) + \mathbf{b}^{\boldsymbol{\mu}_\mathbf{x}} \tag{4b}$$

$$\boldsymbol{\sigma}_{\mathbf{x_t}} = softplus(\mathbf{w}^{\boldsymbol{\sigma}_\mathbf{x}} h^{\theta}(\mathbf{d_t}) + \mathbf{b}^{\boldsymbol{\sigma}_\mathbf{x}}) + \boldsymbol{\epsilon}^{\boldsymbol{\sigma}_\mathbf{x}} \tag{4c}$$

where $\mathbf{r_t^d} = sigmod(\mathbf{w^{r^d} z_t} + \mathbf{u^{r^d} d_{t-1}} + \mathbf{b^{r^d}})$ and $\mathbf{c_t^d} = sigmod(\mathbf{w^{c^d} z_t} + \mathbf{u^{c^d} d_{t-1}} + \mathbf{b^{c^d}})$, which are the reset and update gate, respectively.

The reconstructed data $\mathbf{x_t'}$ is sampled from $\mathcal{N}(\boldsymbol{\mu}_{\mathbf{x_t}}, \boldsymbol{\sigma}^2_{\mathbf{x_t}}\mathbf{I})$ and created from $\mathbf{z_t}$. If there is an anomaly at time t, $\mathbf{x_t'}$ may vary significantly from the original data $\mathbf{x_t}$. Therefore, we can detect anomalies based on the reconstruction probability of $\mathbf{x_t}$.

4.2 Offline Model Training

The *qnet* and *pnet* in *OmniAnomaly* are trained simultaneously by tuning the network parameters ($\mathbf{u}^*$-s, $\mathbf{w}^*$-s, and $\mathbf{b}^*$-s). Similar to VAE models, we can train our model straightforwardly by optimizing ELBO, as described in Section 3.3. The length of each input sequence data (*e.g.*, $\mathbf{x_{t-T:t}}$) in training dataset is T+1. For the l-th sample $\mathbf{z^{(l)}_{t-T:t}}$, where $1 \le l \le L$ and L is the sample length, the loss function can be formulated as:

$$\widetilde{\mathcal{L}}(\mathbf{x_{t-T:t}}) \approx \frac{1}{L}\sum_{l=1}^{L}[\log(p_\theta(\mathbf{x_{t-T:t}}|\mathbf{z^{(l)}_{t-T:t}})) + \log(p_\theta(\mathbf{z^{(l)}_{t-T:t}})) - \log(q_\phi(\mathbf{z^{(l)}_{t-T:t}}|\mathbf{x_{t-T:t}}))] \tag{5}$$

For each sample, the first term of Eq. 5 is the negative reconstruction error: $\log(p_\theta(\mathbf{x_{t-T:t}}|\mathbf{z_{t-T:t}})) = \sum_{i=t-T}^{t}\log(p_\theta(\mathbf{x_i}|\mathbf{z_{t-T:i}}))$, and the posterior probability of $\mathbf{x_i}$ can be calculated as: $p_\theta(\mathbf{x_i}|\mathbf{z_{t-T:i}}) \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_i}, \boldsymbol{\sigma}^2_{\mathbf{x}_i}\mathbf{I})$. The sum of the second and third terms is regularization (*i.e.*, Kullback-Leibler loss). The second term $\log(p_\theta(\mathbf{z_{t-T:t}})) = \sum_{i=t-T}^{t}\log(p_\theta(\mathbf{z_i}|\mathbf{z_{i-1}}))$, where $\mathbf{z_i}$ can be obtained by Linear Gaussian SSM [8] initialized with the standard multivariate normal distribution. The third term is to approximate the true posterior distribution of $\mathbf{z_i}$ in the $\mathbf{z}$-space in the *qnet*: $-\log(q_\phi(\mathbf{z_{t-T:t}}|\mathbf{x_{t-T:t}})) = -\sum_{i=t-T}^{t}\log(q_\phi(\mathbf{z_i}|\mathbf{z_{i-1}}, \mathbf{x_{t-T:i}}))$. $\mathbf{z_i}$ (*i.e.*, $\mathbf{z_i^K}$) is transformed through planar NF. $\mathbf{z_i^K} = f^K(f^{K-1}(...f^1(\mathbf{z_i}^0)))$, where $\mathbf{z_i^0} = \boldsymbol{\mu}_{\mathbf{z_i}} + \boldsymbol{\xi}_i\boldsymbol{\sigma}_{\mathbf{z_i}}$, $\boldsymbol{\xi}_i \sim \mathcal{N}(0, \mathbf{I})$, and the formulation of f^k can be seen in Section 3.3.

4.3 Online Detection

Now we can determine whether an observation at a time step (say t, denoted as $\mathbf{x_t}$) is anomalous or not using the trained *OmniAnomaly* model. Note that the input of *OmniAnomaly* is a sequence data of length T+1. Thus, we take the sequence $\mathbf{x_{t-T:t}}$, *i.e.*, $\mathbf{x_t}$ and T consecutive observations preceding to it, as an input to reconstruct $\mathbf{x_t}$. As suggested by [24], this reconstruction can be evaluated by the conditional probability $\log(p_\theta(\mathbf{x_t}|\mathbf{z_{t-T:t}}))$. This reconstruction probability is used as the anomaly score in our model. The anomaly score of $\mathbf{x_t}$ is denoted as S_t, so $S_t = \log(p_\theta(\mathbf{x_t}|\mathbf{z_{t-T:t}}))$. A high score means the input $\mathbf{x_t}$ can be well reconstructed. If an observation follows the normal patterns of time series, it can be reconstructed with high confidence. On the other hand, the smaller the score, the less likely the observation can be reconstructed and thus it is more likely to be anomalous. Formally, if S_t is lower than an anomaly threshold, then $\mathbf{x_t}$ is marked as anomalous; otherwise $\mathbf{x_t}$ is normal. Next, we describe how to automatically determine the anomaly threshold offline.

4.4 Automatic Threshold Selection

As shown in Fig. 2, *during offline training*, with a multivariate time series of N' observations, we compute an anomaly score for every observation. Then all anomaly scores form a univariate time series $\{S_1, S_2, ..., S_{N'}\}$. Next, we set the anomaly threshold th_F offline following the principle of Extreme Value Theory (EVT) [22].

EVT is a statistical theory whose goal is to find the law of extreme values, and extreme values are usually placed at the tails of a probability distribution. The advantage of EVT is that it makes no assumption on data distribution when finding extreme values. Peaks-Over-Threshold (POT) [22] is the second theorem in EVT. The basic idea of POT is to fit the tail portion of a probability distribution by a generalized Pareto distribution (GPD) with parameters. We adopt POT to learn the threshold of anomaly scores. Unlike classical POT applications which focus on "values at the high end of a distribution", anomalies in our analysis are located at the low end of the distribution. So we adapt the GPD function as follows:

$$\bar{F}(s) = P(th - S > s|S < th) \sim (1 + \frac{\gamma s}{\beta})^{-\frac{1}{\gamma}} \tag{6}$$

where th is the initial threshold of anomaly scores, γ and β are shape and scale parameters of GPD, S is any value in $\{S_1, S_2, ..., S_{N'}\}$. The portion below a threshold th is denoted as $th - S$, and it is empirically set to a low quantile. Similar to [22], we estimate the values of parameters $\hat{\gamma}$ and $\hat{\beta}$ by Maximum Likelihood Estimation (MLE). The final threshold th_F is then computed by:

$$th_F \simeq th - \frac{\hat{\beta}}{\hat{\gamma}}((\frac{qN'}{N'_{th}})^{-\hat{\gamma}} - 1) \tag{7}$$

where q is the desired probability to observe $S < th$, N' is the number of observations, and N'_{th} is the number of S_i s.t. $S_i < th$. For POT method, there are only two parameters (low quantile and q) that need to be tuned. These two parameters are model-wide and can be set empirically: low quantile (*e.g.*, less than 7%) and q (*e.g.*, 10^{-4}) [22], seeing Appendix B.

4.5 Anomaly Interpretation

As described in Section 1, the goal of our anomaly interpretation solution is to annotate the detected entity anomaly with the top few univariate time series ranked by their reconstruction probabilities.

Thus, we have to obtain the reconstruction probability for individual x_t^i (the i-th dimension of $\mathbf{x_t}$). However, in *OmniAnomaly*, the reconstruction probability is calculated for the M-dimensional $\mathbf{x_t}$. Fortunately, as presented in Section 4.2, $p_\theta(\mathbf{x_t}|\mathbf{z_{t-T:t}}) \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_t}, \boldsymbol{\sigma}^2_{\mathbf{x}_t}\mathbf{I})$, thus, $p_\theta(\mathbf{x_t}|\mathbf{z_{t-T:t}}) = \prod_{i=1}^{M} p_\theta(x_t^i|\mathbf{z_{t-T:t}})$. Therefore, the conditional probability of $\mathbf{x_t}$ can be factorized as:

$$\log(p_\theta(\mathbf{x_t}|\mathbf{z_{t-T:t}})) = \sum_{i=1}^{M} \log(p_\theta(x_t^i|\mathbf{z_{t-T:t}})) \qquad (8)$$

$S_t = \sum_{i=1}^{M} S_t^i$, where $S_t^i = \log(p_\theta(x_t^i|\mathbf{z_{t-T:t}})$ and S_t^i is the anomaly score of x_t^i. Note that S_t^i benefits from the rich information in the multivariate time series $\mathbf{x_{t-T:t-1}}$, thus its interpretation power is higher than the anomaly score that is obtained by just utilizing $x_{t-T:t-1}^i$ (as in univariate time series).

For a detected anomaly $\mathbf{x_t}$, we interpret it by estimating the contribution (*i.e.*, reconstruction probability) of each dimension of $\mathbf{x_t}$. We sort S_t^i $(1 \leq i \leq M)$ in ascending order and form the list AS_t. For x_t^i, the higher ranked in AS_t, the smaller S_t^i, and the greater contribution of x_t^i to $\mathbf{x_t}$. The ordered list is presented to the operators as the anomaly interpretation, and hopefully the top few dimensions can provide sufficient clues for operators to understand and troubleshoot the detected entity anomaly.

Note that the factorization Eq. 8 holds as long as $p_\theta(\mathbf{x_t}|\mathbf{z_{t-T:t}}) \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_t}, \boldsymbol{\sigma}^2_{\mathbf{x}_t}\mathbf{I})$ holds. Therefore, our proposed anomaly interpretation approach is applicable to other multivariate time series anomaly detection algorithms such as [16].

5 EVALUATION

In this section, we first describe experimental datasets and performance metrics. Then, we conduct many experiments to show the effectiveness of our model.

5.1 Datasets and Performance Metrics

To demonstrate the effectiveness of *OmniAnomaly*, we conduct experiments on three datasets: SMD (Server Machine Dataset), SMAP (Soil Moisture Active Passive satellite) and MSL (Mars Science Laboratory rover). More details can be seen in Appendix A.

We use Precision, Recall, F1-Score (denoted as F1) to evaluate the performance of *OmniAnomaly* and baseline models: F1 = $\frac{2 \times Precision \times Recall}{Precision + Recall}$, where Precision= $\frac{TP}{TP+FP}$, Recall= $\frac{TP}{TP+FN}$. Some anomaly detection models provide methods to choose anomaly thresholds, and thus their F1 can be calculated based on the selected thresholds accordingly. In case a model gives no specific way to select thresholds, or we want to calculate a model's best F1 in theory, we enumerate all possible anomaly thresholds to search for the best F1, denoted as F1_{best}, in contrast with F1.

In practice, anomalous observations usually occur continuously to form contiguous anomaly segments. It is acceptable if an alert for anomalies is triggered within any subset of a ground truth anomaly segment. Thus, a point-adjust approach was proposed by [24] to calculate the performance. For any observation in the ground truth anomaly segment, if it is detected as an anomaly, we think this segment is detected correctly and all observations in the segment are considered to have been correctly detected as anomalies. The observations outside the ground truth anomaly segment are treated as usual. We adopt the point-adjust way to calculate the performance metrics in our paper.

5.2 Results and Analysis

5.2.1 *OmniAnomaly vs. other approaches.*

To demonstrate the effectiveness of *OmniAnomaly*, we first compare it with four state-of-the-art unsupervised approaches for multivariate time series anomaly detection: LSTM with nonparametric dynamic thresholding (LSTM-NDT for short) [6], EncDec-AD [11], DAGMM [27], and LSTM-VAE [16], which have been described in Section 2. Moreover, we select Donut [24], a state-of-the-art univariate time series anomaly detection approach based on VAE, as another baseline. To apply Donut [24] to multivariate time series anomaly detection, we use a simple rule to define entity-level anomalies as follows: *for an M-dimensional entity, at time t, if there are at least M' $(1 \leq M' \leq M$, where M' can be tuned by different datasets) univariate time series that are anomalous, then the entity is declared as anomalous.*

Table 2 shows the precision, recall, F1 of LSTM-NDT, DAGMM, LSTM-VAE and *OmniAnomaly* on three datasets and Total dataset (*i.e.*, union of these three datasets). Each of these approaches provides a specific method for us to choose anomaly thresholds and F1 is calculated accordingly. *OmniAnomaly* outperforms all baselines on MSL and SMD datasets, and its F1 is only slightly lower than the best baseline on SMAP dataset. Also, *OmniAnomaly* exceeds the best performing state-of-the-art method (*i.e.*, LSTM-NDT) by 0.09 on the F1 for Total dataset. *OmniAnomaly*'s robustness is better than baseline models in that the precision and recall of *OmniAnomaly* are both higher than 0.74 for *all* three datasets, which is not achieved by any baseline approaches.

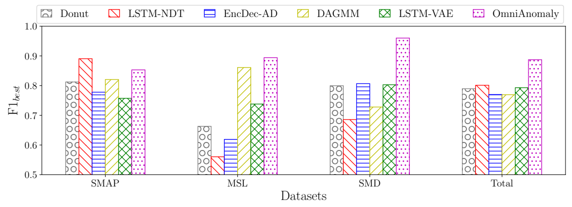


Figure 4: F1_{best} of *OmniAnomaly* and all baseline models.

Since Donut [24] and EncDec-AD [11] provide no specific methods for choosing anomaly thresholds, they are not shown in Table 2. Instead, they are evaluated using F1_{best}, as shown in Fig. 4. Again, *OmniAnomaly* excels all baseline models at MSL and MSD datasets and is ranked the second at SMAP dataset. In particular, *OmniAnomaly* outperforms the best performing state-of-the-art method (*i.e.*, LSTM-NDT) by 0.086 on F1_{best} (0.8871 vs. 0.8012) for Total dataset. For those algorithms already listed in Table 2, there is no significant difference between their F1 and F1_{best}.

Overall, these experimental results demonstrate the superiority of *OmniAnomaly* compared with the state-of-the-art approaches. Next, we analyze the performance of these methods in detail.

Donut [24] aims at univariate time series anomaly detection. We use the aforementioned rule combining the anomaly result of each univariate time series to do multivariate time series anomaly detection. For each dataset, we enumerate all values to get the M' which can make Donut perform the best, and the values of M' for SMAP, MSL, SMD are 1, 10, 5 respectively. We can see that, M' varies by different datasets, because it heavily relies on characteristics of the datasets. In reality, for each dataset, it would be challenging and requires extensive domain knowledge to choose a proper M', and one specific rule may not be appropriate for all

Table 2: Performance of *OmniAnomaly* and 3 baseline approaches. F1$_{best}$ of Donut [24] and EncDec-AD [11] are in Fig. 4.

Methods	SMAP			MSL			SMD			Total		
	P (Precision)	R (Recall)	F1	P	R	F1	P	R	F1	P	R	F1
LSTM-NDT [6]	0.8965	0.8846	**0.8905**	0.5934	0.5374	0.5640	0.5684	0.6438	0.6037	0.7598	0.7794	**0.7694**
DAGMM [27]	0.5845	0.9058	0.7105	0.5412	0.9934	**0.7007**	0.5951	0.8782	0.7094	0.5835	0.9042	0.7093
LSTM-VAE [16]	0.8551	0.6366	0.7298	0.5257	0.9546	0.6780	0.7922	0.7075	**0.7842**	0.7782	0.7075	0.7411
OmniAnomaly	0.7416	0.9776	**0.8434**	0.8867	0.9117	**0.8989**	0.8334	0.9449	**0.8857**	0.7797	0.9586	**0.8599**

anomaly types because of the diversity of the faults. Donut performs the worst on MSL, because MSL has more univariate time series and anomalous types than SMAP and SMD. These observations confirm our intuition that we should model the multivariate time series as an entity, instead of examining each univariate time series individually, to understand their behaviors.

LSTM-NDT [6] is a deterministic model without leveraging stochastic information. Stochastic information can improve model performance because it can learn the inherent stochasticity of time series [5]. As a stochastic model, *OmniAnomaly* performs better than LSTM-NDT. Moreover, LSTM-NDT is a prediction-based model. However, some time series are inherently unpredictable [11] due to external factors. As a result, the prediction of time series may not be accurate and the prediction-based models may not be appropriate. According to [6], for MSL, a wide variety of behaviors with varying regularity make it hard to be predicted. Some time series (*e.g.*, TCP retransmissions, the second univariate time series in Fig. 1) in SMD are also unpredictable because of uncontrollable factors (*e.g.*, complex and dynamic network environment). Therefore, LSTM-NDT does not perform well on these two datasets. *OmniAnomaly* is a reconstruction-based model which aims at learning normal patterns of multivariate time series, so it works well with both predictive and unpredictable time series.

EncDec-AD [11] is a seq2seq model based on encoder-decoder reconstruction. This model uses the final hidden variable of its encoder as the initial hidden variable of its decoder. When the length or dimension of an input sequence is large, it is difficult for the final hidden variable in the encoder to remember all the information of the entire sequence such that the input can be perfectly reconstructed. Thus it is not surprising that EncDec-AD performs the worst on MSL, because the dimension of MSL is larger than the other two datasets, and the final hidden variable in the encoder may not be able to remember sufficient information. In addition, EncDec-AD's hidden layer is composed of LSTM units, which are not able to handle stochastic information. This deterministic nature perhaps also contributes to its inferior performance.

DAGMM [27] focuses on anomaly detection for multivariate data *without* temporal information between observations. The input of DAGMM is just one observation (*i.e.*, multivariate observation) instead of a sequence of T+1 observations. However, for multivariate time series, the temporal information is important and necessary, because observations are dependent and historical data is helpful in reconstructing current observations. For example, without considering temporal information, the observation around 40-th hour in Fig. 1 would be easily mistaken as anomalous by DAGMM and causes a false positive, because it has a low probability among all multivariate observations. In our model, for both training and detection, the input is a sequence of observations which contains the temporal relationship in time series. As a result, *OmniAnomaly* performs better than DAGMM.

LSTM-VAE [16] simply combines LSTM and VAE by replacing the feed-forward network in a VAE with LSTM. As suggested by [2], for sequential data modeling, in order to well represent the input data, it is beneficial to include information coming from $\mathbf{z_{t-1}}$ as part of $\mathbf{z_t}$. However, LSTM-VAE does not consider the temporal dependence among stochastic (*i.e.*, $\mathbf{z}$-space) variables. This explains its worse performance compared to *OmniAnomaly*.

Summary. Compared with deterministic approaches like LSTM-NDT and EncDec-AD, *OmniAnomaly* is a deep Bayesian network which extends the modeling capabilities of recurrent neural networks with stochastic variables. Moreover, *OmniAnomaly* is a reconstruction based model that can work well regardless of the predictability of the multivariate time series. Unlike DAGMM, *OmniAnomaly* works well with the temporal dependence of time series data by GRU. Besides, compared with LSTM-VAE, *OmniAnomaly* captures the dependence of the stochastic variables through the $\mathbf{z}$-space variable connection such that our $\mathbf{z}$-space layer can better represent the distributions of input data. Moreover, planar NF is also helpful in constructing $\mathbf{z}$-space variables in the *qnet*. The effectiveness of $\mathbf{z}$-space variable connection and planar NF will be described shortly.

5.2.2 *Effects of major techniques in OmniAnomaly.*

In this section, we experimentally show the effects of four major techniques in *OmniAnomaly*: (1) GRU; (2) $\mathbf{z}$-space variable connection; (3) planar NF; (4) POT method for automatic anomaly threshold selection.

We reconfigure *OmniAnomaly* to create four categories of variants, denoted as C1-C4, described as follows. (C1) In this category, the GRU in *OmniAnomaly* is replaced by a simple RNN, denoted as "C1-RNN", or by LSTM, denoted as "C1-LSTM". (C2) The $\mathbf{z}$-space variables are connected in *qnet* only (denoted as "C2-qnet"), in *pnet* only (denoted as "C2-pnet"), or in neither net (denoted as "C2-no z connected"). (C3) Planar NF in *qnet* of *OmniAnomaly* is replaced by Gaussian function (*i.e.*, "C3-no planar NF"). (C4) Instead of applying the POT method to set a threshold automatically, here we enumerate thresholds to obtain F1$_{best}$. For fair comparison, the results for variants in Categories C1-C3 are all evaluated by F1$_{best}$.

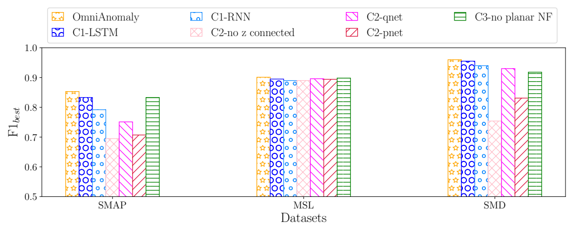


Figure 5: F1$_{best}$ of *OmniAnomaly* and the variants in C1-C3.

Effect of GRU. From Fig. 5, we can see that "C1-RNN" is inferior to "C1-LSTM" and *OmniAnomaly*, because a simple RNN is not able to capture long-term dependence of time series. Moreover, *OmniAnomaly* is slightly better than "C1-LSTM". This may be possibly explained by the fact that GRU has fewer parameters and simpler structure [1] than LSTM so it is easier for model training. Thus, we choose GRU to capture complex temporal dependence of multivariate time series (*i.e.*, **x**-space) in our model.

Effect of **z***-space variable connection.* Explicitly modeling the temporal dependence of **z**-space variables is an indispensable technique in our model, and it is critical for latent representations to learn the normal patterns of input data. Fig. 5 shows the performance of *OmniAnomaly* and its three variants in C2. We can see that with **z**-space variable connection both in *qnet* and *pnet*, *OmniAnomaly* performs better than the other three variants. **z**-space variable connection in *qnet* can help the stochastic variables in *qnet* capture more information from historical stochastic variables so that they can represent the input data better. Rather than simply choosing $\mathcal{N}(0, \mathbf{I})$ as the prior of **z**-space variables, *OmniAnomaly* can fit the prior expectation of a data distribution better by applying Linear Gaussian SSM for **z**-space variable connection in *pnet*. MSL contains substantial one-hot encoded data and its anomalies involve dramatic changes. As a result, it is easy to reconstruct the input data and detect anomalies. Thus, the benefit of **z**-space variable connection in MSL is not significant.

Effect of Planar NF. Planar NF can capture complicated data patterns and help construct **z**-space variables in *qnet*. Fig. 5 compares *OmniAnomaly* and "C3-no planar NF". We can see that, the performance of *OmniAnomaly* is improved by using planar NF. Instead of assuming the approximate posterior distribution in *qnet* is Gaussian [24], planar NF transforms an initial distribution (*e.g.*, Gaussian) with a sequence of invertible mappings, so it can capture complex and flexible distributions of input data [18] and improve the performance of our model. In MSL, the effect of planar NF is not as significant as in the other two datasets, because simple **z**-space variables can already represent its sparse input data.

Table 3: F1 obtained through POT vs. $F1_{best}$.

Evaluation metrics for *OmniAnomaly*	SMAP	MSL	SMD
F1 obtained through POT	0.8434	0.8989	0.8857
$F1_{best}$	0.8535	0.9014	0.9620

Effect of POT method. An effective method for anomaly threshold selection is very necessary and useful in practical applications. In *OmniAnomaly*, we apply the POT method to set the threshold automatically. From Table 3, we can see that, F1 obtained through POT is only slightly lower than $F1_{best}$ (0.003~0.077), indicating that POT method is effective for anomaly threshold selection.

5.2.3 Performance of anomaly interpretation. For a detected anomaly $\mathbf{x_t}$, *OmniAnomaly* estimates the contribution (*i.e.* reconstruction probability) of $\mathbf{x_t}$'s each dimension to this anomaly and record all dimensions into a list, AS_t, ordered by their contributions. Let GT_t be the ground truth array containing the dimensions indeed contributing to anomaly $\mathbf{x_t}$. Since there exist no established metrics to evaluate the interpretability of entity anomalies, motivated by the idea of HitRate@K for recommender systems [25], we define a new metric HitRate@P%= $\frac{Hit@\lfloor P\% \times |GT_t| \rfloor}{|GT_t|}$, where$|GT_t|$ is the length of GT_t and P can be 100 or 150. Hit@P% equals the number of overlapping dimensions between GT_t (ground truth) and the top $\lfloor P\% \times |GT_t| \rfloor$ contributing dimensions in AS_t suggested by *OmniAnomaly*. We give a toy example to explain HitRate@P%. For a 6-dimensional observation $\mathbf{x_t}$, its AS_t is $\{2, 3, 6, 1, 5, 4\}$ and GT_t is $\{2, 6\}$. The result is 0.5 for HitRate@100% and 1.0 for HitRate@150%.

There is no ground truth provided for SMAP and MSL for anomaly interpretation, thus we evaluate *OmniAnomaly* only on SMD. The average interpretation accuracy for all detected anomalies is: HitRate@100%=0.8002 and HitRate@150%=0.8919, which demonstrates that *OmniAnomaly* can give reasonable interpretation for anomalies in practice. Similarly, we can also evaluate the interpretability of entity anomalies detected by LSTM-VAE in SMD dataset: HitRate@100%=0.5046 and HitRate@150%=0.6239. *OmniAnomaly* achieves better anomaly interpretability than LSTM-VAE, because our **z**-space variables can represent the input data more robustly to improve reconstruction.

6 DISCUSSION

6.1 Visualization on z-space representations

In this section, we explain how *OmniAnomaly* works for anomaly detection through visualizing the **z**-space representations. *OmniAnomaly* is a reconstruction-based model. For an input observation, *OmniAnomaly* compresses it to a low dimension **z**-space representation and then uses the representation to reconstruct it. During model training, *OmniAnomaly* learns the representations of normal behaviors of the training data. If an input observation is anomalous, its **z**-space representation and the reconstructed value are still normal, so the reconstruction probability is low.

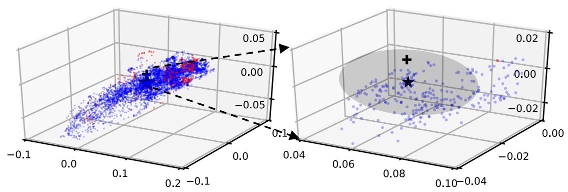


Figure 6: (Left) The 3-dimensional z-space variables of SMD by *OmniAnomaly* (where red points are from anomalous class and blue ones are from normal class). (Right) Randomly choose a normal observation $\mathbf{x_t}$, "★" denotes $\mu_{\mathbf{z_t}}$ and the ellipsoid denotes its 2-$\sigma_{\mathbf{z_t}}$ region; $\mathbf{x_t}$ is set to zeros to obtain an anomalous sample, denoted by "+" in z-space.

The left figure in Fig. 6 shows the 3-d **z**-space variables learned from SMD by *OmniAnomaly*. All **z**-space variables are sampled from $q_\phi(\mathbf{z_t}|\mathbf{x_{t-T:t}})$. We find anomalous samples highly overlap with normal samples, indicating that their **z**-space representations are quite similar. Following [24], we randomly select a normal observation and change its values to make it an anomaly, *i.e.*, setting its values to all zeros, with a change magnitude equal to 385, 069 times of **x**'s standard deviations. As shown in the right figure in Fig. 6, the **z**-space variables of the original observation and the manipulated one are still very close. This further demonstrates that, despite anomalies, their **z**-space variables learned by *OmniAnomaly* are still normal. These examples illustrate **z**-space variables in *OmniAnomaly* capture the observations' normal patterns well.

6.2 Lessons Learned

In this study, we learned three lessons that are in general potentially applicable to the robust modeling of those complex time series/sequence data (*e.g.*, speech, music, and monitoring) with both temporal dependence and stochasticity. First, a combination of stochastic deep Bayesian model and deterministic RNN model is necessary. Second, the connection of stochastic variables is necessary and effective. In particular, *both* the concatenation of RNN hidden variable and **z**-space variable in *qnet and* the Linear Gaussian SSM connection of **z**-space variables in *pnet* help learn more information from historical stochastic variables, and thus improves the quality of latent representations. Third, it is necessary to assume non-Gaussian distributions in **z**-space, which can be learned through flow models such as normalizing flows.

The lessons learned for multivariate time series anomaly detection are the following. First, reconstruction-based models are more robust than prediction-based models, as the time series data in practice could be unpredictable. Second, for reconstruction-based models, it is critical to obtain *robust* latent representations which can accurately capture the normal patterns of time series. Third, reconstruction-based stochastic approaches (*e.g.*, *OmniAnomaly* and[16]) offer an opportunity to interpret the anomalies with physical significance, based on the reconstruction probabilities of the anomalous observation's individual dimensions.

7 CONCLUSION

Entity-level anomaly detection can greatly help operation engineers discover and troubleshoot abnormal behaviors of devices timely. In this paper, we propose *OmniAnomaly*, a novel stochastic recurrent neural network for multivariate time series anomaly detection that works well robustly for various devices. We believe its key techniques, such as stochastic variable connection, are applicable to other time series modeling tasks. Moreover, *OmniAnomaly* provides an intuitive and effective way to interpret detected entity anomalies, based on reconstruction probability. Through extensive experiments, *OmniAnomaly* outperforms state-of-the-art approaches on three large datasets. *OmniAnomaly*'s excellent performance on each dataset also demonstrates that it is a robust model and can be applied to various devices such as server machines and spacecrafts.

8 ACKNOWLEDGMENT

We thank Wenxiao Chen, Haowen Xu, Weibin Meng, Ping Liu and Nengwen Zhao for their helpful discussions on this work. We thank Juexing Liao for proofreading this paper. This work has been supported by the Beijing National Research Center for Information Science and Technology (BNRist) key projects, and the Okawa Research Grant.

REFERENCES

[1] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[2] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*. 2980–2988.

[3] Dhivya Eswaran, Christos Faloutsos, Sudipto Guha, and Nina Mishra. 2018. Spotlight: Detecting anomalies in streaming graphs. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining*. ACM, 1378–1386.

[4] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. 2016. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. *arXiv preprint arXiv:1612.06676* (2016).

[5] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. 2016. Sequential neural models with stochastic layers. In *Advances in neural information processing systems*. 2199–2207.

[6] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. 2018. Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &#38; Data Mining (KDD '18)*. ACM, New York, NY, USA, 387–395.

[7] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of the Second International Conference on Learning Representations (ICLR 2014)*.

[8] Genshiro Kitagawa and Will Gersch. 1996. Linear Gaussian State Space Modeling. In *Smoothness Priors Analysis of Time Series*. Springer, 55–65.

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

[10] Chen Luo, Jian-Guang Lou, Qingwei Lin, Qiang Fu, Rui Ding, Dongmei Zhang, and Zhe Wang. 2014. Correlating events with time series for incident diagnosis. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1583–1592.

[11] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. LSTM-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148* (2016).

[12] Emaad Manzoor, Sadegh M Milajerdi, and Leman Akoglu. 2016. Fast memory-efficient anomaly detection in streaming heterogeneous graphs. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1035–1044.

[13] José-Antonio Martínez-Heras and Alessandro Donati. 2014. Enhanced telemetry monitoring with novelty detection. *AI Magazine* 35, 4 (2014), 37–46.

[14] Vinod Nair, Ameya Raul, Shubhabh Khanduja, Vikas Bahirwani, Qihong Shao, Sundararajan Sellamanickam, Sathiya Keerthi, Steve Herbert, and Sudheer Dhulipalla. 2015. Learning a hierarchical monitoring system for detecting and diagnosing service issues. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2029–2038.

[15] Animesh Nandi, Atri Mandal, Shubham Atreja, Gargi B Dasgupta, and Subhrajit Bhattacharya. 2016. Anomaly detection using program control flow graph mining from execution logs. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 215–224.

[16] Daehyung Park, Yuuna Hoshi, and Charles C. Kemp. 2018. A Multimodal Anomaly Detector for Robot-Assisted Feeding Using an LSTM-Based Variational Autoencoder. *IEEE Robotics and Automation Letters* 3 (2018), 1544–1551.

[17] Daehyung Park, Hokeun Kim, Yuuna Hoshi, Zackory Erickson, Ariel Kapusta, and Charles C Kemp. 2017. A multimodal execution monitor with anomaly classification for robot-assisted feeding. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 5406–5413.

[18] Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational Inference with Normalizing Flows. *Computer Science* (2015), 1530–1538.

[19] Christian Robert and George Casella. 2013. *Monte Carlo statistical methods*. Springer Science & Business Media.

[20] A. Rodriguez, D. Bourne, M. Mason, G. F. Rossano, and J. Wang. 2010. Failure detection in assembly: Force signature analysis. In *2010 IEEE International Conference on Automation Science and Engineering*. 210–215.

[21] Terrence J Sejnowski and Charles R Rosenberg. 1987. Parallel networks that learn to pronounce English text. *Complex systems* 1, 1 (1987), 145–168.

[22] Alban Siffer, Pierre-Alain Fouque, Alexandre Termier, and Christine Largouet. 2017. Anomaly detection in streams with extreme value theory. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1067–1075.

[23] Fei Wu, Pranay Anchuri, and Zhenhui Li. 2017. Structural event detection from log messages. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1175–1184.

[24] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. 2018. Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 187–196.

[25] Xiwang Yang, Harald Steck, Yang Guo, and Yong Liu. 2012. On top-k recommendation using social networks. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 67–74.

[26] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. 2012. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 5, 5 (2012), 363–387.

[27] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*.

A DATASETS

SMAP (Soil Moisture Active Passive satellite) and MSL (Mars Science Laboratory rover) are two public datasets from NASA [6]. Each dataset has a training and a testing subsets, and anomalies in both testing subsets have been labeled [6]. SMD (Server Machine Dataset) is a new 5-week-long dataset which was collected by us from a large Internet company, and it was publicly published on Github. We divided the SMD into two subsets of equal size: the first half is the training set and the second half is the testing set. Anomalies and their anomalous dimensions in SMD testing set have been labeled by domain experts based on incident reports.

Table 4: Dataset Information

Dataset name	No. of entities	No. of dimensions	Training set size	Testing set size	Anomaly ratio(%)
SMAP	55	25	135183	427617	13.13
MSL	27	55	58317	73729	10.72
SMD	28	38	708405	708420	4.16

The observations in these three datasets are all equally-spaced 1 minute apart. Table 4 shows the details of these datasets, including name, the number of entities, the number of dimensions of each observation, size (number of observations) of the training and testing sets, and the ratio of anomalies in each testing subset.

B HYPER-PARAMETERS

We set the hyper-parameters of *OmniAnomaly* empirically in our experiments as follows. The length of input data sequence is set to 100 (*i.e.*, $T + 1 = 100$). The GRU layers and dense layers have 500 units. The ϵ in the standard deviation layer is set to 10^{-4}. The dimension of **z**-space variables is fixed to 3. We have conducted sensitivity analysis on the dimension in Appendix C. The length of planar NF is 20. We set the batch size as 50 for training, and run for 20 epochs with early stopping. We use Adam optimizer for stochastic gradient descent with an initial learning rate of 10^{-3} during model training. When back-propagating gradients through the network layers, gradient values may grow extremely large such that some model parameters overflow (*i.e.*, become NaN). To deal with such "gradient explosion", we use gradient clipping by norm with 10.0 as the limit. We apply L2 regularization with a coefficient of 10^{-4} to all layers of our model. During training, 30% of the training data is held for validation. For POT parameters, $q = 10^{-4}$ for all data sets, low quantile is 0.07 for SMAP, 0.01 for MSL, and 0.0001,0.0025 and 0.005 for three subsets of SMD. All experiments in this study are conducted on NVIDIA GeForce GTX 1080 Ti 11GB GDDR5X GPU. Using the above hyper-parameters, the training time of our model for SMAP, MSL and SMD are about 48, 11 and 87 minutes per epoch respectively.

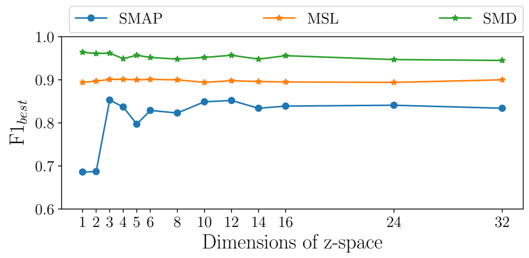


Figure 7: F1$_{best}$ of *OmniAnomaly* with different z-space dimensions.

C IMPACT OF z-SPACE DIMENSION

Dimension of **z**-space is important for *OmniAnomaly*. A large value would make dimension reduction have little effect so the reconstruction probability fails to find a good posterior [24], and too small of it may cause under-fitting.

Fig. 7 shows F1$_{best}$ of *OmniAnomaly* by varying different **z**-space dimensions. For our three datasets, their F1$_{best}$ do not change significantly when **z**-space dimensions are from 3∼32, thus we have a large room to choose the **z**-space dimension. We set **z**-space dimension to 3 empirically for all three datasets. Automatic selection of its value for other different datasets is difficult and not be studied in our paper.