ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10447891

DACR: DISTRIBUTION-AUGMENTED CONTRASTIVE RECONSTRUCTION FOR TIME-SERIES ANOMALY DETECTION

Lixu Wang[1], Shichao Xu[1], Xinyu Du[2], Qi Zhu[1]

[1]Northwestern University, IL, USA
[2]General Motors Global R&D, MI, USA

ABSTRACT

Anomaly detection in time-series data is crucial for identifying faults, failures, threats, and outliers across a range of applications. Recently, deep learning techniques have been applied to this topic, but they often struggle in real-world scenarios that are complex and highly dynamic, e.g., the normal data may consist of multiple distributions, and various types of anomalies may differ from the normal data to different degrees. In this work, to tackle these challenges, we propose *Distribution-Augmented Contrastive Reconstruction* (**DACR**). **DACR** generates extra data disjoint from the normal data distribution to compress the normal data's representation space, and enhances the feature extractor through contrastive learning to better capture the intrinsic semantics from time-series data. Furthermore, **DACR** employs an attention mechanism to model the semantic dependencies among multivariate time-series features, thereby achieving more robust reconstruction for anomaly detection. Extensive experiments conducted on nine benchmark datasets in various anomaly detection scenarios demonstrate the effectiveness of **DACR** in achieving new state-of-the-art time-series anomaly detection.

Index Terms— Anomaly Detection, Time-Series Data

1. INTRODUCTION

System malfunctions and anomalies are unavoidable in many real-world applications across various fields [1]. Accurate anomaly detection is critically important for monitoring and alarming potential faults, threats, and risks in these systems [2]. Recently, data-driven methods have become the mainstream for anomaly detection, among which algorithms based on deep learning perform the best and can be divided into three categories: reconstruction prediction [3, 4], anomaly exposure [5, 6], and self-supervised learning (SSL) [7, 8]. Reconstruction prediction utilizes the difference between the reconstruction outputs for normal and abnormal data. Anomaly exposure synthesizes extra data that is different from the normal data to better model the normal data distribution. SSL leverages auxiliary tasks to help models extract the intrinsic semantics of normal data.

However, anomaly detection scenarios in the real world are often very complex and highly dynamic. For example, normal data may consist of multiple distributions, and various types of anomalies may differ from normal data to different degrees. When faced with these challenging scenarios, the aforementioned deep anomaly detection methods expose a number of shortcomings. For instance, reconstruction prediction [3, 4] performs poorly when the normal data consists of multiple distributions that have different learning difficulties. The performance of anomaly exposure [5, 6] depends on the similarity between ground-truth anomaly data and simulated extra data. SSL [7, 8] performs better when dealing with the above scenarios, but has its own challenges. For example, the feature extractor trained with contrastive learning [9] eventually converges to a uniform hyperspherical space [10] that is not suitable for anomaly detection [11].

To address the aforementioned challenges, we propose a novel method called *Distribution-Augmented Contrastive Reconstruction* (**DACR**), which comprises three stages. Specifically, in the first stage, we train a variational auto-encoder (VAE) to reconstruct normal data. In the second stage, we introduce random noise into the latent space when applying the trained VAE to generate extra data from a different distribution, a process we refer to as *distribution augmentation*. With the extra data, a series of simple feature extractors are trained with contrastive learning, enabling them to extract intrinsic semantics from each univariate time-series feature. In the final stage, **DACR** employs a transformer to model the inter-feature semantic dependency. **This allows DACR to reconstruct time series on the basis of intrinsic semantics rather than overfitting to artificial features that are only specific to the reconstruction task, with such being highly generalizable to more anomalies**. The overall workflow of **DACR** is depicted in Fig. 1. Extensive experiments on nine benchmark datasets, which encompass various anomaly detection scenarios, demonstrate that our methods outperform existing state-of-the-art baseline methods substantially. In summary, the main contributions include:

1. We address practical anomaly detection scenarios where the normal data or the anomalies consist of multiple

We gratefully acknowledge the support from the NSF awards 1834701, 1724341, 2038853, and a grant from General Motors.

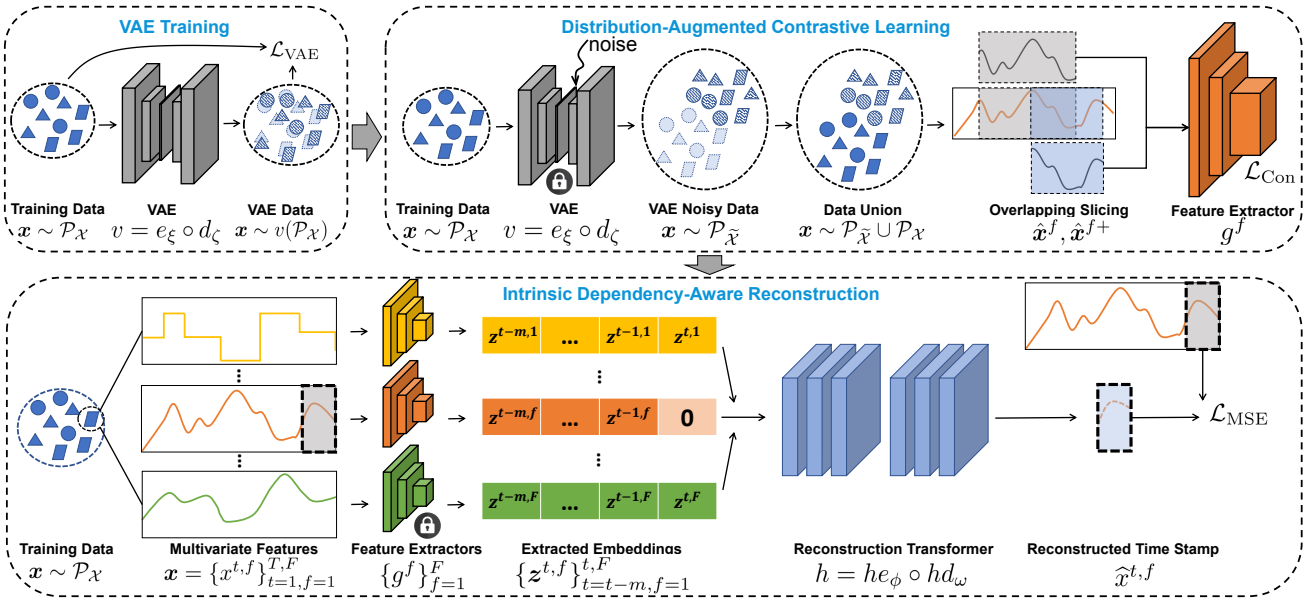

Fig. 1: Overall workflow of *Distribution-Augmented Contrastive Reconstruction* (**DACR**) for time-series anomaly detection.

distributions. The challenge is to model the implicit discrepancies between these normal and anomaly distributions.

2. We develop **DACR**, which combines the strengths of self-supervised learning and attention-based reconstruction prediction. By capturing the intrinsic semantic dependency between multivariate features of time series, **DACR** is significantly more sensitive to potential anomalies.

3. We conduct extensive experiments on nine benchmark datasets, covering scenarios with varying degrees of discrepancies between normal and anomalous data. Experiment results demonstrate that **DACR** consistently and substantially outperforms existing state-of-the-art baselines.

Related Work: For *reconstruction prediction*, anomaly detectors are strongly over-fitted to normal data. Recent studies apply convolution neural networks [12], LSTM [13], and transformers [14] to build such reconstruction models. *Anomaly exposure* usually relies on extra data generation, and by training the model in supervised learning [5, 6, 15] to distinguish extra data from normal data, the distribution of normal data can be better modeled. *SSL-based methods* rely on various auxiliary tasks, such as rotation prediction [16] and data augmentation [11, 7, 17, 8, 18, 19]. While these methods are designed for image data, no study explores whether SSL can work on time-series anomaly detection. Other anomaly detection works include graph neural network [20] and ensemble learning [21].

2. METHODOLOGY

Problem Formulation: Suppose an unlabeled dataset with N time-series samples $\mathcal{X} = \{\boldsymbol{x}_i \| \boldsymbol{x}_i \sim \mathcal{P}_{\mathcal{X}}\}_{i=1}^{N}$ is given, where $\mathcal{P}_{\mathcal{X}}$ is the input feature distribution. Each sample $\boldsymbol{x}_i$ can be regarded as an observation of a matrix-valued random variable with dimensions $T \times F$, where T is the sequence length and F is the feature dimension, i.e., $\boldsymbol{x}_i = \{x_i^{t,f}\}_{t=1,f=1}^{T,F}$. Following standard anomaly detection assumptions [22, 23], dataset $\mathcal{X}$ is considered as being full of normal data. The objective is to learn a model based on $\mathcal{X}$ to accurately infer each time stamp of each testing sample as either normal or anomaly.

2.1. Distribution-Augmented Contrastive Learning

Time Series Contrastive Learning

Different from contrastive learning on visual data [11, 7], it is not yet straightforward to find suitable augmentation techniques for producing positive pairs of time-series data. Inspired by TS2Vec [17], we extend overlapping slicing to our problem. More specifically, overlapping slicing here means randomly cutting out two fragments $(\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{x}}_i^+)$ from a given time-series instance $\boldsymbol{x}_i$ while ensuring that there is an overlapping part between them, i.e., $\hat{\boldsymbol{x}}_i = \{x_i^t\}_{t=a}^{b}$, $\hat{\boldsymbol{x}}_i^+ = \{x_i^t\}_{t=c}^{d}$, where $1 \le a < c < b < d \le T$. The instance-wise contrastive comparison between positive pairs ($\hat{\boldsymbol{z}}_i^t$ & $\hat{\boldsymbol{z}}_i^{t+}$) is constructed as comparing representations of fragments from the same data instance, while that of negative pairs ($\hat{\boldsymbol{z}}_i^t$ & $\hat{\boldsymbol{z}}_j^{t+}$, and $\hat{\boldsymbol{z}}_i^t$ & $\hat{\boldsymbol{z}}_j^t$ where $i \neq j$) is constructed as comparing representations of fragments from different data instances.

$$\mathcal{L}_{\text{In},i}^t = -\log \frac{\exp(\hat{\boldsymbol{z}}_i^t \cdot \hat{\boldsymbol{z}}_i^{t+})}{\sum_j \left(\exp(\hat{\boldsymbol{z}}_i^t \cdot \hat{\boldsymbol{z}}_j^{t+}) + \mathbf{1}_{i \neq j} \exp(\hat{\boldsymbol{z}}_i^t \cdot \hat{\boldsymbol{z}}_j^t)\right)}. \quad (1)$$

Here the range of j is $[1, N_B]$ if the batch size is N_B. $\mathbf{1}_{(\cdot)}$ is an indicator function so that if the subscript condition is true, $\mathbf{1}_{\text{True}} = 1$, otherwise, $\mathbf{1}_{\text{False}} = 0$. Considering the temporal consistency in time-series data, we also need to conduct temporal contrastive comparisons. However, we only consider the comparison in the overlapping part $t \in [c, b]$ of the augmented fragments $\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{x}}_i^+$, instead of considering the entire

sequence as in TS2Vec [17]. The positive pairs ($\hat{\boldsymbol{z}}_i^t$ & $\hat{\boldsymbol{z}}_i^{t+}$) of temporal CL are constructed as the representations of augmented fragments at the same time stamp, while the negative pairs ($\hat{\boldsymbol{z}}_i^t$ & $\hat{\boldsymbol{z}}_i^{t'+}$, and $\hat{\boldsymbol{z}}_i^t$ & $\hat{\boldsymbol{z}}_i^{t'}$ where $t \neq t'$) are the representations of augmented fragments at different time stamps.

$$\mathcal{L}_{\mathrm{Te},i}^t = -\log \frac{\exp(\hat{\boldsymbol{z}}_i^t \cdot \hat{\boldsymbol{z}}_i^{t+})}{\sum_{t'} \left(\exp(\hat{\boldsymbol{z}}_i^t \cdot \hat{\boldsymbol{z}}_i^{t'+}) + \mathbf{1}_{t \neq t'} \exp(\hat{\boldsymbol{z}}_i^t \cdot \hat{\boldsymbol{z}}_i^{t'})\right)}. \tag{2}$$

Finally, for a data batch in the mini-batch training, we have an overall contrastive loss as:

$$\mathcal{L}_{\mathrm{Con}} = \frac{1}{N_B(b-c)} \sum_{i=1}^{N_B} \sum_{t=c}^{b} \left(\mathcal{L}_{\mathrm{In},i}^t + \mathcal{L}_{\mathrm{Te},i}^t\right). \tag{3}$$

VAE-Based Distribution Augmentation

For standard CL as shown in Eq. (1), it has been shown that the optimal solution shapes like a perfect uniform distribution for all training data in the representation space [10]. In such cases, it is difficult to distinguish outliers from their proximal inliers (training data). In this work, we generate extra data from a disjoint distribution to the normal data to occupy a certain space of the final uniform distribution. Through this *distribution augmentation* process, the uniformity of the original normal data is greatly reduced. To make the extra data diverse enough [24], we achieve that by introducing random noise to a variational auto-encoder (VAE) $v = e_\xi \circ d_\zeta$. Specifically, we first train v with the task of reconstructing the input data. The training loss is Mean Square Error (MSE) and Kullback–Leibler Divergence, as shown below:

$$\mathcal{L}_{\mathrm{VAE}} = \mathbb{E}_{\boldsymbol{x}_i \sim \mathcal{P}_\mathcal{X}} \|\boldsymbol{x}_i - v(\boldsymbol{x}_i)\|_2 + \mathcal{D}_{\mathrm{KL}}(\mathcal{P}_\mathcal{X} \| v(\mathcal{P}_\mathcal{X})). \tag{4}$$

After the VAE training, we can generate extra data from disjoint distributions with various levels of discrepancies by injecting Gaussian noise into the low-dimensional latent space:

$$e_\xi(\boldsymbol{x}_i)' = \alpha \odot e_\xi(\boldsymbol{x}_i) + \beta, \tag{5}$$

where α and β are vectors of the same dimensions with $e_\xi(\boldsymbol{x}_i)$, and we set $\alpha \sim \mathcal{N}(1, 0.1 \cdot \mathbf{I})$ and $\beta \sim \mathcal{N}(0, 0.1 \cdot \mathbf{I})$, where $\mathbf{I}$ is the unit matrix. We can use VAE decoder d_ζ to decode $e_\xi(\boldsymbol{x}_i)'$ into the input space, and then regard the decoded data as $\widetilde{\boldsymbol{x}}_i$. With the united dataset $\mathcal{X} \cup \widetilde{\mathcal{X}}$, we train a dedicated feature extractor g^f for each univariate feature f with the time-series contrastive loss (Eq. (3)).

2.2. Intrinsic Dependency-Aware Reconstruction

As shown in Fig. 1, after the **DACL** stage, we can obtain a feature extractor g^f for each feature dimension f. In the third stage of our method, with Intrinsic Dependency-Aware Reconstruction (**IDAR**) training, these feature extractors are all frozen, and a transformer model h is incorporated to take the embedding vectors $\boldsymbol{z}$ produced by feature extractors as input and tries to reconstruct the time-series instances.

Specifically, for example, suppose that our task is to reconstruct the t-th time stamp of the f-th feature dimension of the i-th time-series instance. The input matrix I of h is

$$\left[[\boldsymbol{z}^{t-m,1}, ..., \boldsymbol{z}^{t,1}], ..., [\boldsymbol{z}^{t-m,f}, ..., \boldsymbol{z}^{t-1,f}, \mathbf{0}], ..., [\boldsymbol{z}^{t-m,F}, ..., \boldsymbol{z}^{t,F}]\right], \tag{6}$$

where m is a hyper-parameter that controls how long the model can observe in history ($m = 20$; please see Section 3.3 for the sensitivity analysis). Note that different from any autoregressive forecasting model, in addition to feeding historical time stamps of all feature dimensions, we also feed embeddings of the t-th stamp of all dimensions except for the f-th. The additional input can help the model better capture the inter-feature dependency of a shorter time period, improving the model's sensitivity to the anomalies. The used transformer architecture consists of an encoder he_ϕ and a decoder hd_ω. To train he_ϕ and hd_ω, we leverage the MSE loss as follows:

$$\mathcal{L}_{\mathrm{MSE}} = \frac{1}{N_B F(T-m)} \sum_{i=1}^{N_B} \sum_{t=m+1}^{T} \sum_{f=1}^{F} \|\widehat{x}_i^{t,f} - x_i^{t,f}\|_2^2. \tag{7}$$

Anomaly Scoring. We also design an anomaly-scoring mechanism to coordinate the usage of **DACR**, which picks the maximum increase percentage among all feature dimensions at the same time stamp compared to the maximum MSE error in the training data as the anomaly score:

$$S_i^t = \max_f \left\{ \frac{\|\widehat{x}_i^{t,f} - x_i^{t,f}\|_2^2 - \mathrm{Err}^f}{\mathrm{Err}^f} \times 100\% \right\}, \tag{8}$$

where Err^f is the maximum MSE error of the f-th feature dimension among all training data $\mathcal{X}$. Finally, for a particular timestamp t of a time-series instance $\boldsymbol{x}_i$, if its anomaly score S_i^f is larger than zero, it is labeled as an anomaly.

3. EXPERIMENTAL RESULTS

3.1. Experimental Settings

Our code is implemented in PyTorch. All experiments are conducted on Ubuntu 18.04 LTS with NVIDIA TITAN RTX.

Table 1: Time-Series Anomaly Detection Dataset Information (SAD). N_{sample} – sample quantity, F – dimension number, N_C – class number, and T – sequence length.

Dataset	SAD[8]	CT	RS	PCoE[25]	NAT	ASD[26]	SWaT	WaQ	SMD
N_{sample}	8800	2858	300	7565	360	8528	0.48M	0.14M	28703
F	13	3	6	3	24	19	51	11	38
N_C	10	20	4	4	6	-	-	-	-
T	50	182	30	256	51	100	100	100	100

Dataset Settings. Two scenarios called Explicit Anomaly Detection (EAD) and Implicit Anomaly Detection (IAD) are considered. EAD refers to the setting where there is an explicit discrepancy between normal and abnormal data. We build the EAD by randomly selecting n classes as the normal data ($1 < \mathrm{n} < N_C$) and viewing the remaining classes as anomalies [8]. IAD refers to the setting where there is no explicit discrepancy, which is more challenging, since all normal data may have corresponding near out-of-distribution

Table 2: Performance comparison between **DACR** and baselines in the settings of both EAD (n=N_C-1) and IAD. AUC$_{\pm\text{standard deviation}}$ is used to evaluate the performance. **DACR** significantly outperforms the second-best by 2.8-8.2% on EAD, and 0.4-3.3% on IAD.

Baseline	EAD					IAD			
	SAD	CT	RS	PCoE	NAT	ASD	SWaT	WaQ	SMD
DROCC	58.8$_{\pm0.5}$	57.6$_{\pm1.5}$	60.9$_{\pm0.2}$	69.7$_{\pm1.1}$	60.7$_{\pm1.6}$	69.3$_{\pm1.9}$	83.3$_{\pm0.9}$	68.9$_{\pm3.4}$	86.6$_{\pm0.9}$
TS2Vec	62.7$_{\pm0.6}$	62.4$_{\pm1.0}$	67.6$_{\pm0.6}$	72.4$_{\pm1.5}$	66.6$_{\pm1.8}$	78.9$_{\pm2.3}$	86.0$_{\pm1.8}$	67.8$_{\pm3.5}$	89.9$_{\pm0.9}$
DROC	63.5$_{\pm0.7}$	63.0$_{\pm0.7}$	68.9$_{\pm0.9}$	75.0$_{\pm1.1}$	69.2$_{\pm1.3}$	75.1$_{\pm1.5}$	84.9$_{\pm1.1}$	68.4$_{\pm2.2}$	86.6$_{\pm0.8}$
MSC	55.7$_{\pm2.0}$	58.0$_{\pm1.5}$	61.4$_{\pm1.0}$	65.0$_{\pm0.7}$	61.5$_{\pm0.9}$	80.0$_{\pm2.8}$	85.4$_{\pm0.9}$	70.6$_{\pm0.8}$	90.2$_{\pm1.3}$
NTL	**85.1**$_{\pm0.3}$	**87.4**$_{\pm0.2}$	**80.0**$_{\pm0.4}$	75.5$_{\pm1.1}$	**74.8**$_{\pm0.9}$	59.2$_{\pm4.5}$	85.0$_{\pm2.6}$	61.6$_{\pm9.1}$	74.6$_{\pm6.7}$
GDN	74.9$_{\pm2.1}$	66.4$_{\pm0.7}$	69.6$_{\pm0.9}$	73.8$_{\pm2.5}$	71.1$_{\pm1.3}$	77.9$_{\pm4.2}$	88.5$_{\pm3.6}$	65.9$_{\pm4.3}$	95.9$_{\pm1.6}$
TranAD	64.4$_{\pm1.1}$	61.3$_{\pm0.9}$	70.9$_{\pm0.6}$	72.7$_{\pm0.5}$	66.0$_{\pm1.0}$	91.5$_{\pm1.8}$	81.0$_{\pm0.7}$	**72.9**$_{\pm1.7}$	66.2$_{\pm0.3}$
COUTA	65.0$_{\pm1.1}$	65.5$_{\pm0.8}$	72.2$_{\pm0.2}$	75.0$_{\pm0.9}$	68.0$_{\pm1.1}$	**95.5**$_{\pm3.0}$	**90.0**$_{\pm1.7}$	71.4$_{\pm0.6}$	**98.4**$_{\pm1.5}$
UMS	68.0$_{\pm3.0}$	71.5$_{\pm0.6}$	75.2$_{\pm0.9}$	77.3$_{\pm1.4}$	69.7$_{\pm2.1}$	91.0$_{\pm2.5}$	86.6$_{\pm2.0}$	69.9$_{\pm1.6}$	96.5$_{\pm1.2}$
DACR-ab1	68.3$_{\pm3.3}$	82.0$_{\pm0.9}$	75.9$_{\pm0.5}$	**78.5**$_{\pm0.9}$	69.8$_{\pm2.0}$	82.0$_{\pm1.1}$	87.1$_{\pm0.9}$	68.2$_{\pm1.3}$	87.0$_{\pm2.2}$
DACR-ab2	75.0$_{\pm2.2}$	66.9$_{\pm3.8}$	71.0$_{\pm0.7}$	74.5$_{\pm1.9}$	75.1$_{\pm1.0}$	88.0$_{\pm2.5}$	86.4$_{\pm0.8}$	72.0$_{\pm1.9}$	94.5$_{\pm0.6}$
DACR	**90.7**$_{\pm0.3}$	**91.9**$_{\pm1.5}$	**88.2**$_{\pm3.2}$	**81.3**$_{\pm0.7}$	**81.5**$_{\pm1.6}$	**96.2**$_{\pm2.1}$	**93.3**$_{\pm0.9}$	**75.9**$_{\pm0.9}$	**98.8**$_{\pm1.0}$

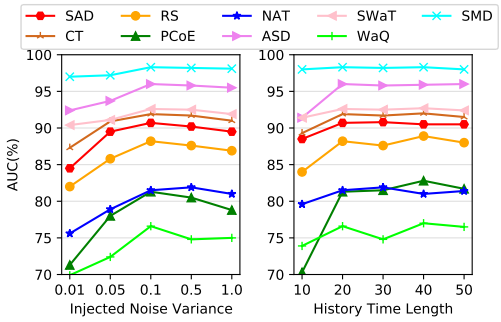

Fig. 2: Sensitivity analysis of various VAE injected noise degrees and different history time lengths of reconstruction transformer input.

data. We follow COUTA [26] to build the IAD setting. Please refer to Table 1 for dataset details.

Implementation Details. The batch size is set as 8. The learning rate of the Adam optimizer for training the feature extractor is 0.001, while that for VAE is 0.0001. For the EAD datasets, the default number of training iterations for the feature extractor is 200. For the IAD datasets, given the larger data size, we train 1000 iterations. We follow NTL [8] to conduct EAD based on the entire sequence of time-series instances and follow COUTA [26] to adopt a window sliding mode for IAD. The backbone architecture of feature extractors is a dilated CNN [17], and the VAE consists of 4 LSTM layers. We repeat our experiments 3 times with different seeds and report the average value and standard deviation for AUC.

Baseline Methods for Comparison. We compare **DACR** with DROCC [6], TS2Vec [17], DROC [11], MSC [18], NTL [8], GDN [23], TranAD [14], COUTA [26], UMS [21].

3.2. Performance Comparison

Table 2 presents experiment results of both EAD and IAD. From the results, we can see that **DACR achieves the best performance for all cases in EAD and IAD, clearly outperforming baselines**. In addition, we also carry out experiments in setting different normal class numbers for EAD, and the results of the SAD dataset are shown in Fig. 3. The figure shows that **DACR** always performs the best, demonstrating its effectiveness under challenging scenarios where the normal and abnormal data consist of multiple different distributions. The results on other datasets show similar trends.

3.3. Ablation Study and Sensitivity Analysis

We conducted an ablation study by modifying **DACR** in two ways: 1) removing the VAE-based Distribution Augmentation (**DACR**-ab1), or 2) replacing the **DACL** feature extractor with that trained by GDN [23] (**DACR**-ab2). The results in Table 2 show that the performance of **DACR**-ab1 and **DACR**-ab2 are significantly worse than **DACR**, showing that all modules in **DACR** are essential and complement each other well. We

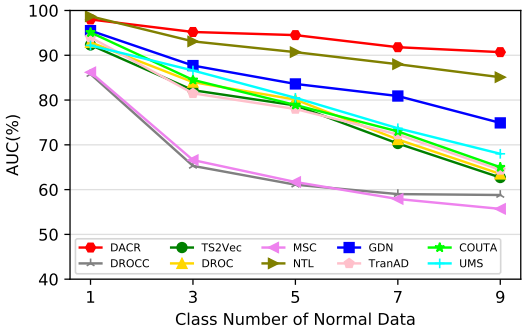

Fig. 3: Performance comparison in EAD with different normal class numbers on the SAD dataset.

conducted a sensitivity analysis on the noise degree (α, β in Eq. (5)) with different variances (0.01, 0.05, 0.1, 0.5, 1.0). According to Fig. 2, we can observe that our method is not very sensitive to noise degrees larger than 0.05. The performance at 0.01 is poor because VAE cannot generate sufficiently diverse data at that moment. We also conducted a sensitivity analysis on the history time length of the transformer input, setting it from 10 to 50 with a stride of 10. Fig. 2 shows that **DACR** performs stably when the length exceeds 20.

4. CONCLUSION

We present *Distribution-Augmented Contrastive Reconstruction* (**DACR**) for time-series anomaly detection. **DACR** leverages a VAE to conduct distribution augmentation, which helps extract intrinsic semantics from univariate time-series features through contrastive learning. Then **DACR** applies the attention mechanism to model the semantic dependency between multivariate features and achieve reconstruction-based anomaly detection. Extensive experiments on nine benchmark datasets in various scenarios demonstrate that **DACR** achieves new state-of-the-art performance.

5. REFERENCES

[1] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A Lozano, "A review on outlier/anomaly detection in time series data," *ACM Computing Surveys*, 2021.

[2] Jining Chen, Weitu Chong, Siyu Yu, Zhun Xu, Chaohong Tan, and Ningjiang Chen, "Tcn-based lightweight log anomaly detection in cloud-edge collaborative environment," in *Tenth International Conference on Advanced Cloud and Big Data*, 2022.

[3] 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 *ACM KDD*, 2019.

[4] Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei, "Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding," in *ACM KDD*, 2021.

[5] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich, "Deep anomaly detection with outlier exposure," in *ICLR*, 2018.

[6] Sachin Goyal, Aditi Raghunathan, Moksh Jain, Harsha Vardhan Simhadri, and Prateek Jain, "Drocc: Deep robust one-class classification," in *ICML*, 2020.

[7] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin, "Csi: Novelty detection via contrastive learning on distributionally shifted instances," *NeurIPS*, 2020.

[8] Chen Qiu, Timo Pfrommer, Marius Kloft, Stephan Mandt, and Maja Rudolph, "Neural transformation learning for deep anomaly detection beyond images," in *ICML*, 2021.

[9] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio, "Learning deep representations by mutual information estimation and maximization," in *ICLR*, 2018.

[10] Tongzhou Wang and Phillip Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *ICML*, 2020.

[11] Kihyuk Sohn, Chun-Liang Li, Jinsung Yoon, Minho Jin, and Tomas Pfister, "Learning and evaluating representations for deep one-class classification," in *ICLR*, 2020.

[12] Kushal Chauhan, Pradeep Shenoy, Manish Gupta, Devarajan Sridharan, et al., "Robust outlier detection by de-biasing vae likelihoods," in *CVPR*, 2022.

[13] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv:1607.00148*, 2016.

[14] Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings, "Tranad: Deep transformer networks for anomaly detection in multivariate time series data," *arXiv:2201.07284*, 2022.

[15] Lixu Wang, Shichao Xu, Ruiqi Xu, Xiao Wang, and Qi Zhu, "Non-transferable learning: A new approach for model ownership verification and applicability authorization," in *ICLR*, 2021.

[16] Nikos Komodakis and Spyros Gidaris, "Unsupervised representation learning by predicting image rotations," in *ICLR*, 2018.

[17] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu, "Ts2vec: Towards universal representation of time series," in *AAAI*, 2022.

[18] Tal Reiss and Yedid Hoshen, "Mean-shifted contrastive loss for anomaly detection," *arXiv:2106.03844*, 2021.

[19] Payal Mohapatra, Bashima Islam, Md Tamzeed Islam, Ruochen Jiao, and Qi Zhu, "Efficient stuttering event detection using siamese networks," in *ICASSP*, 2023.

[20] Ailin Deng and Bryan Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *AAAI*, 2021.

[21] Mononito Goswami, Cristian Ignacio Challu, Laurent Callot, Lenon Minorics, and Andrey Kan, "Unsupervised model selection for time series anomaly detection," in *ICLR*, 2023.

[22] Nikolay Laptev, Saeed Amizadeh, and Ian Flint, "Generic and scalable framework for automated time-series anomaly detection," in *ACM KDD*, 2015.

[23] Ailin Deng and Bryan Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *AAAI*, 2021.

[24] Chenxi Liu, Lixu Wang, Lingjuan Lyu, Chen Sun, Xiao Wang, and Qi Zhu, "Deja vu: Continual model generalization for unseen domains," in *ICLR*, 2022.

[25] B Saha and K Goebel, "Nasa ames prognostics data repository," *NASA Ames, Moffett Field, CA, USA*, 2007.

[26] Hongzuo Xu, Yijie Wang, Songlei Jian, Qing Liao, Yongjun Wang, and Guansong Pang, "Calibrated one-class classification for unsupervised time series anomaly detection," *arXiv:2207.12201*, 2022.