Detecting Multivariate Time Series Anomalies with Zero Known Label

Qihang Zhou[1], **Jiming Chen**[1], **Haoyu Liu**[1,2*], **Shibo He**[1,3], **Wenchao Meng**[1]

[1]Zhejiang University, Hangzhou, China,
[2]NetEase Fuxi AI Lab, Hangzhou, China,
[3]Key Laboratory of Collaborative Sensing and Autonomous Unmanned Systems of Zhejiang Province, Hangzhou, China
{zqhang, cjm, haoyu_liu, s18he, wmengzju}@zju.edu.cn, liuhaoyu03@corp.netease.com

Abstract

Multivariate time series anomaly detection has been extensively studied under the one-class classification setting, where a training dataset with all normal instances is required. However, preparing such a dataset is very laborious since each single data instance should be fully guaranteed to be normal. It is, therefore, desired to explore multivariate time series anomaly detection methods based on the dataset without any label knowledge. In this paper, we propose MTGFlow, an unsupervised anomaly detection approach for Multivariate Time series anomaly detection via dynamic Graph and entity-aware normalizing Flow, leaning only on a widely accepted hypothesis that abnormal instances exhibit sparse densities than the normal. However, the complex interdependencies among entities and the diverse inherent characteristics of each entity pose significant challenges to density estimation, let alone to detect anomalies based on the estimated possibility distribution. To tackle these problems, we propose to learn the mutual and dynamic relations among entities via a graph structure learning model, which helps to model the accurate distribution of multivariate time series. Moreover, taking account of distinct characteristics of the individual entities, an entity-aware normalizing flow is developed to describe each entity into a parameterized normal distribution, thereby producing fine-grained density estimation. Incorporating these two strategies, MTGFlow achieves superior anomaly detection performance. Experiments on five public datasets with seven baselines are conducted, MTGFlow outperforms the SOTA methods by up to 5.0 AUROC%.

Introduction

Multivariate time series (MTS) broadly exist in many important scenarios, such as production data produced by multiple devices in smart factories and monitoring data generated by various sensors in smart grids. Anomalies in MTS exhibit unusual data behaviors at a specific time step or during a time period. To identify these anomalies, previous methods mostly focus on training one-class classification (OCC) models from only normal data (Wu and Keogh 2021; Schölkopf et al. 1999; Su et al. 2019; Chen et al. 2021; Deng and Hooi 2021; Xu et al. 2021; Zhang et al. 2019). They heavily rely on an assumption that the training dataset with all normal samples can be easily obtained (Ruff et al. 2021).

However, this assumption may not always hold in real-world scenarios (Goodge et al. 2021; Zhang, Zhao, and Li 2019; Zong et al. 2018; Qiu et al. 2022), leading to noisy training datasets with the mixture of normal and abnormal data instances. Meanwhile, it is already verified that model training procedure is prone to overfitting noisy labels (Zhang et al. 2021), so that the performance of those OCC-based methods could be severely degraded (Wang et al. 2019; Huyan et al. 2021). Therefore, it is rewarding to develop unsupervised MTS anomaly detection methods based on the dataset with absolute zero known labels.

An effective unsupervised strategy is modeling the dataset into a distribution, relying only on a widely accepted hypothesis that abnormal instances exhibit sparse densities than the normal, i.e., the low-density regions consist of abnormal samples and the high-density regions are formed by the normal samples (Gupta et al. 2013; Pang, Cao, and Aggarwal 2021; Wang et al. 2020). Methods have been explored along side this strategy and the key challenge lies in the accurate density estimation of the distribution. Time series density is modeled as the parameterized probability distribution (Salinas et al. 2020; Rasul et al. 2021; Feng et al. 2022), while it is still challenging to model a more complex data distribution. To improve the model capacity of density estimation, Rasul *et al.* (Rasul et al. 2020) further exploits normalizing flow to model complex distribution for high-dimensional MTS (Rasul et al. 2020). However, they neglect the interdependencies among constituent series which also play an important role in accurate density estimation.

The most related work is GANF (Dai and Chen 2021), which tackles the same MTS anomaly detection task. In their design, the static directed acyclic graph (DAG) is leveraged to model intractable dependence among multiple entities, and normalizing flow (Dinh, Sohl-Dickstein, and Bengio 2016; Papamakarios, Pavlakou, and Murray 2017) is employed to estimate an overall distribution for all entities together. Although GANF has achieved state-of-the-art (SOTA) results previously, it still suffers from two drawbacks. First, rather than a static inter-relationship, in real-world applications, the mutual dependencies among entities could not only be complex but also evolving. This dynamic property can not be simply characterized via a DAG struc-

[*]Corresponding author.

ture. Second, entities usually have diverse working mechanisms, leading to diverse sparse characteristics when anomalies occurred. GANF projects all entities into the same distribution, resulting in a compromise for the density estimation of each individual time series. Thereby, the final anomaly detection performance could also be degraded.

In this paper, we propose MTGFlow, an unsupervised anomaly detection method for MTS anomaly detection, to tackle the above problems. First, considering the evolving relations among entities, we introduce a graph structure learning module to model these changeable interdependencies. To learn the dynamic structure, a self-attention module (Vaswani et al. 2017) is plugged into our model for its superior performance on quantifying pairwise interaction. Second, aiming at the diverse inherent characteristics existed among individual entities, we design an entity-aware normalizing flow to model the entity-specific density estimation. Thereby, each entity can be assigned to a unique target distribution and the diverse entity densities can be estimated independently. In addition, we also propose to control the model size by sharing entity-specific model parameters, which helps MTGFlow achieve fine-grained density estimation without much memory consumption.

We summarize our contributions as follows:

- We propose MTGFlow, a new SOTA method for unsupervised MTS anomaly detection. It essentially enables anomaly localization and interpretation.
- We model the complicated interdependencies among entities into the dynamic graph, which captures the complex and evolving mutual dependencies among entities. Also, entity-aware normalizing flow is introduced to produce entity-specific density estimation.
- Experiments on five datasets with seven baseline methods are conducted, outperforming the SOTA methods by up to 5.0 AUROC%.

Related Work

Time Series Anomaly Detection

Time series anomaly detection has been extensively investigated under OCC setting (Chalapathy and Chawla 2019; Hundman et al. 2018). Previous influential methods like DeepSVDD (Ruff et al. 2018), EncDecAD (Malhotra et al. 2016), OmniAnomaly (Su et al. 2019), USAD (Audibert et al. 2020) and DAEMON (Chen et al. 2021) firstly train a model with absolutely normal instances so that the abnormal instances would exhibit differently when they are fed into the model during testing. Along side this line, Anomaly Transformer (Xu et al. 2021) and TranAD (Tuli, Casale, and Jennings 2022) further investigate fine-grained representation learning procedure via techniques like self-attention (Vaswani et al. 2017), achieving good detection performance.

However, all these works are based on the assumption that a sufficient training dataset with all normal instances can be acquired, which is very hard for real-world applications since each instance should be manually checked carefully. In addition, once there exist abnormal instances in the training data, the performance of these OCC-based detection methods could be severely degraded (Wang et al. 2019; Huyan et al. 2021). Therefore, instead of fitting distribution of the normal training dataset, Dai and Chen propose GANF (Dai and Chen 2021) to detect MTS anomalies in an unsupervised manner. Inspired by them, we propose MTGFlow to facilitate the learning capacity and improve detection performance.

Graph Structure Learning

Graph convolution networks (GCN) (Kipf and Welling 2016) have achieved great success in modeling intrinsic structure patterns. However, such a graph structure is usually unknown in real-world scenarios so graph structure learning methods are in need (Veličković et al. 2017). In the field of anomaly detection of MTS, some recent works attempt to explore this area. GDN (Deng and Hooi 2021) learns a directed graph via node embedding vectors. According to the cosine similarity of embedding vectors, top-K candidates of each node are considered to have interdependencies. GANF (Dai and Chen 2021) models relations among multiple sensors, using DAG, and learns the structure of the DAG through continuous optimization with a simplified constraint that facilitates backward propagation. Our work MTGFlow models the mutual complex dependence as a fully connected graph via the self-attention mechanism, so that a much more flexible relation among entities can be represented.

Normalizing Flow for Anomaly Detection

Normalizing flow is an important technology on density estimation and has been successfully utilized in image generation tasks (Dinh, Sohl-Dickstein, and Bengio 2016; Papamakarios, Pavlakou, and Murray 2017). Recently, it is also explored for anomaly detection based on the assumption that anomalies are in low-density regions, like DifferNet (Rudolph, Wandt, and Rosenhahn 2021) and CFLOW-AD (Gudovskiy, Ishizaka, and Kozuka 2022), which leverage normalizing flow to estimate likelihoods of normal embeddings and declare image defects when the embedding is far away from the dense region. GANF is a great work that employs normalizing flow for unsupervised MTS anomaly detection. We follow this research line and facilitate model capacity through an entity-aware normalizing flow design.

Preliminary

In this section, we give a brief introduction of normalizing flow to better understand MTGFlow.

Normalizing Flow

Normalizing flow is an unsupervised density estimation approach to map the original distribution to an arbitrary target distribution by the stack of invertible affine transformations. When density estimation on original data distribution $\mathcal{X}$ is intractable, an alternative option is to estimate z density on target distribution $\mathcal{Z}$. Specifically, suppose a source sample $x \in \mathcal{R}^D \sim \mathcal{X}$ and a target distribution sample $z \in \mathcal{R}^D \sim \mathcal{Z}$. Bijective invertible transformation $\mathcal{F}_\theta$ aims to achieve one-to-one mapping $z = f_\theta(x)$ from $\mathcal{X}$

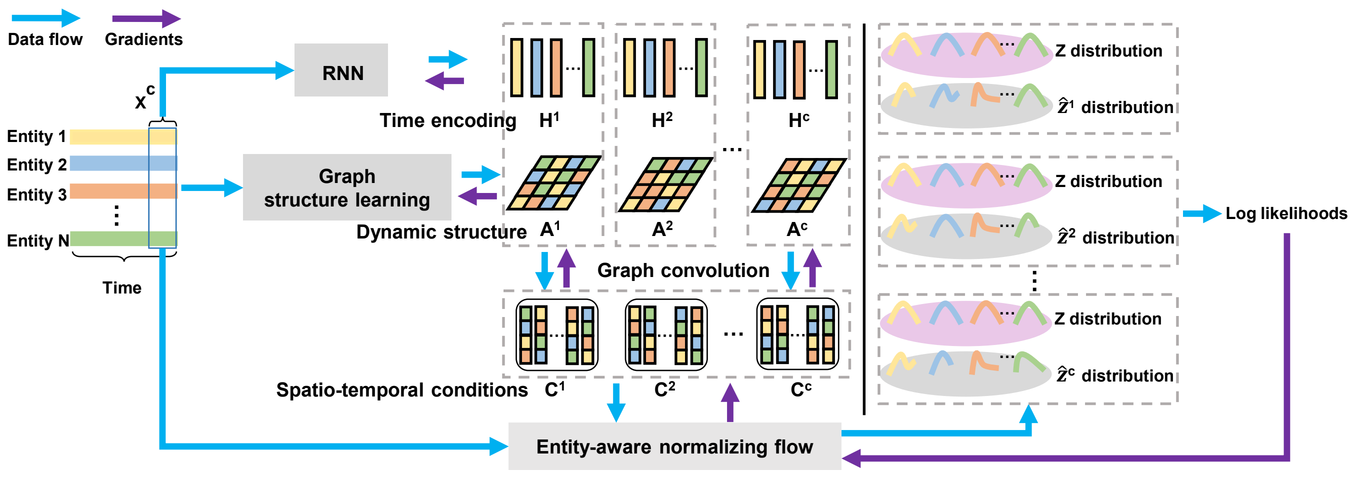

Figure 1: Overview of the proposed MTGFlow. Within a sliding window of size T, time series x^c is fed to the RNN module to capture the temporal correlations. Hidden states of RNN are regarded as time encoding, H^c. Meanwhile, x^c is also input to the graph structure learning module to capture dynamic interdependencies among entities, which are modeled as adjacency matrix A^c. The spatio-temporal conditions C^c are derived via the graph convolution operation for H^c and A^c. Finally, C^c is used to help entity-aware normalizing flow model to produce entity-specific density estimation for the distribution of time series.

to $\mathcal{Z}$. According to the change of variable formula, we can get $P_{\mathcal{X}}(x) = P_{\mathcal{Z}}(z)\left|\det \frac{\partial f_\theta}{\partial x^T}\right|$. Benefiting from the invertibility of mapping functions and tractable jacobian determinants $\left|\det \frac{\partial f_\theta}{\partial x^T}\right|$. The objective of flow models is to achieve $\hat{z} = z$, where $\hat{z} = f_\theta(x)$. Flow models are able to achieve more superior density estimation performance when additional conditions C are input (Ardizzone et al. 2019). Such a flow model is called conditional normalizing flow, and its corresponding mapping is derived as $z = f_\theta(x|C)$. Parameters θ of f_θ are updated by maximum likelihood estimation (MLE): $\theta^* = \arg\max_\theta (log(P_{\mathcal{Z}}(f_\theta(x|C)) + log(\left|\det \frac{\partial f_\theta}{\partial x^T}\right|))$

Method

Data Preparation

MTS are defined as $x = (x_1, x_2, ..., x_K)$ and $x_i \in \mathcal{R}^L$, where K represents the total number of entities, and L denotes the total number of observations of each entity. We use the z-score to normalize the time series from different entities. $\bar{x}_i = \frac{x_i - mean(x_i)}{std(x_i)}$, where $mean(x_i)$ and $std(x_i)$ represent the mean and standard deviation of the i-th entity along the time dimension, respectively. To preserve temporal correlations of the original series, we use a sliding window with size T and stride size S to sample the normalized MTS. T and S can be adjusted to obtain the training sample x^c, where c is the sampling count. x^c is short for $x^{cS:cS+T}$.

Overall Structure

The core idea behind MTGFlow is to dynamically model mutual dependence so that fine-grained density estimation of the multivariate time series can be obtained. Such accurate estimation enables the superiority of capturing low-density regions, and further promotes the anomaly detection performance even if there is high anomaly contamination in the training dataset. Fig. 1 shows the overview of MTGFlow. In particular, we model the temporal variations of each entity, using RNN model. Meanwhile, a graph structure learning module is leveraged to model the dynamic interdependencies. Then, the derived time encoding, output of RNN, performs the graph convolution operation with the above corresponding learned graph structure. We regard above outputs as spatio-temporal *conditions* as they contain temporal and structural information. Next, the spatio-temporal conditions are input to help entity-aware normalizing flow achieve precise fine-grained density estimation. The deviations of $\hat{z}$ and z are measured by log likelihoods. Finally, all modules of MTGFlow are jointly optimized through MLE.

Graph Structure Learning via Self-attention

Since dependence among entities is mutual and evolves over time, we exploit self-attention to learn a dynamic graph structure. Entities in multivariate time series are regarded as graph nodes. Given the window sequence x^c, the query and key of node i are represented by vectors $x_i^c W^Q$ and $x_i^c W^K$, where $W^Q \in \mathcal{R}^{T\times T}$ and $W^K \in \mathcal{R}^{T\times T}$ are the query and key weights. The pairwise relationship e_{ij}^c at the c-th sampling count between node i and node j is described as $e_{ij}^c = \frac{(x_i^c W^Q)(x_j^c W^K)^T}{\sqrt{T}}$. The attention score a_{ij}^c is used to quantify the pairwise relation from node i to node j, calculated by $a_{ij}^c = \frac{\exp(e_{ij}^c)}{\sum_{j=1}^K exp(e_{ij}^c)}$. And the attention matrix consists of attention scores of each node, thus including mutual dependence among entities. Naturally, we treat the attention matrix as the adjacency matrix A^c of the learned graph. Since input time series are evolving over time, A^c also changes to capture the dynamic interdependencies.

Spatio-temporal Condition

To better estimate the density of multiple time series, the robust spatio-temporal condition information is important. As described above, underlying structure information is modeled as the dynamic graph. Besides spatio information, temporal correlations also play an important role to feature time series. Here, we follow the most prevalent idea, where RNN is utilized to capture the time correlations. For a window sequence of entity k, x^c, the time representation H_k^t at time $t \in [cS : cS + T)$ is derived by $H_k^t = \mathbf{RNN}(x_k^t, H_k^{t-1})$, where RNN can be any sequence model such as LSTM (Hochreiter and Schmidhuber 1997) and GRU (Cho et al. 2014), and H_k^t is the hidden state of RNN. To derive the spatio and temporal information C^t of all entities at t, a graph convolution operation is performed through the learned graph A^c. As mentioned in GANF, we also find that history information of the node itself helps enhance temporal relationships of time series. Hence, the spatio-temporal condition at t:

$$C^t = ReLU(A^c H^t W_1 + H^{t-1} W_2) W_3, \tag{1}$$

where W_1 and W_2 are graph convolution and history information weights, respectively. W_3 is used to improve the expression ability of condition representations. The spatio-temporal condition C^c for window c is the concatenation of C^t along the time axis.

Entity-aware Normalizing Flow

Distributions of individual entities have discrepancies because of their different work mechanisms, and thus their respective anomalies will generate distinct sparse characteristics. If we map time series from all entities to the same distribution $N(0, I)$, as does in GANF, then the description capacity of the model will be largely limited and the unique inherent property of each entity will be ignored. Therefore, we design the entity-aware normalizing flow $z_k = f_\theta^k(x|C)$ to make more detailed density estimation, where x, C, k are the input sequence, condition, and the k-th entity, respectively. Technically, for one entity, we assign the multivariate Gaussian distribution as the target distribution. The covariance matrix of the above target distribution is the identity matrix I for better convergence. Moreover, in order to generate different target distributions $\mathcal{Z}_k$, we independently draw mean vectors $\mu_k \in R^T$ from $N(0, I)$ (Izmailov et al. 2020), However, we find that such setting results in performance degradation. So, in our experiment, each element of μ_k is kept the same. Specifically, for the time series of the entity k, the density estimation is given by:

$$\begin{gathered} P_{\mathcal{X}_k}(x_k) = P_{\mathcal{Z}_k}(f_\theta^k(x_k|C)) \left|\det \frac{\partial f_\theta^k}{\partial {x_k}^T}\right| \\ \mathcal{Z}_k = N(\mu_k, I) \end{gathered} \tag{2}$$

where each element of μ_k is the same, and is drawn from the $N(0, 1)$. In such a case, model parameters will increase with the number of entities. To mitigate this problem, we share entity-aware normalizing flow parameters across all entities. So, the density estimation for k reads:

$$P_{\mathcal{X}_k}(x_k) = P_{\mathcal{Z}_k}(f_\theta(x_k|C)) \left|\det \frac{\partial f_\theta}{\partial {x_k}^T}\right| \tag{3}$$

Joint Optimization

As described above, MTGFlow combines graph structure learning and RNN to capture the spatio and temporal dependence on multiple time series. Then, derived saptio-temporal conditions are utilized to contribute to entity-aware normalizing flow accurately estimating density of time series. To avoid getting stuck in the local optimum for each module of MTGFlow, we jointly optimize all modules for overall performance of MTGFlow. The whole parameters W^* are estimated via MLE.

$$\begin{aligned} W^* &= \arg\max_W log(P_{\mathcal{X}}(x)) \\ &\approx \arg\max_W \frac{1}{NK} \sum_{c=1}^{N} \sum_{k=1}^{K} log(P_{\mathcal{Z}_k}(f_\theta(x_k^c|C_k^c)) \left|\det \frac{\partial f_\theta}{\partial {x_k^c}^T}\right|) \\ &\approx \arg\max_W \frac{1}{NK} \sum_{c=1}^{N} \sum_{k=1}^{K} -\frac{1}{2}\|\hat{z_k^c} - \mu_k\|_2^2 + log \left|\det \frac{\partial f_\theta}{\partial {x_k^c}^T}\right|, \end{aligned}$$

where N is the total number of windows.

Anomaly Detection and Interpretation

Based on the hypothesis that anomalies tend to be sparse on data distributions, low log likelihoods indicate that the observations are more likely to be anomalous.

Anomaly Detection Taking the window sequence x_k^c as the input, the density of all entities can be estimated. The mean of the negative log likelihoods of all entities serves as the anomaly score S_c, which is calculated by:

$$S_c = -\frac{1}{K} \sum_{k=1}^{K} log(P_{\mathcal{X}_k}(x_k^c)) \tag{4}$$

A higher anomaly score represents that x_k^c locates in the lower density region, indicating a higher possibility to be abnormal. Since abnormal series exist in the training set and validation set, we cannot directly set the threshold to label the anomaly, such as the maximum deviation in validation data (Deng and Hooi 2021). Therefore, to reduce the anomaly disturbance, we store S_c of the whole training set, and the interquartile range (IQR) is used to set the threshold: $Threshold = Q_3 + 1.5 * (Q_3 - Q_1)$, where Q_1 and Q_3 are 25-th and 75-th percentile of S_c.

Anomaly Interpretation Abnormal behaviors of any entity could lead to the overall abnormal behavior of the whole window sequence. Naturally, we can get the entity anomaly score S_{ck} for entity k according to Eq. (4).

$$S_c = -\frac{1}{k} \sum_{k=1}^{K} log(P_{\mathcal{X}_k}(x_k^c)) = \sum_{k=1}^{K} S_{ck} \tag{5}$$

Since we map time series of each entity into unique target distributions, different ranges of S_{ck} are observed. This bias will assign each entity to different weights in terms of its contribution to S_c. To circumvent the above-unexpected bias, we design the entity-specific threshold for each entity. Considering different scales of S_{ck}, IQR is used to set respective thresholds. Therefore, the threshold for S_{ck} is given as: $Threshold_k = \lambda_k(Q_3^k + 1.5 * (Q_3^k - Q_1^k))$, where Q_1^k and Q_3^k are 25-th and 75-th percentile of S_{ck} across all observations, respectively. And λ_k is used to adjust $Threshold_k$ because normal observations in different entities also fluctuate with different scales.

Experiment

Experiment Setup

Dataset The commonly used public datasets for MTS anomaly detection in OCC are MSL (Mars Science Laboratory rover) (Hundman et al. 2018), SMD (Server Machine Dataset) (Su et al. 2019), PSM (Pooled Server Metrics) (Abdulaal, Liu, and Lancewicki 2021), SWaT (Secure Water Treatment) (Goh et al. 2016) and WADI (Water Distribution) (Ahmed, Palleti, and Mathur 2017). The sensor data in SWaT and WADI is from water Treatment with 51 and 123 entities. MSL, SMD, and PSM are from Mars rover with 55 features, server metrics with 38 features, and server nodes at eBay with 25 features, respectively. Since only normal time series are provided in these datasets for training in OCC setting, we follow the dataset setting of GANF (Dai and Chen 2021) and split the original testing dataset by 60% for training, 20% for validation, and 20% for testing in SWaT. For other datasets, the training split contains 60% data, and the test split contains 40% data.

Implementation Details For all datasets, we set the window size as 60 and the stride size as 10. Adam optimizer with a learning rate 0.002 is utilized to update all parameters. One layer of LSTM is sufficient to extract time representations in our experiment. One self-attention layer with 0.2 dropout ratio is adopted to learn the graph structure. We use MAF as the normalizing flow model. For SWaT, one flow block and 512 batch size are employed. For other datasets, we arrange two flow blocks for it and set the batch size as 256. λ is set as 0.8 for thresholds of all entities. The epoch is 40 for all experiments, which are performed in PyTorch-1.7.1 with a single NVIDIA RTX 3090 24GB GPU[1].

Evaluation Metric As in previous works, MTGFlow aims to detect window-level anomalies, and labels are annotated as abnormal when there exists any anomalous time point. The performance is evaluated by the Area Under the Receiver Operating Characteristic curve (AUROC).

Baselines SOTA methods include semi-supervised methods DeepSAD (Ruff et al. 2019), OCC methods DeepSVDD (Ruff et al. 2018), ALOCC (Sabokrou et al. 2020), DROCC (Goyal et al. 2020), and USAD (Audibert et al. 2020) and unsupervised methods DAGMM (Zong et al. 2018) and GANF (Dai and Chen 2021).

Performance We list the AUROC metric results in Table 1. Note that the standard deviation of SMD is large because it comprises 28 sub-datasets, where we test the performance on each of them and average all the results. MTGFlow has superior performance over all the other seven baselines. Compared with MTGFlow, DeepSVDD and DROCC project all training samples into the hypersphere so that they cannot learn the accurate decision boundary distinguishing normal from abnormal samples. Adversarial learning used by ALOCC and USAD and semi-supervised learning strategy in DeepSAD leverage a more informative training procedure to mitigate the effect of high anomaly contamination. As for DAGMM, it is restricted to the distribution estimation ability of GMM for multiple entities. Although GANF obtains a better result, its detection performance is still limited by inadequate dependence modeling and indiscriminative density estimation. Due to a much more flexible modeling structure, MTGFlow outperforms the above baseline methods. Moreover, we study log likelihoods for anomalies ranging from 2016/1/2 11:07:00 to 11:37:00 in Fig. 2. It is clear that log likelihoods are high for the normal series but lower for labeled abnormal ones (highlighted in red). This variation of log likelihoods validates that MTGFlow can detect anomalies according to low density regions of modeled distribution. Meanwhile, to investigate anomaly discrimination ability of MTGFlow, we present the normalized S_c for MTGFlow and GANF in Fig. 3. As it is displayed, for normal series, anomaly scores of MTGFlow are more centered at 0 than these of GANF, and the overlap areas of normal and abnormal scores are also smaller in MTGFlow, reducing the false positive ratio. This larger score discrepancy corroborates that MTGFlow has superior detection performance.

[1]Code is available at github.com/zqhang/MTGFLOW.

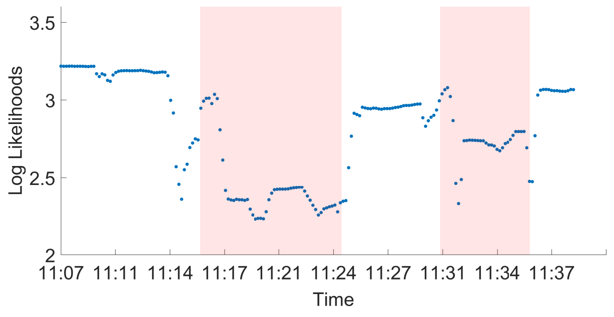

Figure 2: Log likelihoods for anomalies.

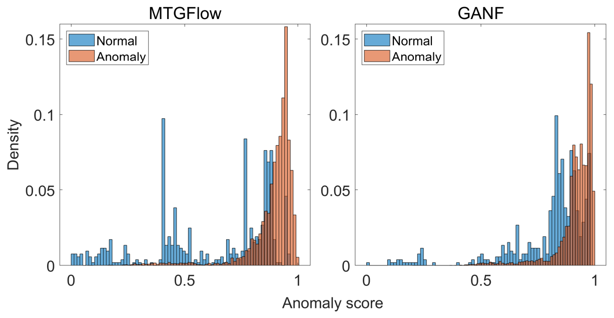

Figure 3: Comparison on normalized anomaly scores between MTGFlow and GANF.

Ablation Study

Module Ablation Study To test the validity of each designed module, we give several ablation experiments. We denote MTGFlow without graph and entity-aware normalizing flow as MTGFlow/(G, E), MTGFlow only without graph as MTGFlow/G, and MTGFlow only without entity-aware normalizing flow as MTGFlow/E. Results are presented in

Dataset	DeepSVDD	ALOCC	DROCC	DeepSAD	USAD	DAGMM	GANF	MTGFlow
SWaT	66.8±2.0	77.1±2.3	72.6±3.8	75.4±2.4	78.8±1.0	72.8 ±3.0	79.8±0.7	**84.8±1.5**
WADI	83.5±1.6	83.3±1.8	75.6±1.6	85.4±2.7	86.1±0.9	77.2±0.9	90.3±1.0	**91.9±1.1**
PSM	67.5±1.4	71.8±1.3	74.3±2.0	73.2±3.3	78.0±0.2	64.6 ±2.6	81.8±1.5	**85.7±1.5**
MSL	60.8±0.4	60.3±0.9	53.4±1.6	61.6±0.6	57.0±0.1	56.5 ±2.6	64.5±1.9	**67.2±1.7**
SMD	75.5±15.5	80.5±11.1	76.7±8.7	85.9 ±11.1	86.9±11.7	78.0±9.2	89.2±7.8	**91.3±7.6**

Table 1: Anomaly detection performance of AUROC(%) on five public datatsets.

	Graph	Entity	SWaT	WADI
MTGFlow/(G, E)	✗	✗	78.3±0.9	89.7±0.5
MTGFlow/G	✗	✓	82.4±1.0	91.3±0.4
MTGFlow/E	✓	✗	81.2±1.1	91.0±0.7
MTGFlow	✓	✓	**84.8±1.5**	**91.9±1.1**

Table 2: Module ablation study (AUROC%).

Window size \ Blocks		1	2	3
SWaT	40	81.4±3.2	82.7±2.1	81.7±0.9
	60	**84.8±1.5**	83.6±2.0	83.1±0.9
	80	82.8±1.0	82.7±0.8	83.4±0.6
	100	82.6±0.5	83.4±0.9	83.5±0.6
	120	83.2±2.0	83.4±2.3	84.5±2.6
WADI	40	90.8±1.3	91.7±1.2	91.7±1.3
	60	89.2±1.9	**91.9±1.1**	91.5±0.8
	80	89.8±2.0	90.7±0.8	91.7±0.7
	100	89.6±1.1	90.9±0.8	91.8±0.6
	120	88.6±1.4	91.0±0.6	91.5±0.9

Table 3: Ablation study of hyperparameters (AUROC%).

Table 2, where MTGFlow/(G, E) obtains the worst performance. It is attributed to the lack of relational modeling among entities and indistinguishable density estimation. Applying graph structure learning to model pairwise relations, MTGFlow/E achieves better performance. Also, considering more fine-grained density estimation, MTGFlow/G achieves an improvement over MTGFlow/(G, E). Integrating these two modules, MTGFlow accomplishes the best results.

Hyperparameter Robustness We conduct a comprehensive study on the choice of hyperparameters, the results are shown in Table 3. Concretely, we conduct experiments with various sizes for the sliding window and the number of the normalizing flow blocks in Table 3. When the window size is small, such as 40, 60, and 80, the increase in the number of blocks does not necessarily improve anomaly detection performance. A larger model may cause overfitting to the whole distribution, where abnormal sequences are undesirably located in high-density regions of this distribution. When the window size is larger, the distribution to be estimated is high-dimensional so that model needs more capacity. Hence, detection performance derives the average gain with blocks increasing due to more accurate distribution modeling.

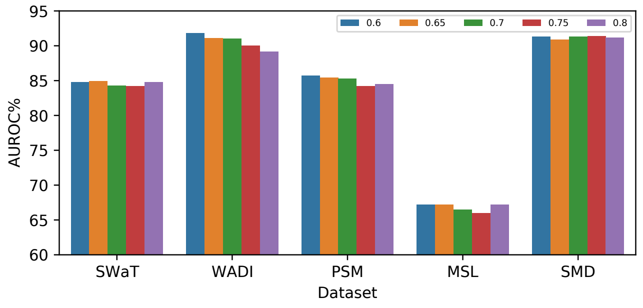

Figure 4: Effect of anomaly contamination ratio.

Anomaly Ratio Analysis To further investigate the influence of anomaly contamination rates, we vary training splits to adjust anomalous contamination rates. For all the above-mentioned datasets, the training split increases from 60% to 80% with 5% stride. We present an average result over five runs in Fig. 4. Although the anomaly contamination ratio of training dataset rises, the anomaly detection performance of MTGFlow remains at a stable high level.

Result Analysis

In order to further investigate the effectiveness of MTGFlow, we give a detailed analysis based on SWaT dataset.

Dynamic Graph Structure Interdependencies among entities are not guaranteed to be immutable. In fact, pairwise relations evolve with time. Benefiting from self-attention, MTGFlow can model this characteristic into a dynamic graph structure. We treat the attention matrix as the graph adjacent matrix. An empirical threshold of 0.15 is set for the adjacency matrix to show an intuitive learned graph structure in the test split. In Fig. 5, the node size represents its node degrees, the arrow direction represents the learned directed dependence and the arrow width indicates the weight of the corresponding interdependencies. The graph structure at 2016/1/1 14:00:00 is centered on the sensor $P201$, while the edges in the graph have completely changed and the center has shifted from $P201$ to $AIT503$ at 2016/1/2 7:00:00. This alteration of the graph structure may result from changing in working condition of water treatment plant. Besides, two similar graph structures can be found at 2016/1/2 13:00:00 and 2016/1/2 14:00:00. This suggests that the graph structure will be consistent if the interdependencies remain unchanged over a period of time, possibly due to repetitive work patterns of entities. In addition, the

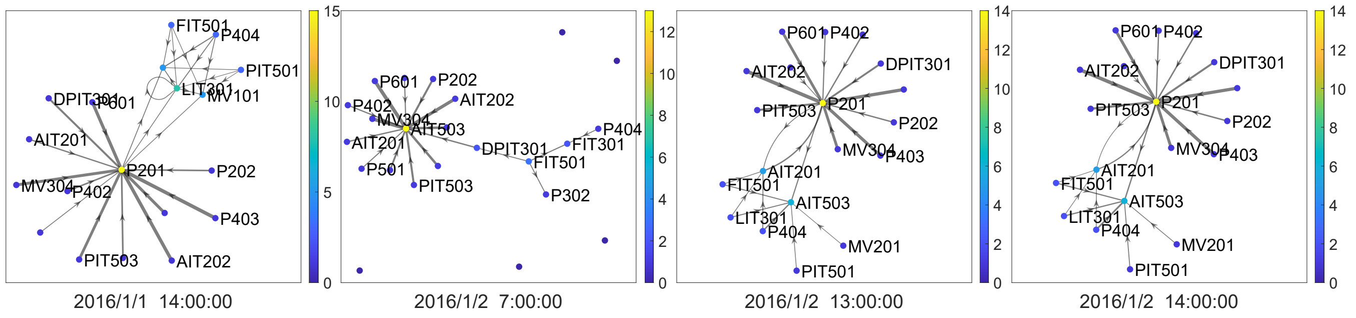

Figure 5: Dynamic graph structure in MTGFlow.

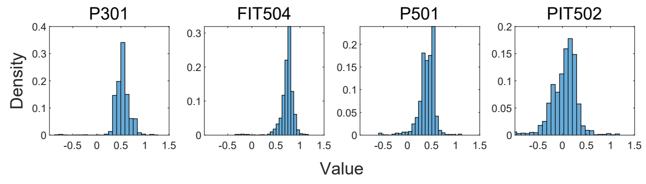

Figure 6: Transformed distributions of multiple entities.

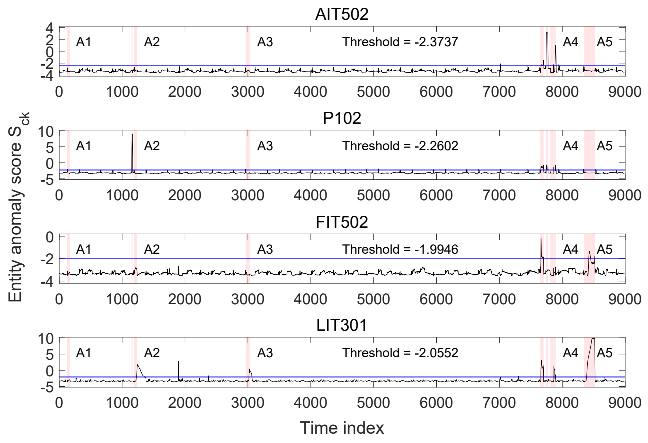

Figure 7: Variation of log likelihoods for different entities on the whole testing dataset (anomalies are highlighted in red, and the blue line is the threshold according to S_{ck}).

main pairwise relations (thick arrow) at 2016/1/1 14:00:00 are similar as the ones at 2016/1/2 14:00:00, both centered on $P201$. It indicates that the interdependencies on multiple sensors are periodic. We also find mutual interdependencies from learned graph structures, such as the edges between $P201$ and $AIT201$ at 2016/1/2 13:00:00. We summarize the findings: (1) Dynamic interdependencies among multiple entities. (2) Consistent interdependencies among multiple entities. (3) Periodic interdependencies among multiple entities. (4) Mutual interdependencies among multiple entities. Therefore, it is necessary to use a dynamic graph to model such changeable interdependencies.

Entity-specific Density Estimation We further explore whether the distributions of all entities are transformed into different target distributions to verify our entity-aware design. Since the window size is 60, the corresponding transformed distributions are also 60-dimensional distributions. Every single dimension of the multivariate Gaussian distribution is a Gaussian distribution. For better visualization, we present the 0-th dimension of the transformed distributions in Fig. 6. Four distributions of different entities are displayed. It can be seen that these distributions have been projected as unique distributions. Moreover, these distributions are successfully converted to preset Gaussian distributions with different mean vectors. The one-to-one mapping models entity-specific distributions and captures their respective sparse characteristics of anomalies.

Distinct Sparse Characteristics To demonstrate that the sparse characteristics vary with different entities, we study changes of S_{ck} along time on SWaT. As shown in Fig. 7, S_{ck} of $AIT502$, $P102$, $FIT502$, and $LIT301$ are presented. The highlighted regions denote marked anomalies. For a better illustration, we divide the anomalous regions as A_1, A_2, A_3, A_4, and A_5 along the timeline. We can observe that different entities react to different anomalies because of their different work mechanisms. Specifically, $AIT502$ has obvious fluctuations at A_4, while $P102$ reacts to A_2. In addition, $FIT502$ is sensitive to A_4 and A_5, yet $LIT301$ is sensitive to A_2, A_3, A_4, and A_5. Nevertheless, MTGFlow is still able to accurately distinguish and detect these anomalies.

Conclusion

In this work, we proposed MTGFlow, an unsupervised anomaly detection approach for MTS based on the dataset with absolute zero known label. Extensive experiments on real-world datasets demonstrate its superiority, even if there exists high anomaly contamination. The superior anomaly detection performance of MTGFlow is attributed to dynamic graph structure learning and entity-aware density estimation. In addition, we explore various interdependencies that exist between individual entities from the learned dynamic graph structure. And a detected anomaly can be understood and localized via entity anomaly scores. In the future, we plan to explore the performance of our method from more realistic scenarios (Wu and Keogh 2021), and further improve its practicality.

Acknowledgements

This work was supported by the National Natural Science Foundation Program of China under Grant U1909207 and Grant U21B2029.

References

Abdulaal, A.; Liu, Z.; and Lancewicki, T. 2021. Practical approach to asynchronous multivariate time series anomaly detection and localization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2485–2494.

Ahmed, C. M.; Palleti, V. R.; and Mathur, A. P. 2017. WADI: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd international workshop on cyber-physical systems for smart water networks*, 25–28.

Ardizzone, L.; Lüth, C.; Kruse, J.; Rother, C.; and Köthe, U. 2019. Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*.

Audibert, J.; Michiardi, P.; Guyard, F.; Marti, S.; and Zuluaga, M. A. 2020. USAD: unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 3395–3404.

Chalapathy, R.; and Chawla, S. 2019. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*.

Chen, X.; Deng, L.; Huang, F.; Zhang, C.; Zhang, Z.; Zhao, Y.; and Zheng, K. 2021. DAEMON: Unsupervised Anomaly Detection and Interpretation for Multivariate Time Series. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, 2225–2230. IEEE.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Dai, E.; and Chen, J. 2021. Graph-Augmented Normalizing Flows for Anomaly Detection of Multiple Time Series. In *International Conference on Learning Representations*.

Deng, A.; and Hooi, B. 2021. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 4027–4035.

Dinh, L.; Sohl-Dickstein, J.; and Bengio, S. 2016. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*.

Feng, S.; Xu, K.; Wu, J.; Wu, P.; Lin, F.; and Zhao, P. 2022. Multi-scale Attention Flow for Probabilistic Time Series Forecasting. *arXiv preprint arXiv:2205.07493*.

Goh, J.; Adepu, S.; Junejo, K. N.; and Mathur, A. 2016. A dataset to support research in the design of secure water treatment systems. In *International conference on critical information infrastructures security*, 88–99. Springer.

Goodge, A.; Hooi, B.; Ng, S. K.; and Ng, W. S. 2021. LUNAR: Unifying Local Outlier Detection Methods via Graph Neural Networks. *arXiv preprint arXiv:2112.05355*.

Goyal, S.; Raghunathan, A.; Jain, M.; Simhadri, H. V.; and Jain, P. 2020. DROCC: Deep robust one-class classification. In *International Conference on Machine Learning*, 3711–3721. PMLR.

Gudovskiy, D.; Ishizaka, S.; and Kozuka, K. 2022. Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 98–107.

Gupta, M.; Gao, J.; Aggarwal, C. C.; and Han, J. 2013. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and data Engineering*, 26(9): 2250–2267.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Hundman, K.; Constantinou, V.; Laporte, C.; Colwell, I.; and Soderstrom, T. 2018. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 387–395.

Huyan, N.; Quan, D.; Zhang, X.; Liang, X.; Chanussot, J.; and Jiao, L. 2021. Unsupervised outlier detection using memory and contrastive learning. *arXiv preprint arXiv:2107.12642*.

Izmailov, P.; Kirichenko, P.; Finzi, M.; and Wilson, A. G. 2020. Semi-supervised learning with normalizing flows. In *International Conference on Machine Learning*, 4615–4630. PMLR.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

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

Pang, G.; Cao, L.; and Aggarwal, C. 2021. Deep Learning for Anomaly Detection: Challenges, Methods, and Opportunities. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, 1127–1130. New York, NY, USA: Association for Computing Machinery. ISBN 9781450382977.

Papamakarios, G.; Pavlakou, T.; and Murray, I. 2017. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30.

Qiu, C.; Li, A.; Kloft, M.; Rudolph, M.; and Mandt, S. 2022. Latent Outlier Exposure for Anomaly Detection with Contaminated Data. *arXiv preprint arXiv:2202.08088*.

Rasul, K.; Seward, C.; Schuster, I.; and Vollgraf, R. 2021. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, 8857–8868. PMLR.

Rasul, K.; Sheikh, A.-S.; Schuster, I.; Bergmann, U.; and Vollgraf, R. 2020. Multivariate probabilistic time series forecasting via conditioned normalizing flows. *arXiv preprint arXiv:2002.06103*.

Rudolph, M.; Wandt, B.; and Rosenhahn, B. 2021. Same same but differnet: Semi-supervised defect detection with normalizing flows. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 1907–1916.

Ruff, L.; Kauffmann, J. R.; Vandermeulen, R. A.; Montavon, G.; Samek, W.; Kloft, M.; Dietterich, T. G.; and Müller, K.-R. 2021. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5): 756–795.

Ruff, L.; Vandermeulen, R.; Goernitz, N.; Deecke, L.; Siddiqui, S. A.; Binder, A.; Müller, E.; and Kloft, M. 2018. Deep one-class classification. In *International conference on machine learning*, 4393–4402. PMLR.

Ruff, L.; Vandermeulen, R. A.; Görnitz, N.; Binder, A.; Müller, E.; Müller, K.-R.; and Kloft, M. 2019. Deep semi-supervised anomaly detection. *arXiv preprint arXiv:1906.02694*.

Sabokrou, M.; Fathy, M.; Zhao, G.; and Adeli, E. 2020. Deep end-to-end one-class classifier. *IEEE transactions on neural networks and learning systems*, 32(2): 675–684.

Salinas, D.; Flunkert, V.; Gasthaus, J.; and Januschowski, T. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3): 1181–1191.

Schölkopf, B.; Williamson, R. C.; Smola, A. J.; Shawe-Taylor, J.; Platt, J. C.; et al. 1999. Support vector method for novelty detection. In *NIPS*, volume 12, 582–588. Citeseer.

Su, Y.; Zhao, Y.; Niu, C.; Liu, R.; Sun, W.; and Pei, D. 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2828–2837.

Tuli, S.; Casale, G.; and Jennings, N. R. 2022. TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data. *arXiv preprint arXiv:2201.07284*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

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

Wang, R.; Nie, K.; Chang, Y.-J.; Gong, X.; Wang, T.; Yang, Y.; and Long, B. 2020. Deep Learning for Anomaly Detection. KDD '20, 3569–3570. New York, NY, USA: Association for Computing Machinery. ISBN 9781450379984.

Wang, S.; Zeng, Y.; Liu, X.; Zhu, E.; Yin, J.; Xu, C.; and Kloft, M. 2019. Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network. *Advances in neural information processing systems*, 32.

Wu, R.; and Keogh, E. 2021. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering*.

Xu, J.; Wu, H.; Wang, J.; and Long, M. 2021. Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy. *arXiv preprint arXiv:2110.02642*.

Zhang, C.; Bengio, S.; Hardt, M.; Recht, B.; and Vinyals, O. 2021. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3): 107–115.

Zhang, C.; Song, D.; Chen, Y.; Feng, X.; Lumezanu, C.; Cheng, W.; Ni, J.; Zong, B.; Chen, H.; and Chawla, N. V. 2019. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 1409–1416.

Zhang, L.; Zhao, J.; and Li, W. 2019. Online and unsupervised anomaly detection for streaming data using an array of sliding windows and PDDs. *IEEE Transactions on Cybernetics*, 51(4): 2284–2289.

Zong, B.; Song, Q.; Min, M. R.; Cheng, W.; Lumezanu, C.; Cho, D.; and Chen, H. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*.