POSSIBILITY FOR PROACTIVE ANOMALY DETECTION

Jinsung Jeon[1]***, Jaehyeon Park**[2]**, Sewon Park**[3]**, Jeongwhan Choi**[4]**,
Minjung Kim**[3]**, Noseong Park**[2]
[1]University of California, San Diego, [2]KAIST, [3]Samsung SDS, [4]Yonsei University
jjsjjs0902@gmail.com

ABSTRACT

Time-series anomaly detection, which detects errors and failures in a workflow, is one of the most important topics in real-world applications. The purpose of time-series anomaly detection is to reduce potential damages or losses. However, existing anomaly detection models detect anomalies through the error between the model output and the ground truth (observed) value, which makes them impractical. In this work, we present a *proactive* approach for time-series anomaly detection based on a time-series forecasting model specialized for anomaly detection and a data-driven anomaly detection model. Our proactive approach establishes an anomaly threshold from training data with a data-driven anomaly detection model, and anomalies are subsequently detected by identifying predicted values that exceed the anomaly threshold. In addition, we extensively evaluated the model using four anomaly detection benchmarks and analyzed both predictable and unpredictable anomalies. We attached the source code as supplementary material.

1 INTRODUCTION

Time-series anomaly detection, which detects anomalous points caused by failures and errors, is one of the most important topics in the time-series task (Munir et al., 2018; He & Zhao, 2019; Zhao et al., 2020; Deng & Hooi, 2021; Chen et al., 2021). In practice, various real-world applications reduce potential damage by determining the anomaly of events that have occurred. While previous anomaly detection models have shown good results in time-series anomaly detection, they are still limited in preventing anomalies in advance. However, since some anomalies appear with small precursors before they occur, these precursors can help predict the anomaly.

In practice, many applications rely on reactive approaches to address anomalies after they have happened, which inherently limits their ability to prevent damages proactively. In contrast, a proactive approach aims to detect anomalies before they occur, enabling early intervention and damage mitigation. By identifying potential issues ahead of time, proactive anomaly detection can significantly reduce risks and damages compared to reactive methods, which focus on minimizing harm after anomalies have already happened. This proactive strategy offers a critical advantage in preventing or mitigating the impact of anomalies.

In this paper, we propose a *proactive* approach for time-series anomaly detection, which integrates a time-series forecasting model specialized for anomaly detection (cf. Figure 1). Our approach performs forecasting-based anomaly detection without relying on the ground truth of the testing data. It involves fitting a data-driven model using only training data, setting the extreme values of the distribution or boundary as a threshold, and then using the predicted values from our forecasting model as input to the trained data-driven model for anomaly detection.

2 RELATED WORK

Data-driven anomaly detection relies entirely on the intrinsic properties of the data, such as density and distance. They learn the intrinsic properties of the training data and detect anomalies based on their respective criteria. Depending on the criteria, data-driven anomaly detection is generally

---

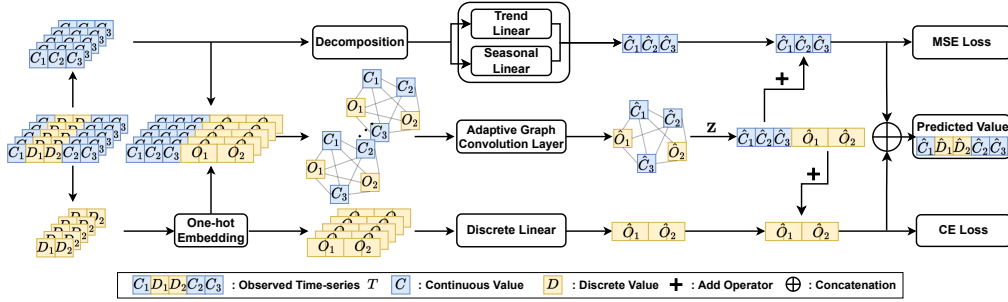*This work was done when he was at Yonsei University.

Figure 1: The architecture of our proposed time-series forecasting model specialized for anomaly detection in multivariate time-series data that considers both continuous and discrete values.

divided into distribution-based methods and boundary-based methods. Distribution-based methods include the Gaussian mixture model (GMM) and empirical cumulative distribution-based outlier detection (ECOD). GMM assumes the training data follows a mixture of Gaussian distributions and uses log-likelihood values to detect outliers. ECOD (Li et al., 2022) estimates the joint cumulative distribution across all features and considers data points in the distribution's tail as anomalies.

Boundary-based methods encompass Support Vector Data Description (SVDD) and Deep Support Vector Data Description (DeepSVDD). SVDD (Schölkopf et al., 1999) obtains a hypersphere in high-dimensional feature space using kernel functions to encompass observations. DeepSVDD (Ruff et al., 2018) employs a neural network to reduce high-dimensional data to a low-dimensional space before applying SVDD. Both approaches treat data points outside the learned hypersphere as anomalies.

3   PROACTIVE ANOMALY DETECTION

3.1   TIME-SERIES FORECASTING MODEL

Proactive anomaly detection is based on a forecasting model for detecting anomalies in advance. Therefore, we design accurate predictive models suitable for anomaly detection datasets containing both continuous and discrete values. For this purpose, our model trains continuous and discrete features separately by each linear layer while utilizing a graph structure. Therefore, our model can consider the unique characteristics of both continuous and discrete features separately and learn the correlations between them simultaneously using the graph structure. This reduces errors in each feature and contributes to overall error reduction (See the Appendix for further details).

3.2   HOW TO PROACTIVELY DETECT ANOMALY

Figure 2 shows a process of proactive time-series anomaly detection. First, we train a data-driven model on the training data. Then, we extract a minimum value of training scores (which indicates a significant difference from the training data) from the trained model and set it as a threshold. Second, we extract the predicted value using testing data as input to the trained forecasting model. Finally, by using the predicted value as input to the trained data-driven model, it classifies the predicted value as normal if the score exceeds the threshold and an anomaly if the score is below the threshold.
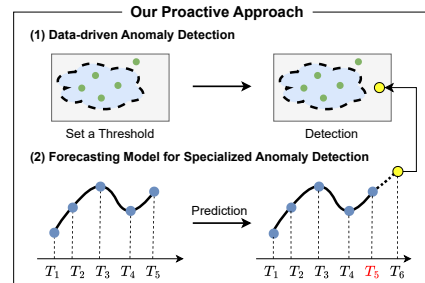

Figure 2: Overall process of our *proactive* anomaly detection approach.

4   EXPERIMENTS

We conducted experiments on four real-world datasets. MSL, SMAP (Hundman et al., 2018), SMD (Su et al., 2019) and PSM (Abdulaal et al., 2021) dataset. We use six time-series anomaly detection models and four time-series forecasting models to compare our forecasting model. To

evaluate our detection model, we use F1-@K (Kim et al., 2022), F1-Composite (Garg et al., 2022), and F1-Range (Wagner et al., 2023), which are mainly used for anomaly detection (See the Appendix for further details).

4.1 PROACTIVE ANOMALY DETECTION PERFORMANCE

Table 1 summarizes the anomaly detection results based on rankings. We determined the ranking of each model for the data-driven models (GMM, ECOD, and DeepSVDD) by anomaly detection performance. After establishing the rankings for the three data-driven models, we calculated an integrated ranking by averaging across all metrics. We presented all results of our experiments in Appendix Table E to H.

Table 1: Results of anomaly detection experiment on the four benchmark datasets. Each number represents the ranking of the anomaly detection results across all models.

Dataset	MSL			SMAP			SMD			PSM		
Metric	F1-@K	F1-C	F1-R	F1-@K	F1-C	F1-R	F1-@K	F1-C	F1-R	F1-@K	F1-C	F1-R
LSTM-P	8	7	7	3	2	3	4	8	9	8	6	8
DeepAnT	2	1	2	9	4	6	1	10	5	10	8	7
TCN-S2S-P	3	4	4	2	6	3	8	9	4	8	10	11
MTAD-GAT	6	3	2	5	3	2	11	6	11	7	9	9
GDN	5	6	6	6	11	7	4	5	7	6	3	6
GTA	4	5	5	11	6	9	9	11	5	11	10	10
NLinear	10	9	10	6	9	8	3	1	3	1	4	2
DLinear	10	11	10	6	9	11	6	3	2	4	4	4
TimesNet	9	9	9	9	1	9	9	7	10	4	7	5
PatchTST	7	8	8	3	4	5	2	3	1	2	2	3
Ours	1	1	1	1	6	1	7	1	8	3	1	1

Among the four benchmark datasets, all datasets except for PSM contain discrete values. Table 1 shows that our model achieved the best performance on MSL and SMAP datasets with the highest ratio of discrete values. For datasets containing only continuous features, such as PSM, our model does not use prediction layers for discrete values. In this way, our forecasting model can be flexible even when there are only continuous features. As a result, it shows excellent performance in anomaly detection.

4.2 IS PROACTIVE ANOMALY DETECTION PRACTICAL?

This discussion question is the most important part of our proactive approach. Figure 3 illustrates that our proactive approach identifies anomalies before abnormal situations occur in the SMAP dataset (Red lines). For example, an anomaly at timestamp 29304 is detected at timestamp 29303 before abnormal situations occur in the SMAP dataset (Figure 3, bottom left). Furthermore, in situations where abnormal situations persist, our proactive approach could minimize potential losses through early detection of the anomaly sample (Figure 3, bottom right).
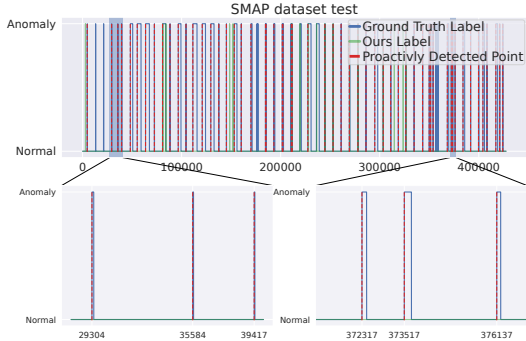

Figure 3: Visualization of proactive anomaly detection in SMAP dataset. The x-axis represents samples over time. Red line is the time point where proactive detection works.

5 FORECASTABLE ANOMALY

Although successful, our proactive approach still has a drawback that makes it impractical. In real-world scenarios, some anomalies accompany small precursors before they happen (e.g., a volcano eruption), but there are also anomalies without precursors. One can consider that our method captures the anomalies with precursors. Therefore, we analyze to verify whether test anomalies can be forecasted by a time-series forecasting model. We rely on the Fourier transform for this analysis, which projects time-series onto orthogonal bases. Forecastable anomalies need to satisfy the following conditions. (i) The set of bases for test anomalies should be identical to that of training data. (ii) The coefficients of the bases for test anomalies should be within the range of those for training data. This predictability analysis using the convex hull of the bases has been conducted in various domains (Yousefzadeh, 2021; Yousefzadeh & Huang, 2020).

To analyze the test time-series with an anomaly in the frequency domain, we divide the time-series data into segments with a time length of 6, mirroring the forecasting process in our experiments, with
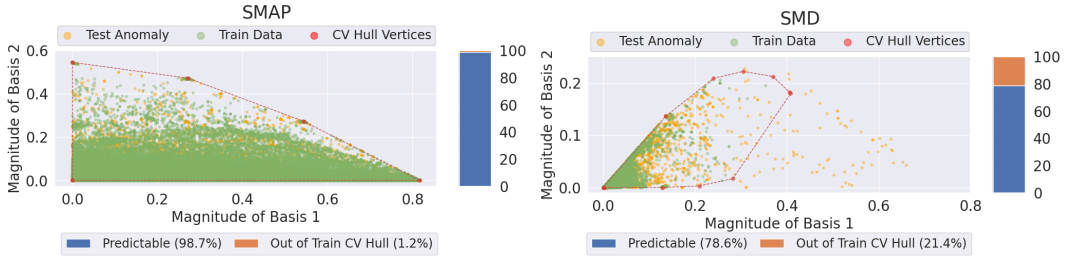
Figure 4: Visualization of forecastable anomaly values. A convex hull (CV hull) is constructed using the magnitudes of the training data, which represent the absolute values of the coefficients. Anomalies are considered unforecastable if they have magnitudes outside the CV hull.

five observations and one last anomaly. Then, each time-series segment $\{T_t\}_{t=1}^6$ can be represented as follows:

$$\{T_t\}_{t=1}^6 = a_1 f_1 + a_2 f_2 + a_3 f_3 + a_4 f_4, \tag{1}$$

where each f_i is a Fourier basis and a_i is its corresponding coefficient calculated by the real-valued discrete Fourier transform (DFT). When the segment length is N, the DFT result is represented by $\frac{N}{2} + 1$ frequencies. In our proposed method, a time-series segment can be expressed using the Fourier bases in $F = \{f_1, f_2, f_3, f_4\}$ with the coefficients $A = (a_1, a_2, a_3, a_4)$. Different time-series segments can have different bases, although their set sizes are always 4. For convenience, we refer to the segmented training data and the segmented testing anomaly data as the training sample and the anomaly sample, respectively.

Given N training samples and M anomaly samples, we first calculate the superset of their bases and then verify whether these supersets are the same, i.e., $\cup_{i=1}^N F_i = \cup_{j=1}^M F_j$, where F_i and F_j are a set of bases for a training and a testing anomaly sample, respectively. Then we find that their supersets are identical in all our datasets used for experiments. Therefore, the first condition is satisfied. This analysis shows that all training and testing samples are in the same manifold, which is an amenable situation for a forecasting model to learn.

For the second condition, we also compare the basis coefficients of the training and anomaly samples. Given N training samples, we first construct a convex polytope using the magnitude of $\{A_i\}_{i=1}^N$, where A_i means the coefficients of the i-th training sample. This convex polytope is a region made by simple interpolations with the magnitude of $\{A_i\}_{i=1}^N$. The interpolation is one of the simplest types of the inferences that deep neural networks can do (Raissi et al., 2019; Tashiro et al., 2021; Harvey et al., 2022; Constantinescu & Popescu, 2023). Therefore, we check whether the coefficients' magnitudes of each testing anomaly sample in $\{A_j\}_{j=1}^M$, where A_j means the coefficients of the j-th anomaly sample, lie inside or outside the convex region.

Figure 4 shows two examples with SMAP and SMD. In each figure, a convex hull with two bases, a two-dimensional convex polytope, is shown since we cannot visualize higher than two in this paper — however, our analysis is done in the complete convex polytope. In the case of the SMAP dataset in Figure 4, it shows that there are very few anomaly samples that are not included in the convex hull of the training data. In total, 1.2% of the anomaly samples are not inside the convex polytope. On the other hand, for the SMD data, around 21% of test time-series with an anomaly fall outside the convex polytope. We expect that it is difficult for our model to perform well on the SMD dataset compared to other datasets including SMAP.

6 CONCLUSION

Time-series anomaly detection is one of the important research topics in real-world applications. However, previous works have applied a *reactive* approach that cannot prevent anomalies in advance. Our *proactive* anomaly detection is a forecasting-based approach specialized for anomaly detection in multivariate time-series data with continuous and discrete features. Thus, it is able to detect and prevent anomalies in advance by accurate forecasting. Our experiments incorporating real-world datasets and various metrics support the efficacy of our specialized forecasting model in time-series anomaly detection. In conclusion, we analyze both predictable and unpredictable anomalies in our proactive anomaly detection system, thereby leaving room for unpredictable anomalies.

REFERENCES

Ahmed Abdulaal, Zhuanghua Liu, and Tomer Lancewicki. Practical approach to asynchronous multivariate time series anomaly detection and localization. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 2485–2494, 2021.

Zekai Chen, Dingshuo Chen, Xiao Zhang, Zixuan Yuan, and Xiuzhen Cheng. Learning graph structures with transformer for multivariate time-series anomaly detection in iot. *IEEE Internet of Things Journal*, 9(12):9179–9189, 2021.

Vlad-Raul Constantinescu and Ionel Popescu. Interpolation property of shallow neural networks. *arXiv preprint arXiv:2304.10552*, 2023.

Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI conference on artificial intelligence*, pp. 4027–4035, 2021.

Astha Garg, Wenyu Zhang, Jules Samaran, Ramasamy Savitha, and Chuan-Sheng Foo. An evaluation of anomaly detection and diagnosis in multivariate time series. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2508–2517, 2022. doi: 10.1109/TNNLS.2021.3105827.

William Harvey, Saeid Naderiparizi, Vaden Masrani, Christian Weilbach, and Frank Wood. Flexible diffusion modeling of long videos. *Advances in Neural Information Processing Systems*, 35: 27953–27965, 2022.

Yangdong He and Jiabao Zhao. Temporal convolutional networks for anomaly detection in time series. In *Journal of Physics: Conference Series*, pp. 042050. IOP Publishing, 2019.

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 387–395, 2018.

Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a rigorous evaluation of time-series anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 7194–7201, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George Chen. ECOD: Unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, Puneet Agarwal, et al. Long short term memory networks for anomaly detection in time series. In *Esann*, volume 2015, pp. 89, 2015.

Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, 7:1991–2005, 2018.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pp. 4393–4402. PMLR, 2018.

Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12, 1999.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2828–2837, 2019.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.

Dennis Wagner, Tobias Michels, Florian C. F. Schulz, Arjun Nair, Maja Rudolph, and Marius Kloft. TimeSeAD: Benchmarking deep multivariate time-series anomaly detection. *Trans. Mach. Learn. Res.*, 2023, 2023. URL https://openreview.net/forum?id=iMmsCI0JsS.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference*, pp. 187–196, 2018.

Roozbeh Yousefzadeh. Deep learning generalization and the convex hull of training sets. *arXiv preprint arXiv:2101.09849*, 2021.

Roozbeh Yousefzadeh and Furong Huang. Using wavelets and spectral methods to study patterns in image-classification datasets. *arXiv preprint arXiv:2006.09879*, 2020.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, pp. 11121–11128, 2023.

Hang Zhao, Yujing Wang, Juanyong Duan, Congrui Huang, Defu Cao, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Multivariate time-series anomaly detection via graph attention network. In *2020 IEEE International Conference on Data Mining (ICDM)*, pp. 841–850. IEEE, 2020.

A FORECASTING MODEL FOR DETECTING ANOMALIES

In this section, we describe detailed a time-series forecasting model specialized for anomaly detection. For convenience of understanding, notations are defined as follows:

- Observed time-series $\boldsymbol{T} \in \mathbb{R}^{(c+d) \times N}$ is denoted by a set of time points $\{T_1, T_2, \cdots, T_N\}$.

- The observed value at time t, $T_t \in \mathbb{R}^{c+d}$ contains c continuous columns and d discrete columns. It is expressed as $T_t = \{\mathbf{C}^t, \mathbf{D}^t\}$ at time t.

- $\mathbf{C}^t \in \mathbb{R}^c$ is denoted by a set of columns with a continuous value $\{C_1, C_2, \cdots, C_c\}^t$ at time t.

- $\mathbf{D}^t \in \mathbb{R}^d$ is denoted by a set of columns with a discrete value $\{D_1, D_2, \cdots, D_d\}^t$ at time t.

A.1 EXTRACT CONTINUOUS & DISCRETE FEATURES.

First, continuous and discrete features are separately extracted for prediction according to the data characteristics. Therefore, we divide continuous and discrete values for observed values $\{T_t\}_{t=1}^N \in \mathbb{R}^{(c+d) \times N}$ as follows:

$$\begin{aligned}
\{T_t\}_{t=1}^N &= \{\mathbf{C}^t, \mathbf{D}^t\}_{t=1}^N, \\
&= \{\mathbf{C}^t\}_{t=1}^N, \{\mathbf{D}^t\}_{t=1}^N.
\end{aligned} \tag{2}$$

Then, in time-series prediction, the model considering trend and seasonality shows simple but excellent performance (Zeng et al., 2023), so we decompose continuous values into trend and seasonality. To extract the trend $\mathbf{C_T}$ from the continuous values, we use the moving average method, and the seasonality $\mathbf{C_S}$ is taken as the remaining value after subtracting the trend from the continuous values.

$$\begin{aligned}
\{\mathbf{C_T}^t\}_{t=1}^N &= \text{AvgPool}(\text{Padding}(\{\mathbf{C}^t\}_{t=1}^N)), \\
\{\mathbf{C_S}^t\}_{t=1}^N &= \{\mathbf{C}^t\}_{t=1}^N - \{\mathbf{C_T}^t\}_{t=1}^N,
\end{aligned} \tag{3}$$

where AvgPool means average pooling and Padding means pre-padding with the first value and post-padding with the last value. Then, using each component as input to a linear layer, each hidden vector is extracted as follows:

$$\begin{aligned}
\hat{\mathbf{C}}_\mathbf{T}^{N+1} &= \text{Linear}_{trend}(\{\mathbf{C_T}^t\}_{t=1}^N), \\
\hat{\mathbf{C}}_\mathbf{S}^{N+1} &= \text{Linear}_{seasonality}(\{\mathbf{C_S}^t\}_{t=1}^N),
\end{aligned} \tag{4}$$

where Linear_{trend} and $\text{Linear}_{seasonality}$ are linear layers that predict the next time point $N+1$ with N observed time points. After the hidden vectors of each trend and seasonality are extracted, reconstruct continuous features by adding each hidden vector.

$$\hat{\mathbf{C}}^{N+1} = \hat{\mathbf{C}}_\mathbf{T}^{N+1} + \hat{\mathbf{C}}_\mathbf{S}^{N+1}, \tag{5}$$

where $\hat{\mathbf{C}}^{N+1} \in \mathbb{R}^c$ is denoted by a set of $\{\hat{C}_1, \hat{C}_2, \cdots, \hat{C}_c\}^{N+1}$.

For discrete values, due to the difficulty in explaining trend and seasonality, they are converted into one-hot vectors and used as inputs for a linear layer to extract hidden vectors. The process is as follows:

$$\begin{aligned}
\{\mathbf{O}^t\}_{t=1}^N &= \text{One-Hot}(\{\mathbf{D}^t\}_{t=1}^N), \\
\hat{\mathbf{O}}^{N+1} &= \text{Linear}_{discrete}(\{\mathbf{O}^t\}_{t=1}^N),
\end{aligned} \tag{6}$$

where One-Hot indicates a one-hot embedding, $\mathbf{O}^t \in \mathbb{R}^{d \times e}$ is a set of one-hot vectors of discrete columns, e is an embedding dimension and $\hat{\mathbf{O}}^{N+1} \in \mathbb{R}^{d \times e}$ denotes a set of predicted one-hot vectors of discrete columns $\{\hat{O}_1, \hat{O}_2, \cdots, \hat{O}_d\}^{N+1}$. Then, $\hat{O}_t \in \mathbb{R}^e$ is a predicted one-hot vector of discrete value. $\text{Linear}_{discrete}$ is a linear layer that predicts the next time point $N+1$ with N observed time points.

A.2 ADAPTIVE GRAPH CONVOLUTION LAYER.

Second, we utilize an adaptive graph convolution (AGC) layer. The lack of learning dependencies between features due to the separate handling of continuous and discrete values can be addressed by using an adaptive adjacency matrix. The adaptive adjacency matrix is formed by multiplying learnable embedding and is defined as follows:

$$\mathbf{A} = \mathbf{I} + \text{softmax}(\text{ReLU}(\mathbf{E}\mathbf{E}^{\mathsf{T}})), \tag{7}$$

where $\mathbf{E} \in \mathbb{R}^{(c+d)\times b}$ is a trainable node-embedding matrix with embedding dimension b, $\mathbf{I} \in \mathbb{R}^{(c+d)\times(c+d)}$ is the identity matrix, and $\mathbf{A} \in \mathbb{R}^{(c+d)\times(c+d)}$ is the learned adjacent matrix of the graph representing the proximity between time-series features.

We combine the adaptive adjacency matrix and graph convolution network. To utilize AGC, the one-hot embedding of continuous features and discrete features are concatenated into $\mathbf{X} \in \mathbb{R}^{(c+d)\times(e\times N)}$. $\mathbf{X}$ is reshaped from the $\{\tilde{\mathbf{C}}^t, \mathbf{O}^t\}_{t=1}^N \in \mathbb{R}^{(c+d)\times e\times N}$, where $\tilde{\mathbf{C}}^t = [\mathbf{C}^t, \mathbf{0}] \in \mathbb{R}^{(c+d)\times e}$ is zero-padded continuous features. Let $\mathbf{H} \in \mathbb{R}^{(c+d)\times h}$ is the matrix of node features with hidden dimension h transformed by Linear_{input}. Then the AGC layer outputs $\mathbf{Z}$ through the following mapping:

$$\mathbf{X} = \text{reshape}(\{\tilde{\mathbf{C}}^t, \mathbf{O}^t\}_{t=1}^N), \tag{8}$$
$$\mathbf{H} = \text{Linear}_{input}(\mathbf{X}), \tag{9}$$
$$\mathbf{Z} = \sigma(\mathbf{A}\mathbf{H}\mathbf{E}\mathbf{W}), \tag{10}$$

where $\mathbf{W} \in \mathbb{R}^{b\times h\times e}$ is a trainable weight transformation matrix and $\sigma(\cdot)$ is the activation function. $\mathbf{Z} \in \mathbb{R}^{(c+d)\times e}$ is the result of adaptive graph convolution and will be utilized in the prediction stage.

A.3 PREDICTION.

Finally, the dependencies between features are added to each extracted discrete and continuous hidden vector. Through this, it is possible to make predictions considering the dependence between the separately extracted hidden vectors.

$$\hat{\mathbf{C}}^{N+1} = \hat{\mathbf{C}}^{N+1} + \mathbf{Z}_{:c,1},$$
$$\hat{\mathbf{O}}^{N+1} = \hat{\mathbf{O}}^{N+1} + \mathbf{Z}_{-d:,:}, \tag{11}$$

where $\mathbf{Z}_{:c,1} \in \mathbb{R}^c$ is continuous columns and $\mathbf{Z}_{-d:,:} \in \mathbb{R}^{d\times e}$ is discrete columns in $\mathbf{Z}$.

After that, continuous features are trained with the mean square error (MSE) loss as follows:

$$Loss_{\mathbf{C}} = \frac{\sum_{i=1}^c (C_i^{N+1} - \hat{C}_i^{N+1})^2}{c}, \tag{12}$$

where C_i^{N+1} is the i-th element of $\mathbf{C}^{N+1}$, c is the number of continuous elements. For discrete features, we need to train the model with cross-entropy (CE) loss for each discrete feature as follows:

$$Loss_{\mathbf{D}} = \frac{\sum_{i=1}^d \sum_{j=1}^e -O_{ij}^{N+1} log(\hat{O}_{ij}^{N+1})}{d}, \tag{13}$$

where O_{ij}^{N+1} is the j-th one-hot vector element in i-th one hot vector O_i^{N+1}. e is one-hot embedding dimension and d is the number of discrete elements. The total loss of our prediction model is:

$$Loss_{\textbf{Total}} = Loss_{\mathbf{C}} + Loss_{\mathbf{D}}. \tag{14}$$

A.4 TRAINING ALGORITHM

We present the training process of the proposed time-series forecasting model in Algorithm 1. At each iteration, we first divide the continuous and discrete values in the observed time-series T_{train}. Each of the continuous values and discrete values is passed through separate prediction models to predict $\hat{\mathbf{C}}^{N+1}$ and $\hat{\mathbf{O}}^{N+1}$. In order to consider the dependency between values regardless of continuous or discrete, $\mathbf{Z}$ is extracted by passing through the adaptive graph convolution (AGC) layer with the observed time-series T_{train} as an input. Then add $\mathbf{Z}$ to $\hat{\mathbf{C}}^{N+1}$ and $\hat{\mathbf{O}}^{N+1}$. Finally, the prediction model is trained with the mean squared error loss for continuous values and the cross entropy loss for discrete values.

Algorithm 1: How to train time-series forecasting model

Input: training time-series data $\boldsymbol{T}_{\text{Train}}$, Iteration number of prediction model K_{Pred}
Parameter: Prediction model $\boldsymbol{\theta}_{\text{Pred}}$
Output: Prediction model $\boldsymbol{\theta}_{\text{Pred}}$

1: Initialize $\boldsymbol{\theta}_{\text{Pred}}$
2: $k \leftarrow 0$
3: **while** $k < K_{\text{Pred}}$ **do**
4:    $\{\mathbf{C}^t\}_{t=1}^N, \{\mathbf{D}^t\}_{t=1}^N \leftarrow \boldsymbol{T}_{\text{Train}}$
5:    $\{\mathbf{C_T}^t\}_{t=1}^N, \{\mathbf{C_S}^t\}_{t=1}^N \leftarrow \text{Decomp}\{\mathbf{C}^t\}_{t=1}^N$
6:    $\hat{\mathbf{C}}_\mathbf{T}^{N+1} \leftarrow \text{Linear}_{trend}(\{\mathbf{C_T}^t\}_{t=1}^N)$
7:    $\hat{\mathbf{C}}_\mathbf{S}^{N+1} \leftarrow \text{Linear}_{seasonal}(\{\mathbf{C_S}^t\}_{t=1}^N)$
8:    $\hat{\mathbf{C}}^{N+1} \leftarrow \hat{\mathbf{C}}_\mathbf{T}^{N+1} + \hat{\mathbf{C}}_\mathbf{S}^{N+1}$
9:    $\mathbf{O}^{N+1} \leftarrow \text{One-hot}(\{\mathbf{D}^t\}_{t=1}^N)$
10:    $\hat{\mathbf{O}}^{N+1} \leftarrow \text{Linear}_{discrete}(\{\mathbf{O}^t\}_{t=1}^N)$
11:    $\mathbf{Z} \leftarrow \text{AGC layer with Eq. 7 to Eq. 10}$
12:    $\hat{\mathbf{C}}^{N+1} = \hat{\mathbf{C}}^{N+1} + \mathbf{Z}_{:c,1}$
13:    $\hat{\mathbf{O}}^{N+1} = \hat{\mathbf{D}}^{N+1} + \mathbf{Z}_{-d:,:}$
14:    **if** Continuous value **then**
15:      $Loss_\mathbf{C} = \frac{\sum_{i=1}^{c}(C_i^{N+1} - \hat{C}_i^{N+1})^2}{c}$
16:    **end if**
17:    **if** Discrete value **then**
18:      $Loss_\mathbf{D} = \frac{\sum_{i=1}^{d}\sum_{j=1}^{e} -O_{ij}^{N+1} log(\hat{O}_{ij}^{N+1})}{d}$
19:    **end if**
20:    $Loss_{\text{Total}} \leftarrow Loss_\mathbf{C} + Loss_\mathbf{D}$
21:    Update $\boldsymbol{\theta}_{\text{Pred}}$ with $Loss_{\text{Total}}$
22: **end while**
23: **return** Prediction model $\boldsymbol{\theta}_{\text{Pred}}$

B   Experimental Environments

Our detailed software and hardware environments are as follows: Ubuntu 18.04 LTS, Python 3.9.12, CUDA 11.4, NVIDIA Driver 525.125.06 i9 CPU, and GeForce RTX A5000 & A6000.

B.1   Dataset

We used four time-series datasets for our experiments. Summary of the datasets in Table A of the main manuscript.

- Mars Science Laboratory rover and Soil Moisture Active Passive satellite (Hundman et al., 2018) datasets are from NASA. Mars Science Laboratory (MSL) rover contains 55 features, consisting of 54 discrete features and 1 continuous feature. The anomaly proportion in MSL testing data is approximately 10.5%. Soil Moisture Active Passive (SMAP) satellite contains 25 features, consisting of 24 discrete features and 1 continuous feature. The anomaly proportion in SMAP testing data is approximately 12.8%.

- Server Machine Dataset (Su et al., 2019) is collected by large internet company. Server Machine Dataset (SMD) has 38 features, two of which are discrete and 36 of which are continuous. The anomaly proportion in SMD testing data is around 4.16%.

- Pooled Server Metrics (Abdulaal et al., 2021) dataset is provided by eBay by capturing internally from application server nodes. Pooled Server Metrics (PSM) contains 25 features with no discrete feature. Among the testing data, the proportion of anomalies is approximately 27.76%.

Table A: Summary of the datasets. Ratio (%) represents the percentage of anomaly in the testing dataset.

Dataset	Train	Validation	Test	# of Cont (c)	# of Disc (d)	Ratio (%)
MSL	46,655	11,662	73,729	1	54	10.5
SMAP	108,148	27,035	427,617	1	24	12.8
SMD	566,725	141,680	708,420	36	2	4.16
PSM	103,289	26,495	87,841	25	0	27.76

B.2 BASELINES

To compare the performance of the proposed model, we utilized several prediction-based time-series anomaly detection models and time-series forecasting models as baselines.

- LSTM-P (Malhotra et al., 2015) uses two-layer stacked LSTM network and a fully connected layer for final forecasting.
- DeepAnT (Munir et al., 2018) uses a CNN-based prediction model with two 1D convolution layers and two max pooling, and a fully connected layer.
- TCN-S2S-P (He & Zhao, 2019) applies a temporal convolutional network (TCN) with 1D dilated causal convolutions to time-series anomaly detection.
- MTAD-GAT (Zhao et al., 2020) learns complex dependencies in time-series using two graph attention layers: temporal and feature dimensions.
- GDN (Deng & Hooi, 2021) is a graph-based model that has explainability for anomalies with structure learning and attention weights.
- GTA (Chen et al., 2021) is a transformer-based model that learns a graph structure automatically and takes into consideration the long-term temporal dependencies.
- NLinear and DLinear (Zeng et al., 2023) are linear-based models that utilize temporal information with a linear layer. NLinear utilizes the normalization of time-series data, and DLinear utilizes the decomposition of time-series data.
- TimesNet (Wu et al., 2022) is a Timesblock architecture based model, which incorporates a 2D backbone. It transforms 1D time series into a 2D space and analyzes the resulting 2D tensor using various 2D vision backbones. This allows it to effectively capture intra and interperiodic variations within the time series.
- PatchTST (Nie et al., 2022) is a transformer-based model that uses subseries-level patches of time series as input and has channel independence by processing multivariate time series as a single time series.

B.3 THRESHOLD MODELS

We provide detailed information about GMM, ECOD, and DeepSVDD used as threshold models.

- GMM is a density-based model that learns the density of the data and identifies points that do not fit well within that density as anomalies. The farther a data point deviates from the distribution, the lower the score it receives through GMM. Consequently, the lowest score value among the training data is used as the threshold.
- ECOD (Li et al., 2022) is a density-based model, which learns the density of the data and considers points located in the both tail parts of the density distribution as anomalies. As a data point gets closer to the tail of the distribution, its score through ECOD tends to be higher. Hence, the threshold in ECOD-based anomaly detection is set to the highest score value among the training data.
- DeepSVDD (Ruff et al., 2018) is a boundary-based model to find the smallest hypersphere that includes the normal data on the latent feature space. If a data point is far outside the learned sphere, its score get higher and then that point will likely be classified to be an anomaly.

Table B: Components of our forecasting model.

Layer	Design	Input size	Output size
1	Linear_{trend}	$5 \times c$	$1 \times c$
2	$\text{Linear}_{seasonality}$	$5 \times c$	$1 \times c$

(a) Continuous model.

Layer	Design	Input size	Output size
1	$\text{Linear}_{discrete}$	$5 \times (d \times e)$	$1 \times (d \times e)$

(b) Discrete model.

Layer	Design	Input size	Output size
1	Linear_{input}	$(c+d) \times e$	$h \times (c+d) \times e$
2	Linear_{output}	$h \times (c+d) \times e$	$1 \times (c+d) \times e$
3	squeeze	$1 \times (c+d) \times e$	$(c+d) \times e$

(c) Adaptive Graph Convolution Model.

B.4 EVALUATION METRICS

To evaluate the time-series anomaly detection performance of our proposed model and baselines, we consider three evaluation metrics as follows:

- F1-@K (Kim et al., 2022) first computes F1-score with the evaluation scheme in which all observations are considered correctly detected if the proportion of correctly detected anomalies in the consecutive anomaly segment exceeds the predefined criterion K. We use the metric F1-@K as the area under the curve of F1-score where K varies by 0.1 from 0 to 1 to mitigate the overestimation of point-adjusted F1-score (Xu et al., 2018) and the underestimation of point-wise F1-score.

- F1-Composite (Garg et al., 2022) is calculated as the harmonic mean of point-wise precision and segment-wise recall for robust evaluation of segment-wise anomaly detection.

- F1-Range (Wagner et al., 2023) considers time-series precision and recall with a set of actual anomaly segments and a set of predicted anomaly segments in order to overcome the problem that point-wise F1-score fails to discriminate predictive patterns by ignoring temporal dependencies.

C DETAILED SETTINGS OF EXPERIMENTS

We introduce the detailed setting of our experiments including model structure and the best hyperparameter. Here, we set the same sliding window size of 5 and prediction horizon length of 1 including baseline models. Additionally, we use the Adam (Kingma & Ba, 2014) optimizer and set the learning rate to 0.005. Our forecasting models include continuous and discrete models, respectively, and an adaptive graph convolution model that extracts dependencies between features. Where c and d are the number of continuous and discrete features, e is the one-hot embedding size of discrete features and h is the hidden vector of the graph structure (cf. Table B). Finally, we use λ for balancing between continuous features and discrete features in training loss. For each of the reported results, we list the best hyperparameter as follows:

- For MSL, $c = 1$, $d = 54$, $h = 256$, $e = 2$, $\lambda = 1$;
- For SMAP, $c = 1$, $d = 24$, $h = 256$, $e = 2$, $\lambda = 1$;
- For SMD, $c = 36$, $d = 2$, $h = 256$, $e = 16$, $\lambda = 1$;
- For PSM, $c = 25$, $e = 3$, $h = 256$;

D EXPERIMENT RESULTS

In this section, we present the detailed results of our experiments in Table E to H.

D.1 FORECASTING PERFORMANCE

Table D: Mean of evaluation metric and its standard deviation (std). Each value is represented as mean ± std. Ground Truth is the result of fitting the testing data to the trained data-driven model. Bold is the most similar performance to Ground Truth.

Dataset-Data-driven Model		MSL-GMM			SMAP-ECOD			PSM-DeepSVDD		
Metric		F1-@K	F1-C	F1-R	F1-@K	F1-C	F1-R	F1-@K	F1-C	F1-R
Unsupervised Time-series Anomaly Model	LSTM-P	0.1906 ± 0.0000	0.1906 ± 0.0000	0.1905 ± 0.0000	0.2031 ± 0.0146	0.1792 ± 0.0124	0.1491 ± 0.0171	0.4571 ± 0.0031	**0.4347 ± 0.0024**	0.4304 ± 0.0024
	DeepAnT	0.0451 ± 0.0026	**0.2179 ± 0.0323**	0.0404 ± 0.0284	0.1981 ± 0.0276	0.2040 ± 0.0217	0.1669 ± 0.0163	0.4572 ± 0.0017	0.4358 ± 0.0012	0.4316 ± 0.0014
	TCN-S2S-P	0.1906 ± 0.0000	0.1906 ± 0.0000	0.1905 ± 0.0000	0.1284 ± 0.0143	0.1464 ± 0.0060	0.0918 ± 0.0128	0.4505 ± 0.0033	0.4307 ± 0.0043	0.4241 ± 0.0049
	MTAD-GAT	0.1906 ± 0.0000	0.1906 ± 0.0000	0.1905 ± 0.0000	0.1896 ± 0.0070	0.1749 ± 0.0050	0.1547 ± 0.0045	0.4560 ± 0.0021	0.4308 ± 0.0041	0.4263 ± 0.0037
	GDN	0.1906 ± 0.0000	0.1906 ± 0.0000	0.1905 ± 0.0000	0.1351 ± 0.0082	0.1413 ± 0.0012	0.0968 ± 0.0058	0.4601 ± 0.0090	0.4341 ± 0.0010	0.4262 ± 0.0033
	GTA	0.1906 ± 0.0000	0.1906 ± 0.0000	0.1905 ± 0.0000	0.2348 ± 0.0230	0.2020 ± 0.0164	0.1731 ± 0.0159	0.4631 ± 0.0059	0.4428 ± 0.0059	0.4357 ± 0.0065
Time-series Prediction Model	NLinear	0.2451 ± 0.0000	0.1996 ± 0.0000	0.1858 ± 0.0000	0.0435 ± 0.0000	0.1989 ± 0.0001	0.0208 ± 0.0000	**0.4458 ± 0.0001**	0.4332 ± 0.0001	0.4311 ± 0.0002
	DLinear	0.1906 ± 0.0000	0.1906 ± 0.0000	0.1905 ± 0.0000	0.0426 ± 0.0004	0.1990 ± 0.0055	0.0194 ± 0.0008	0.4459 ± 0.0000	0.4331 ± 0.0001	0.4312 ± 0.0002
	TimesNet	0.1906 ± 0.0000	0.1906 ± 0.0000	0.1905 ± 0.0000	0.3048 ± 0.0525	**0.2684 ± 0.0484**	0.2153 ± 0.0389	0.4511 ± 0.0021	0.4369 ± 0.0031	0.4337 ± 0.0028
	PatchTST	0.1906 ± 0.0000	0.1906 ± 0.0000	0.1905 ± 0.0000	0.0423 ± 0.0006	0.2021 ± 0.0047	0.0194 ± 0.0009	0.4459 ± 0.0001	0.4346 ± 0.0001	**0.4326 ± 0.0002**
OURS		**0.0837 ± 0.0014**	0.1972 ± 0.0001	**0.0857 ± 0.0004**	**0.0210 ± 0.0000**	0.1650 ± 0.0000	**0.0014 ± 0.0000**	0.4460 ± 0.0002	0.4350 ± 0.0001	0.4331 ± 0.0001
Ground Truth		0.0916	0.3428	0.0937	0.0290	0.2908	0.0046	0.4437	0.4347	0.4322

Table C shows the prediction performance of forecasting models. Since existing models consider discrete features as continuous features, our model also treats discrete features as continuous features during the evaluation process and only uses mean square error (MSE).

Evaluating the model only with MSE is disadvantageous to our model, which trained discrete features with cross-entropy (CE) loss. Nonetheless, our forecasting model shows comparable or better performance than the latest time-series forecasting models. Additionally, we will show that evaluating predictive performance using MSE does not guarantee the performance of proactive anomaly detection in the next section.

Table C: Results of forecasting performance on the four benchmark datasets. Each value represents the mean of the evaluation metric and its standard deviation (Std).

Dataset	MSL	SMAP	SMD	PSM
Mean Squared Error (MSE)	Mean ± Std.	Mean ± Std.	Mean ± Std.	Mean ± Std.
LSTM-P	1.7052 ± 0.0348	0.0113 ± 0.0000	0.0016 ± 0.0000	0.0020 ± 0.0001
DeepAnT	1.2387 ± 0.9148	0.0314 ± 0.0105	0.0040 ± 0.0002	0.0014 ± 0.0005
TCN-S2S-P	1.0653 ± 0.1072	0.0926 ± 0.0092	0.0233 ± 0.0026	0.0027 ± 0.0000
MTAD-GAT	1.8407 ± 0.0081	**0.0105 ± 0.0005**	0.0013 ± 0.0000	0.0016 ± 0.0002
GDN	20.824 ± 27.073	0.0230 ± 0.0014	0.0061 ± 0.0024	0.0015 ± 0.0003
GTA	1.8140 ± 0.0618	0.0370 ± 0.0062	0.0186 ± 0.0007	0.0042 ± 0.0004
NLinear	0.0671 ± 0.0001	0.0188 ± 0.0000	0.0013 ± 0.0000	**0.0001 ± 0.0000**
DLinear	0.8747 ± 0.0934	0.0938 ± 0.0136	0.0013 ± 0.0000	**0.0001 ± 0.0000**
TimesNet	**0.0278 ± 0.0513**	**0.0044 ± 0.0004**	0.0014 ± 0.0001	0.0010 ± 0.0003
PatchTST	0.0650 ± 0.0028	0.0124 ± 0.0002	**0.0011 ± 0.0000**	**0.0001 ± 0.0000**
Ours	0.0289 ± 0.0016	0.0094 ± 0.0001	0.0013 ± 0.0000	**0.0001 ± 0.0000**

comparable or better performance than the latest time-series forecasting models. Additionally, we will show that evaluating predictive performance using MSE does not guarantee the performance of proactive anomaly detection in the next section.

D.2 ANOMALY DETECTION PERFORMANCE

Diff represents the difference between the Ground Truth and the result of using the predicted value as an input to the threshold model. The closer to 0, the more similar the prediction to the ground truth value. For models that did not predict well and judged all samples to be anomalies, the Diff value is displayed as -. In some cases, there were models with a lower Diff than our forecasting model, but those models had a much greater variance in forecasting performance than our forecasting model (cf. Table C). Therefore, we can find the anomaly in advance through accurate prediction.

For a detailed comparison, we reported a comprehensive analysis by presenting the mean and standard deviation derived from five repeated experiments in Table D. The ground truth in the last row of Table D is the evaluation of testing data by the data-driven model trained with training data. In other words, if the predicted values of each model are similar to the testing data, they are identical to the ground truth. Therefore, the goal of our experiment is for the evaluation scores of predicted values, as determined by the data-driven model, to closely resemble those of the ground truth values.

When GMM is used as the data-driven model in the MSL dataset (MSL-GMM), our model shows the best anomaly detection performance. In addition, all the models except for DeepAnT, NLinear, and ours, predict all samples as anomalies, showing the same performance. In SMAP-ECOD, our model also shows the best and the most consistent performance. These results also imply that the prediction performance of our model has low variability, as shown in most of the results. In PSM-DeepSVDD, our model does not show the best performance, but the performance difference with the best model is very small, up to 0.0005.

In Table D, our model shows better performance in terms of F1-@K and F1-Range when evaluated on MSL-GMM and SMAP-ECOD. However, it shows worse performance in the F1-Composite. To analyze these differences, we visualize the evaluation of predicted values by a trained data-driven model. Figure A (left) reveals that there is a big difference between the ground truth and predicted values of TimesNet. As a result, TimesNet identifies a majority of the samples as anomalies includ-

ing normal samples within testing data, showing better performance than our model F1-Composite in Table D.

Although TimesNet shows better prediction performance in terms of MSE compared to our model, our model shows more similar anomaly scores for each time point than TimesNet. As visualized in the right panel of Figure A, TimesNet produces continuous values for discrete features, while our model predicts properly discrete values. As a result, even though our model shows worse forecasting performance in terms of MSE, our model forecasts values that are more closely aligned with the ground truth compared to other models.
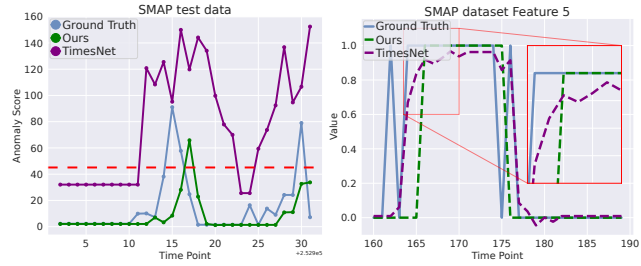

Figure A: Visualization of comparison between ours and TimeNet in the SMAP dataset. *Left*: Anomaly score by the trained ECOD. *Right*: Predicted values for the categorical feature.

Table E: Results of anomaly detection experiment on MSL dataset. When the data-driven model decides all samples to be abnormal, the F1 score is 0.1906.

Data-driven Model		GMM						ECOD						DeepSVDD					
Metrics		F1-@K	Diff	F1-C	Diff	F1-R	Diff	F1-@K	Diff	F1-C	Diff	F1-R	Diff	F1-@K	Diff	F1-C	Diff	F1-R	Diff
Unsupervised Time-series Anomaly Model	LSTM-P	0.1906	-	0.1906	-	0.1905	-	0.1906	-	0.1906	-	0.1905	-	0.2026	0.0296	0.1982	0.0511	0.1973	0.0562
	DeepAnT	0.0451	0.0465	**0.2179**	**0.1249**	0.0404	0.0533	0.1906	-	0.1906	-	0.1905	-	0.1906	0.0176	0.1906	0.0435	0.1905	0.0494
	TCN-S2S-P	0.1906	-	0.1906	-	0.1905	-	0.1906	-	0.1906	-	0.1905	-	0.1908	0.0178	0.1908	0.0437	0.1907	0.0496
	MTAD-GAT	0.1906	-	0.1906	-	0.1905	-	0.1906	-	0.1906	-	0.1905	-	0.2007	0.0277	**0.1846**	**0.0375**	**0.1812**	**0.0401**
	GDN	0.1906	-	0.1906	-	0.1905	-	0.1906	-	0.1906	-	0.1905	-	0.1970	0.0240	0.1958	0.0487	0.1953	0.0542
	GTA	0.1906	-	0.1906	-	0.1905	-	0.1906	-	0.1906	-	0.1905	-	0.1913	0.0183	0.1910	0.0439	0.1908	0.0497
Time-series Prediction Model	NLinear	0.2451	0.1535	0.1996	0.1432	0.1858	0.0921	0.1906	-	0.1906	-	0.1905	-	0.2222	0.0492	0.2178	0.0707	0.2150	0.0739
	DLinear	0.1906	-	0.1906	-	0.1905	-	0.1906	-	0.1906	-	0.1905	-	0.2213	0.0483	0.2169	0.0698	0.2142	0.0731
	TimesNet	0.1906	-	0.1906	-	0.1905	-	0.1906	-	0.1906	-	0.1905	-	0.2048	0.0318	0.2013	0.0542	0.2002	0.0591
	PatchTST	0.1906	-	0.1906	-	0.1905	-	0.1906	-	0.1906	-	0.1905	-	0.2014	0.0284	0.1996	0.0525	0.1987	0.0576
Ours		**0.0837**	**0.0079**	0.1972	0.1456	**0.0853**	**0.0084**	**0.2279**	**0.2136**	**0.1940**	**0.0440**	**0.1774**	**0.1750**	**0.1902**	**0.0172**	0.1870	0.0399	0.1869	0.0458
Ground Truth		0.0916		0.3428		0.0937		0.0143		0.1500		0.0024		0.1730		0.1471		0.1411	

Table F: Results of anomaly detection experiment on SMAP dataset. When the data-driven model decides all samples to be abnormal, the F1 score is 0.2268.

Data-driven Model		GMM						ECOD						DeepSVDD					
Metrics		F1-@K	Diff	F1-C	Diff	F1-R	Diff	F1-@K	Diff	F1-C	Diff	F1-R	Diff	F1-@K	Diff	F1-C	Diff	F1-R	Diff
Unsupervised Time-series Anomaly Model	LSTM-P	0.2269	0.1902	0.2268	-	0.2268	-	0.2031	0.1741	0.1792	0.1116	0.1491	0.1445	0.2501	0.0020	0.2393	0.0034	0.2332	0.0032
	DeepAnT	0.0001	0.0366	0.0059	0.3126	0.0	0.0078	0.1981	0.1691	0.2040	0.0868	0.1669	0.1623	0.2268	0.0253	0.2268	0.0159	0.2268	0.0096
	TCN-S2S-P	0.2268	-	0.2268	-	0.2268	-	0.1284	0.0994	0.1464	0.1444	0.0918	0.0872	**0.2528**	**0.0007**	**0.2438**	**0.0011**	**0.2374**	**0.0010**
	MTAD-GAT	0.2281	0.1914	0.2266	0.0919	0.2263	0.2185	0.1896	0.1606	0.1749	0.1159	0.1547	0.1501	0.2491	0.0030	0.2394	0.0033	0.2334	0.0030
	GDN	0.2268	-	0.2268	-	0.2268	-	0.1351	0.1061	0.1413	0.1495	0.0968	0.0922	0.2385	0.0136	0.2319	0.0108	0.2284	0.0080
	GTA	0.2277	0.1910	0.2268	-	0.2269	0.2191	0.2348	0.2058	0.2020	0.0888	0.1731	0.1685	0.2717	0.0196	0.2540	0.0113	0.2444	0.0080
Time-series Prediction Model	NLinear	0.1373	0.1006	0.1402	0.1783	0.0979	0.0901	0.0435	0.0145	0.1989	0.0919	0.0208	0.0162	0.2315	0.0206	0.2258	0.0169	0.2229	0.0135
	DLinear	0.2268	-	0.2268	-	0.2268	-	0.0426	0.0136	0.1990	0.0918	0.0194	0.2222	0.2309	0.0212	0.2254	0.0173	0.2225	0.0139
	TimesNet	0.2358	0.1991	**0.2305**	**0.0880**	0.2273	0.2195	0.3048	0.2758	**0.2684**	**0.0224**	0.2153	0.2107	0.2495	0.0026	0.2386	0.0041	0.2322	0.0042
	PatchTST	0.2268	-	0.2268	-	0.2268	-	0.0423	0.0133	0.2021	0.0887	0.0194	0.0148	0.2282	0.0239	0.2234	0.0193	0.2209	0.0155
Ours		**0.0185**	**0.0182**	0.2181	0.1004	**0.0026**	**0.0052**	**0.0210**	**0.0080**	0.1650	0.1258	**0.0014**	**0.0032**	0.2416	0.0105	0.2353	0.0074	0.2314	0.0050
Ground Truth		0.0367		0.3185		0.0078		0.0290		0.2908		0.0046		0.2521		0.2427		0.2364	

Table G: Results of anomaly detection experiment on SMD dataset. When the data-driven model decides all samples to be abnormal, the F1 score is 0.0806.

Data-driven Model		GMM						ECOD						DeepSVDD					
Metrics		F1-@K	Diff	F1-C	Diff	F1-R	Diff	F1-@K	Diff	F1-C	Diff	F1-R	Diff	F1-@K	Diff	F1-C	Diff	F1-R	Diff
Unsupervised Time-series Anomaly Model	LSTM-P	**0.1240**	**0.0204**	0.1077	0.0833	0.1050	0.0285	0.1468	0.1077	0.2125	0.1037	0.0636	0.0526	0.0920	0.0024	0.0763	0.0018	0.0757	0.0016
	DeepAnT	0.1131	0.0313	0.0979	0.0931	0.0554	0.0211	0.0424	0.0033	0.0637	0.0451	0.0124	0.0014	**0.0928**	**0.0016**	0.0754	0.0027	0.0746	0.0027
	TCN-S2S-P	0.0832	0.0612	0.0811	0.1099	0.0809	0.0044	0.0203	0.0188	0.0612	0.0476	0.0033	0.0077	0.0924	0.0020	0.0766	0.0015	0.756	0.0017
	MTAD-GAT	0.2176	0.0732	0.1948	0.0038	0.1662	0.0897	0.1407	0.1016	0.2046	0.0958	0.0497	0.0387	0.0908	0.0036	0.0743	0.0008	0.0736	0.0037
	GDN	0.1180	0.0264	0.1051	0.0859	0.1031	0.0266	0.1650	0.1259	0.2091	0.1003	0.0726	0.0616	0.0925	0.0019	**0.0777**	**0.0004**	**0.0770**	**0.0003**
	GTA	0.1047	0.0397	0.0845	0.1065	**0.0768**	**0.0003**	0.0006	0.0385	0.1011	0.1077	0.0001	0.0109	0.0901	0.0043	0.0757	0.0042	0.0749	0.0024
Time-series Prediction Model	NLinear	0.1148	0.0295	0.1208	0.0702	0.0540	0.0225	0.0427	0.0036	0.0807	0.0281	0.0126	0.0016	0.0918	0.0026	**0.0785**	**0.0004**	0.0778	0.0005
	DLinear	0.0917	0.0527	0.1062	0.0848	0.0494	0.0271	0.0358	0.0033	0.0660	0.0428	**0.0103**	**0.0007**	0.0917	0.0027	**0.0785**	**0.0004**	0.0778	0.0005
	TimesNet	0.1972	0.0528	**0.1921**	**0.0011**	0.1483	0.0718	0.1225	0.0834	0.2242	0.1154	0.0581	0.0471	0.0915	0.0029	0.0757	0.0024	0.0751	0.0022
	PatchTST	0.1190	0.0254	0.1489	0.0421	0.0699	0.0066	**0.0410**	**0.0019**	0.0757	0.0331	0.0121	0.0011	0.0910	0.0034	0.0773	0.0008	0.0767	0.0006
Ours		0.1996	0.0552	0.2178	0.0268	0.1278	0.0513	0.0463	0.0072	**0.0865**	**0.0223**	0.0131	0.0021	0.0924	0.0020	0.0787	0.0006	0.0780	0.0007
Ground Truth		0.1444		0.1910		0.0765		0.0391		0.1088		0.0110		0.0944		0.0781		0.0773	

D.3 ABLATION STUDIES

For more accurate prediction performance, our forecasting model consists of various factors. Among them, the most important factors are graph structure and separate training processes of continuous and discrete features. Therefore, we investigate the role and effectiveness of each component through ablation studies of prediction performance using the SMD dataset.

D.3.1 GRAPH STRUCTURE.

Table H: Results of anomaly detection experiment on PSM dataset. When the data-driven model decides all samples to be abnormal, the F1 score is 0.4351.

Data-driven Model		GMM						ECOD						DeepSVDD					
Metrics		F1-@K	Diff	F1-C	Diff	F1-R	Diff	F1-@K	Diff	F1-C	Diff	F1-R	Diff	F1-@K	Diff	F1-C	Diff	F1-R	Diff
Unsupervised Time-series Anomaly Model	LSTM-P	0.0034	0.0452	0.0160	0.1128	0.0007	0.0139	0.0016	0.0228	0.0219	0.0581	0.0013	0.0059	0.4571	0.0134	**0.4347**	**0**	0.4304	0.0018
	DeepAnT	0.0	0.0486	0.0	0.1288	0.0	0.0146	0.0087	0.0157	0.0590	0.0210	0.0057	0.0015	0.4572	0.0135	0.4358	0.0011	0.4316	0.0006
	TCN-S2S-P	0.0	0.0486	0.0	0.1288	0.0	0.0146	0.0	0.0244	0.0	0.0800	0.0	0.0072	0.4505	0.0068	0.4307	0.0040	0.4241	0.0081
	MTAD-GAT	0.0	0.0486	0.0055	0.1233	0.0006	0.0140	0.0090	0.0154	0.0434	0.0366	0.0025	0.0047	0.4560	0.0123	0.4308	0.0039	0.4263	0.0059
	GDN	**0.0322**	**0.0164**	**0.0691**	**0.0597**	**0.0163**	**0.0017**	0.0145	0.0099	0.0799	0.0001	0.0080	0.0008	0.4601	0.0164	0.4341	0.0006	0.4262	0.0060
	GTA	0.0	0.0486	0.0	0.1288	0.0	0.0146	0.0005	0.0239	0.0055	0.0745	0.0002	0.0070	0.4631	0.0194	0.4428	0.0081	0.4357	0.0035
Time-series Prediction Model	NLinear	0.0250	0.0236	0.0539	0.0749	0.0015	0.0131	0.0241	0.0003	**0.0800**	**0**	**0.0072**	**0**	**0.4458**	0.0021	0.4332	0.0015	0.4311	0.0011
	DLinear	0.0181	0.0305	0.0540	0.0747	0.0010	0.0136	0.0239	0.0005	**0.0800**	**0**	0.0069	0.0003	0.4459	0.0022	0.4331	0.0016	0.4312	0.0010
	TimesNet	0.0537	0.0051	0.0160	0.1128	0.0036	0.011	0.017	0.0074	0.0642	0.0158	0.0038	0.0034	0.4511	0.0074	0.4369	0.0022	0.4337	0.0015
	PatchTST	0.0244	0.0242	0.0274	0.1014	0.0006	0.0140	0.0241	0.0003	**0.0800**	**0**	0.0070	0.0002	0.4459	0.0022	0.4346	0.0001	**0.4326**	**0.0004**
Ours		0.0244	0.0242	0.0591	0.0697	0.0020	0.0126	**0.0243**	**0.0001**	**0.0800**	**0**	0.0073	0.0001	0.4460	0.0023	0.4350	0.0003	0.4331	0.0009
Ground Truth		0.0486		0.1288		0.0146		0.0244		0.0800		0.0072		0.4437		0.4347		0.4322	

As mentioned in the proposed method, the graph structure provides dependencies between features in the prediction process. As shown in Figure B, anomaly detection benchmark datasets have a correlation between features. Therefore, the graph structure allows our model to capture correlations for each feature that might not be considered by separate training, which leads to a decrease in overall error for features. Figure C
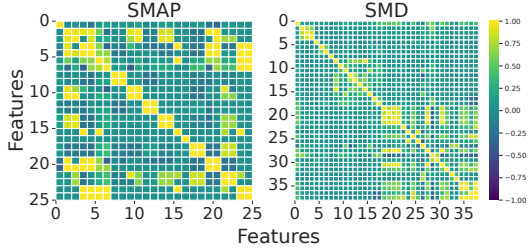

Figure B: Visualization of correlation between each feature. It is closer to a positive (resp. negative) correlation when the color is brighter (resp. darker).

shows that our forecasting model with the graph structure showed lower errors than the model without graph structure in both continuous and discrete features. In other words, it is evident that the graph structure, considering the relationship between each feature, is imperative in anomaly detection datasets with correlations.

D.3.2 SEPARATE TRAINING OF CONTINUOUS AND DISCRETE FEATURES.

We split the two groups of features to predict more accurately the time-series datasets for anomaly detection, which include both continuous and discrete features. In this subsection, we investigated the model without separation training to confirm the effectiveness of separation training. In order to implement the model without separation training, discrete features which consist of one-hot vectors (denoted $\mathbf{O}^t$) and continuous features were concatenated to be used as input to the continuous prediction model using a graph structure. In addition, it was trained with the MSE loss.
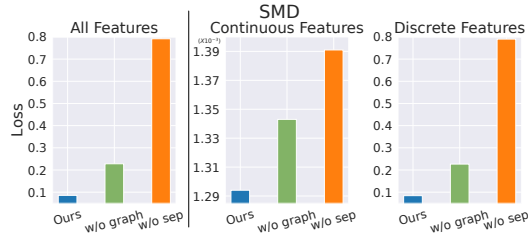

Figure C: Ablation study results for graph structure and separate training of continuous (MSE loss) and discrete (CE loss) features.

As a result, Figure C shows that the model without separation training significantly increases the cross-entropy loss compared to our model. In other words, it shows that separation training is necessary for datasets with continuous and discrete features, particularly in multivariate time-series anomaly detection.