Multivariate Time Series Anomaly Detection with Few Positive Samples

Feng Xue
GE Research
Niskayuna, NY, USA
xue@ge.com

Weizhong Yan
GE Research
Niskayuna, NY, USA
yan@ge.com

***Abstract*—Given the scarcity of anomalies in real-world applications, the majority of literature has been focusing on modeling normality. The learned representations enable anomaly detection as the normality model is trained to capture certain key underlying data regularities under normal circumstances. In practical settings, particularly industrial time series anomaly detection, we often encounter situations where a large amount of normal operation data is available along with a small number of anomaly events collected over time. This practical situation calls for methodologies to leverage these small number of anomaly events to create a better anomaly detector. In this paper, we introduce two methodologies to address the needs of this practical situation and compared them with recently developed state of the art techniques. Our proposed methods anchor on representative learning of normal operation with autoregressive (AR) model along with loss components to encourage representations that separate normal versus few positive examples. We applied the proposed methods to two industrial anomaly detection datasets and demonstrated effective performance in comparison with approaches from literature. Our study also points out additional challenges with adopting such methods in practical applications.**

***Index Terms*—multivariate time series, anomaly detection, neural networks, representation learning, few labels**

I. Introduction

Anomaly detection has been a widely researched topic in machine learning and is of significant importance in many areas such as fraud detection, cyber security, and complex system health monitoring [1]. It still remains as an active and challenging research area. In recent years, deep learning has been widely used for anomaly detection [2]. Anomaly detection has been applied to many different applications, this paper mainly focuses on its application to industrial multivariate time series data. With the increasing number of sensors as well as cost-effective data transmission and storage solutions, industrial systems, such as power plant, wind turbines, engines etc., produce large amounts of time series data during their regular operations. It is important to monitor these systems to spot abnormal behaviors, which, if not detected earlier, could have significant reliability consequences.

Anomalies, also referred to as outliers, are defined as observations which deviate so much from the majority of all the observations. In the context of industrial time series data, the systems are usually operated based on what they are designed for. Under normal operating conditions (NOCs), the system measurements are a partial capture of the dynamic state governed by operation profiles, first principles, and the underlying control logic. Hence, the contextual information is an important factor when developing anomaly detection techniques. These anomalies are often referred as contextual anomalies, one of anomaly types as categorized in [3].

For a typical anomaly detection application, there is usually a lack of abnormal data. Most of the data collected for anomaly detection model development is under NOCs. The core idea of developing anomaly detection is to learn the spatial (cross multiple system measurement and commands) and temporal relationships under normal operations. In an abnormal situation, such relationships will not adhere to the learned representation, resulting in deviations from the normal operating patterns. The higher a deviation is, the more likely it is an anomaly. This type of anomaly detection methods, i.e., to learn the system normal behabior using data under NOCs only, is often referred as semi-supervised anomaly detection (SSAD) in the literature.

In a real-world application setting, some labeled anomalies (faults or events) are sometimes available (albeit the number of such anomalies is usually very small), in addition to the abundant availability of normal data. Leveraging such limited anomaly data in the process of building anomaly detection algorithms to improve their detection performance, mainly reducing false positives, has became an interesting research question. In recent years, a few research efforts have been made towards answering this very research question. For example, Pang et al. [4] proposed the Pairwise Relation prediction-based ordinal regression Network, which simultaneously learns pairwise relations and anomaly scores by training an end-to-end ordinal regression neural network.

In this paper, we explore the use of loss functions based on few anomalous samples to regulate learning normality representation, hence increase the model's effectiveness of detecting anomalies. The contributions of this paper are as follows:

- We described a jointly-learning approach that incorporates both normality representation learning and regularization from few anomalous samples.
- We demonstrated advantages of such learning strategy over state of the art methodology on two public datasets.
- We further examined that the behavior of applying such learning approach to situations where available anoma-

lous samples are not representative of future anomalies, i.e. a domain shift, and revealed the limitation of the current approaches towards real-world applications.

II. Related Work

A. Normal Time Series Data Modeling

In the process control and model-based fault detection community, a number of data driven approaches have been used for anomaly detection of industrial time series data, for example, Principal Component Analysis (PCA), Dynamic PCA [5], subspace aided approach [6]. These traditional approaches take into consideration of multivariate linear relationship, and to some extent of temporal dependence in the case of Dynamic PCA or subspace aided approach. Another often used approach is one-class SVM such as in [7]. In recent years, deep learning has become an active research area for multivariate anomaly detection [8], [8]–[13].

A number of approaches have been proposed in the literature for modeling normal time series data. One class is to learn an autoregressive (AR) model, in which the past observations are used to predict the future. A Recurrent Neural Network (RNN) such as a LSTM [14] is usually used for such tasks, although recent research work [15], [16] demonstrated that a causal convolutional network might be a better alternative in term of effectiveness and training efficiency. Another class is the encoder-decoder based approach. For example, [8] is a direct application of sequence-to-sequence (seq2seq) modeling as in [17] with reconstruction errors as the loss function for time series representation learning. Alternatively in [18], a variational autoencoder (VAE) has been employed along with an LSTM to reconstruct each input at each time step of the series. [10] proposed GGM-VAE, a Gaussian Mixture model that is used to represent the latent space with GRU as an encoder. GGM-VAE aims to better deal with multimodal sensory data, in contrast to a typical single Gaussian latent space representation. In our problem setting, we have both normal data and few faulty samples. The goal is to find a better normality representation by leveraging both for enhancing anomaly detection effectiveness.

B. Semi-Supervised Anomaly Detection

As discussed in [19], traditional semi-supervised anomaly detection (SSAD) methods involve training models using labeled normal samples only and ignoring limited labeled anomaly samples available, which tends to have a high false positive rate. Recently developed SSAD methods focus on improving anomaly detection performance, i.e., reducing false positives, by leveraging the labeled anomaly data (often very small). These new SSAD methods differ primarily in the strategies used for leveraging the labeled anomaly samples, depending on data availability scenarios and the problem settings.

Assuming both labeled and unlabeled samples are available, Ruff, et al. [20] proposed DeepSAD, an end-to-end methodology for general SSAD, which is an extension of DeepSVDD [21]. DeepSVDD is a neural network version of support vector data description (SVDD) with a specially defined objective function such that it can learn feature representation and the smallest hypersphere together. While DeepSVDD works on unlabeled data and assumes most of the unlabeled data are normal, DeepSAD takes advantage of labeled samples in addition to unlabeled data. Essentially it includes an additional term in the DeepSVDD's cost function to force the network to map normal samples closer to the hypersphere center and the known anomalies further from it.

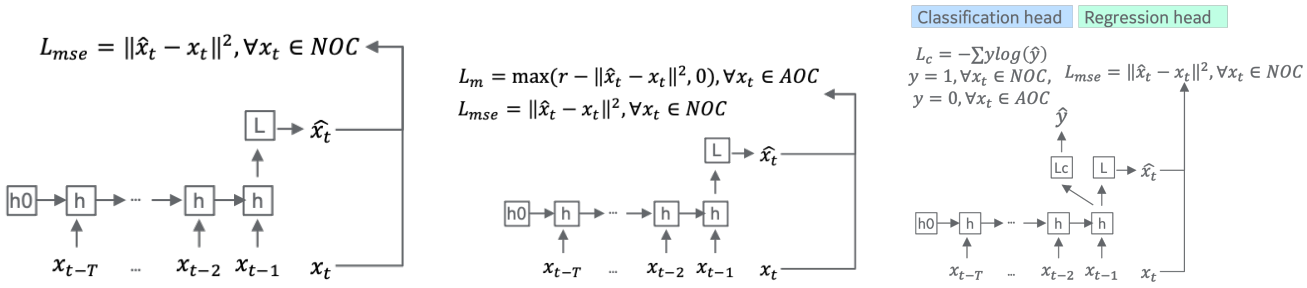
Considering an application scenario where only a small labeled anomaly data and a large number of unlabeled data are available, Pang et al. [4] proposed the Pairwise Relation prediction-based ordinal regression Network, which simultaneously learns pairwise relations and anomaly scores by training an end-to-end ordinal regression neural network. They termed their anomaly detection problem as "weakly-supervised anomaly detection". By addressing the same application scenarios, [22] proposed an anomaly detection with partially observed anomalies. More recently, this type of anomaly detection problem was tackled in [23] where a SSAD method called ConNet was proposed. It utilized a few labeled anomalies as a prior knowledge and trained an anomaly scoring network with the concentration loss, an improved version of the contrastive loss.

Our work in this paper is also related to leveraging few labeled anomaly samples for improving anomaly detection performance. However, our work differs significantly from the aforementioned studies in that we are addressing anomaly detection problem in an industrial setting where the data is primarily time-series sensor measurements. Similarly to DeepSAD, our proposed method uses additional loss term derived from labeled anomalies in comparison to normal data, but our normal representative learning leverage time series data temporal property instead of a mere projection to a hypersphere as in DeepSAD. In our paper, we compared our approach with DeepSAD on two benchmark datasets.

III. Problem Formulation and Approach

In general, anomaly detection with neural networks can be categorized into three paradigms: deep learning for feature extraction, learning feature representation of normality, and end-to-end anomaly score learning [24]. In a typical industrial setting, normal operation data are usually abundant while the number of faulty cases is often very small if there are any. Therefore, the second category, which models the normality, is often a preferable approach. This paradigm learns a representation of data by using an objective function that is not directly aligned to an anomaly score.

When it comes to modeling industrial time series data, an autoregressive (AR) approach is often used for modeling such time series given its connection to dynamic systems. Let $\mathbf{x_t} \in \mathbb{R}^n$ be the multivariate sample of dimension n at time t, and denote the j-th dimension at time t as x_t^j (i.e., $\mathbf{x_t} = [x_t^1, x_t^2, ..., x_t^n]$). The AR approach is trying to estimate $\mathbf{x_t}$ from all observations up to time $t-1$, noted as $\mathbf{x_{t-}}$, which indicates all time series data before t. That is to say, the model



(a) AR model (b) AR model with margin loss (c) AR model with auxiliary classification loss

Fig. 1: Comparing traditional AR model with the proposed approach for handling few fault samples. 1a illustrates a traditional AR model with a recurrent network (such as LSTM) as backbone; 1b illustrates the margin loss approach with the same backbone; while 1c illustrated the auxiliary classification task approach with the same backbone

is trying to learn the relationship $\hat{\mathbf{x}}_\mathbf{t} = f(\mathbf{x}_{\mathbf{t}-})$, such that the prediction $\hat{\mathbf{x}}_\mathbf{t}$ is as close to the real observation $\mathbf{x}_\mathbf{t}$ as possible measured by some distance metric $d(\hat{\mathbf{x}}_\mathbf{t}, \mathbf{x}_\mathbf{t})$. In practice, a window of length T is often used as the inputs to the model instead of all samples prior to time t. This window length can be adjusted to different applications and datasets. The distance metric d can be chosen as the Euclidean distance, corresponding to a mean squared error (MSE) loss during training as in Equation 1:

$$L_{mse} = \|\hat{\mathbf{x}}_\mathbf{t} - \mathbf{x}_\mathbf{t}\|^2 \tag{1}$$

This distance metric measures the deviation of a sample from what it should have been under normal operating conditions. Therefore, a sample whose deviation is above a defined threshold can be regarded as anomalous. Under the assumption of that majority of the time series data gathered in industrial settings is under NOC, this training mechanism essentially tries to learn the dynamic representation of the underlying system. In the case that there are a number of known anomalies, we have majority time series $\mathbf{x}_\mathbf{t} \in \mathcal{N}$ from NOC, while minority time series $\mathbf{x}_\mathbf{t} \in \mathcal{A}$ from Anomalous Operation Conditions (AOC). In order to leverage these samples from AOC, we formulated two approaches: 1) MSE margin loss; 2) auxiliary classification task. The proposed approaches are illustrated in Figure 1.

A. MSE Margin Loss

In this formulation, we want to encourage network to not only reduce the MSE loss L_{mse} in Equation 1 for samples from NOC, but also produce a higher MSE for samples from AOC. To that end, we introduce a margin loss as following:

$$L_m = \max(r - \|\hat{\mathbf{x}}_\mathbf{t} - \mathbf{x}_\mathbf{t}\|^2, 0), \forall x_t \in \mathcal{A} \tag{2}$$

where r is p^{th} percentile of MSEs from the NOC samples. In practice, r is the exponential moving average of p^{th} percentile of MSEs from NOC samples in each batch during training. Here the loss term penalizes the AOC samples with MSE less than the margin r. The overall loss is:

$$L = L_{mse} + \alpha L_m \tag{3}$$

where α is a hyper parameter.

B. Auxiliary Classification Task

In this formulation, we have an auxiliary classification task in addition to the main AR task. In this case, the input $\mathbf{x}_{\mathbf{t}-}$ is transformed in to hidden state via $\mathbf{z} = h(\mathbf{x}_{\mathbf{t}-}, \mathbf{W}_\mathbf{h})$ (the last hidden state as illustrated in 1c). From here, one branch of the network aims to produce AR output, i.e. $\hat{\mathbf{x}}_\mathbf{t} = g(\mathbf{z}, \mathbf{W}_\mathbf{g})$; while the other branch of the network aims to map the hidden state to a binary class output $\hat{\mathbf{y}} = o(\mathbf{z}, \mathbf{W}_\mathbf{o})$. Hence, a cross-entropy loss can be used to encourage the network to distinguish NOC samples versus AOC samples.

$$L_c = \sum_{i=0}^{1} (y_i log(\hat{y}_i)) \tag{4}$$

where y_i is the i-th component of the one-hot vector $\mathbf{y}$ of the true class label $(0, 1)$ representing normal or anomalous samples, while $\hat{\mathbf{y}}$ is the estimated vector. Similarly, the overall loss is:

$$L = L_{mse} + \alpha L_c \tag{5}$$

where α is a hyper parameter.

IV. Experimental Results

A. Datasets

1) TEP Dataset: [1] The Tennessee Eastman process (TEP) is an industrial benchmark by the Eastman Chemical Company for process monitoring and control studies [25]. It models a real industrial process computationally and is widely studied for anomaly detection algorithms [26], [27].

The TEP is comprised of 4 reactants, 2 products, 1 by-product and 1 inert components denoted as A-H. These components undergo a chemical process enabled by 5 major units: a reactor where the reaction happens for the gas feed components (A, C, D and E) into liquid products (G and H), a condenser that cools down the gas stream coming out of the reactor, a separator that separates gas and liquid components from the cooled product stream, a compressor that feeds the

[1]TEP dataset can be downloaded at https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/6C3JR1.

gas stream back into the reactor and a stripper that strips the two products from any unreacted feed components.

The TEP dataset [28] contains 52 variables in total, 41 of which are sensor measurements (XMEAS(1) - XMEAS(41)) and 11 are manipulated variables ((XMV(1) - (XMV11)). Therefore, the multivariate samples $\mathbf{x_t}$ have a dimension of 52, or $n = 52$. The dataset has separate training and testing files, both of which contain a set of "fault-free" and "faulty" files. Each file contains single simulation run of the chemical process. The "fault-free" runs correspond to the processes under normal operating conditions (NOC) while the "faulty" files contains 20 different simulated process faults. There are 500 simulation runs for both normal and each faulty operations. Each test data run has a length of 960 (representing 48 hours of operation sampled at 1/3min). For each faulty run in the test dataset, the first 160 samples are under normal operation, with the remaining 800 samples under certain faulty condition. For fault detection rate (FDR) calculations below, we only considered the length 800 faulty region.

$$\text{FDR} = \frac{\text{number of alarms in faulty region}}{\text{total samples in faulty region}}$$

And the false alarm rate (FAR) is calculated as:

$$\text{FAR} = \frac{\text{number of alarms in NOC}}{\text{total samples in NOC}}$$

It should be noted that in this dataset, controllable faults (Fault $3, 9, 15$) have disturbances that can be dealt with by the control system, and therefore they return to normal regions. In these circumstance, the FDR is not expected to be significantly different from the FAR [27].

In our experiment setup, we take a small number of runs from each fault type in the faulty training dataset along with all the 500 runs under NOC. These faulty data are batched along with the fault free data in the training process .

2) HAI Dataset: [2] The HIL-based Augmented ICS (HAI) Security Dataset was collected from a realistic industrial control system (ICS) testbed augmented with a Hardware-In-the-Loop (HIL) simulator that emulates steam-turbine power generation and pumped-storage hydro power generation [29]. Both normal and abnormal behaviors for ICS anomaly detection are included in the dataset, with the abnormal one collected based on various attack scenarios with the six control loops in three different types of industrial devices.

The HAI testbed consists of four processes: Boiler Process (P1), Turbine Process (P2), Water-treatment Process (P3) and HIL Simulation(P4). During normal operation, it is assumed that the operator operates the facility in a routine manner, while abnormal behaviors occur when some of the parameters are outside the normal range or are in unexpected states due to attacks, malfunctions, and failures. The experiment in this paper is conducted based on the 20.07 version of HAI dataset, which has a training and testing dataset with $n = 59$ process measurements to model. The data also contains label information about whether there is an attack and where in the three processes. There are a total of 177 hours of data in the training set and 123 hours of data in the test set.

In order to mimic the situation of having a few known anomaly cases during model training, we picked 3 primitive attack scenarios from each processes as training data. These anomaly cases are not included in the subsequent testing stage for performance reporting purpose. In Table I we listed the specific cases used as training data in our experiment so readers can repeat the same setting for future studies. These cases are a subset of the 38 attacks in the 20.07 version of HAI dataset [3]

TABLE I: Few positive samples included in HAI training data

ID	Process	Start Time	Duration(sec)
A101	P1	10/29/19 13:40	370
A102	P1	10/29/19 14:35	312
A103	P1	10/29/19 15:45	868
A110	P2	10/30/19 14:30	370
A113	P2	10/31/19 8:42	348
A116	P2	10/31/19 13:25	368
A112	P3	10/30/19 16:33	154
A111	P3	10/30/19 15:35	180
A203	P3	11/1/19 11:23	180

B. Model Setups

In all experiments, we train the models with Adam optimizer [30] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and a learning rate of 0.001. Model batch size is set to be 1000 and number of epochs is 100. The metrics for calculating loss during training and deviations during anomaly detection is MSE. For both training sets, 80% of the data was used for training and 20% for validation.

Our base model is a 2 layer LSTM with hidden state dimension of 50. This is followed by a linear layer that maps the LSTM output to the final output, which has a dimension of $n = 52$ for TEP and $n = 59$ for HAI respectively. For DeepSAD, we use exactly the same model setup to represent final output encoding for a fair comparison.

For both TEP and HAI data, we take a window of length 20 as the inputs to the AR models and length 1 as the output. That is to say, we use $[\mathbf{x_{t-20}}, \mathbf{x_{t-19}}, ..., \mathbf{x_{t-1}}]$ to estimate $\mathbf{x_t}$.

For the proposed auxiliary classification approach (Auxiliary-LSTM in short), the auxiliary branch takes the LSTM output and performs a linear mapping to a dimension of 2 for a binary classification setup.

For the proposed MSE margin loss approach (Margin-LSTM in short), $p^{th}\text{percentile}of95$ is used for radius r calculation from NOC samples in each batch, a momentum of 0.9 is used for exponential moving average over batches to update r, which is initialized at 0.

For experimental results presented in this section, a simple grid search was performed to get the best parameter setting. For TEP, margin loss weight $\alpha = 0.5$, and auxiliary classification loss weight $\alpha = 0.5$. A search for DeepSAD yield

[2]HAI dataset can be downloaded at https://github.com/icsdataset/hai.

[3]HAI data attack details: https://github.com/icsdataset/hai/blob/master/hai_dataset_technical_details_v2.0.pdf

TABLE II: TEP dataset fault detection results. The FAR for the NOC is set to be 5%, and the average FDR (with standard deviation) over 10 random trials is reported for each of the 20 fault types. We compare the results obtained from our proposed approach (with 3 faulty runs), the normal data only LSTM model, and DeepSAD (with the same 3 faulty runs). Highest FDR is presented in bold.

	Normal AR Model	Normal data + Few Faults		
Fault	LSTM	Auxiliary-LSTM	Margin-LSTM	DeepSAD
1	0.9972 (0.0001)	0.9969 (0.0001)	**0.9973** (0.0003)	0.9927 (0.0008)
2	0.9816 (0.0004)	**0.9884** (0.0004)	0.9831 (0.0008)	0.9856 (0.0010)
3	0.0511 (0.0003)	0.0822 (0.0251)	0.0514 (0.0012)	**0.1091** (0.0291)
4	0.9993 (0.0003)	**0.9999** (0.0003)	**0.9999** (0.0001)	0.9092 (0.2700)
5	0.2305 (0.0066)	0.9356 (0.1200)	**0.9991** (0.001)	0.9236 (0.2181)
6	**1.0000** (0.0000)	**1.0000** (0.0000)	**1.0000** (0.0000)	0.9973 (0.0007)
7	**1.0000** (0.0000)	**1.0000** (0.0000)	**1.0000** (0.0000)	0.9861 (0.0325)
8	0.9479 (0.0019)	**0.9765** (0.0007)	0.9633 (0.0028)	0.9643 (0.0118)
9	0.0517 (0.0001)	**0.0717** (0.0126)	0.0529 (0.0011)	0.0710 (0.0088)
10	0.1364 (0.0129)	0.8418 (0.0572)	**0.8861** (0.0244)	0.7378 (0.1973)
11	0.7982 (0.0036)	**0.9818** (0.0141)	0.9091 (0.0198)	0.9081 (0.2254)
12	0.9671 (0.0016)	**0.9914** (0.0002)	0.9858 (0.0018)	0.9808 (0.0160)
13	0.9333 (0.0012)	**0.9513** (0.0012)	0.942 (0.001)	0.9407 (0.0066)
14	0.9996 (0.0001)	0.9995 (0.0000)	**0.9997** (0.0000)	0.9797 (0.0510)
15	0.0533 (0.0001)	0.0584 (0.0008)	0.0535 (0.0002)	**0.0623** (0.0038)
16	0.1282 (0.0072)	0.8974 (0.0766)	**0.9341** (0.0198)	0.7220 (0.2385)
17	0.9397 (0.0113)	**0.9623** (0.0008)	0.9621 (0.0004)	0.9179 (0.1243)
18	0.937 (0.0002)	**0.9398** (0.0007)	0.9382 (0.0005)	0.9338 (0.0077)
19	0.2335 (0.0019)	**0.7275** (0.1635)	0.4533 (0.0601)	0.3040 (0.1255)
20	0.6105 (0.0265)	0.9007 (0.0420)	**0.9096** (0.0163)	0.8303 (0.2636)
Average	0.6498	**0.8152**	0.8110	0. 6625

loss weight $\eta = 0.01$ (Equation 7 in [20], which has the same meaning as α here). For HAI, margin loss weight $\alpha = 1.0$, auxiliary classification loss weight $\alpha = 0.01$, and DeepSAD loss weight $\eta = 0.001$.

C. Results and Discussions

1) TEP Dataset: We report the FDR for all 20 fault types at the FAR of 5% and the overall performance for the TEP data. For HAI data, we report the overall performance based on both ROC (Receiver Operating Characteristic) and PR (Precision-Recall) curves. Both AUC (Area Under Curve) for the ROC curve and Average Precision (AP) are calculated as the overall performance measure.

Results for TEP dataset are summarized in Table II. We compare the fault detection results using our proposed approach with baseline normal data only model, and DeepSAD. The FAR for the normal case is set to be 5%, and the FDR are presented in this table. The FDR results in the table are the average (with standard deviation) over 10 random trial for each model. It can be seen that using as little as 3 faulty runs from each fault type in training can improve the original LSTM model significantly. Both Auxiliary-LSTM and Margin-LSTM show comparable performance, although Auxiliary-LSTM shows a slight edge in this experimental result. Although DeepSAD also improved the anomaly detection performance compared with normal data only model, the improvement is marginal.

We also report the overall performance (ROC-AUC and AP) regardless fault types in Table III. Overall, Auxiliary-LSTM and Margin-LSTM perform better than normal model only. It should be noted that normal model only has the lowest standard deviation, while Margin-LSTM is only slightly higher. On the other hand, DeepSAD has a much large variation among independent random trials.

TABLE III: TEP dataset anomaly detection results. We report average ROC AUC and AP (with standard deviation) for each model. Higher values are in bold.

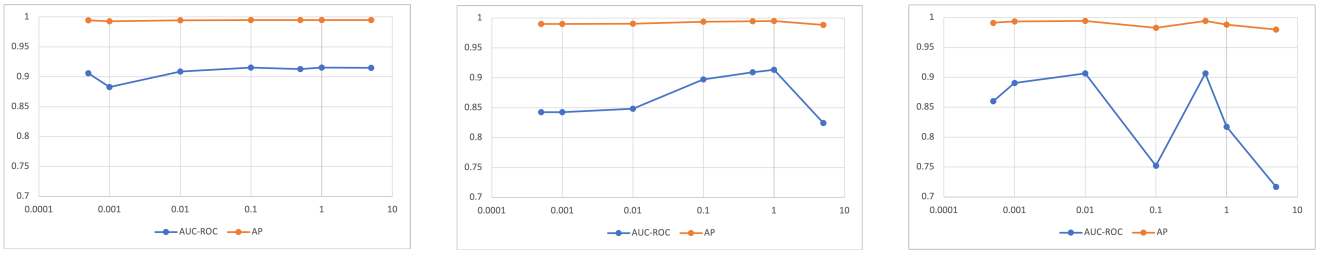
Method	ROC AUC	AP
LSTM	0.8427 (0.0010)	**0.9898** (0.0001)
Auxiliary-LSTM	**0.9133** (0.0073)	0.9948 (0.0005)
Margin-LSTM	0.9071 (0.0022)	0.9945 (0.0001)
DeepSAD	0.8829 (0.0485)	0.9926 (0.0036)

2) HAI Dataset: Results for HAI dataset are reported in Table IV. In a similar way, we compare results from normal data only model as baseline with Auxiliary-LSTM, Margin-LSTM, and DeepSAD. Similarly to the TEP study, we report both the ROC AUC and AP.

TABLE IV: HAI dataset anomaly detection results. We report average ROC AUC and AP (with standard deviation) for each model. Higher values are in bold.

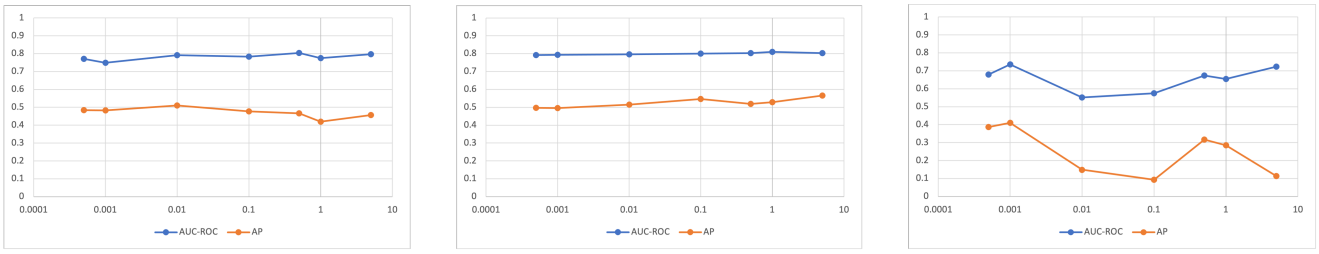
Method	ROC AUC	AP
LSTM	**0.7947** (0.0023)	0.4992 (0.0078)
Auxiliary-LSTM	0.7801(0.0234)	0.4422 (0.0251)
Margin-LSTM	0.7941 (0.0133)	**0.5226** (0.0231)
DeepSAD	0.7020 (0.0356)	0.3344 (0.1285)

In this case, we did not observe a consistent advantage from the proposed few faults model or DeepSAD. Indeed, DeepSAD actually performed worse than normal data only model. Both Auxiliary-LSTM and Margin-LSTM performs very close to normal data only model, although Margin-LSTM has a better AP.



(a) Auxiliary-LSTM (b) Margin-LSTM (c) DeepSAD

Fig. 2: Sensitivity of additional loss term weight α on model performance with TEP data



(a) Auxiliary-LSTM (b) Margin-LSTM (c) DeepSAD

Fig. 3: Sensitivity of additional loss term weight α on model performance with HAI data

3) Summary: The main difference between the proposed approach and DeepSAD is that the training contains an AR loss component. This loss component encourages the network to learn the normal dynamic relationship between future time steps and the past. In contrast, DeepSAD tries to train the network to map a window T of normal operation data to a hyper sphere. Although a network has the capacity to project a multivariate time series data into a hyper spherical space, it ignores the time series property and hence lacks the inherent induction bias that the AR learning formulation emphasizes, i.e., the future can be predicted from the past. Such induction bias aligns with the dynamic nature of the industrial time series data. Therefore, the direct projection representation learning from DeepSAD can be largely influenced by the loss term from labeled fault data, leading to large variation across random trials. Our proposed approach incorporates normality learning approach that takes into consideration of the time series nature, represented by the AR formulation, provides a better solution.

On the other hand, we have noticed that the proposed approach did not improve upon normal data only model on HAI dataset. In HAI dataset, all the attack scenarios are different from each other. Therefore, the inherent characteristics of those anomalies are different. This is in contrary to the TEP dataset, where the training data contains all the fault types. In the TEP setup, a small number of simulation runs from each fault type are included in the training data. However, the domain that represents anomalous behavior has shifted from training to testing in the HAI setup. This could explain the lackluster performance from HAI dataset in our experimental results. This also implies that the proposed approach can improve the anomaly detection performance on fault types for which we have labeled data for training, but it might have little impact on unseen fault types. The experimental results in our work motivates continued research in this area. We encourage future research work to address problems in such a setting.

D. Sensitivity Analysis

Put the base model architecture aside, the only important hyper parameter of the proposed approach is the weight α on the additional loss term as in Equation 5 and 3 . We conducted a simple search on α by setting it to a set values $\{5.0, 1.0, 0.5, 0.1, 1\text{e}{-2}, 1\text{e}{-3}, 5\text{e}{-4}\}$. We performed the same weight parameter search on DeepSAD. This weight sensitivity is graphed in Figure 2 and 3 for TEP and HAI dataset respectively. Both Auxiliary-LSTM and Margin-LSTM showed stable performance between $\alpha \in [0.5, 1]$ on both datasets. On the other hand, we observed a large variation along the search range for DeepSAD. As mentioned in the previous section, we think the induction bias from an AR formulation could alleviate the variance impact caused by a small number of faulty samples during the training process.

V. Conclusion

In this paper, we proposed a new approach to leverage the presence of a few anomalous cases on top of a large amount of normal operation data, an often encountered situation in industrial multivariate time series anomaly detection. We compared our approach with state of the art DeepSAD method, and demonstrated advantages over DeepSAD on two benchmark datasets. We demonstrated the importance of incorporating the dynamic nature of time series data to make the normal representation learning more stable and effective. Our experimental results also revealed the limitation of the current approaches in

this research area, i.e. the limited ability to improve detection sensitivity on anomaly types beyond those presented in the training data. We hope the reported results will motivate future research work to address this problem setting, which is very relevant in real-world applications.

ACKNOWLEDGMENT

This material is based upon work supported by the Department of Energy, National Energy Technology Laboratory under Award Number DE-FE0031763.[4]

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

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

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[4] G. Pang, A. V. Hengel, and C. Shen, "Weakly-supervised deep anomaly detection with pairwise relation learning," *ArXiv*, vol. abs/1910.13601, 2019.

[5] W. Ku, R. H. Storer, and C. Georgakis, "Disturbance detection and isolation by dynamic principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 30, no. 1, pp. 179–196, 1995.

[6] S. X. Ding, P. Zhang, A. Naik, E. L. Ding, and B. Huang, "Subspace method aided data-driven design of fault detection and isolation systems," *Journal of process control*, vol. 19, no. 9, pp. 1496–1510, 2009.

[7] J. Ma and S. Perkins, "Time-series novelty detection using support vector machines," in *Proceedings of the International Joint Conference on Neural Networks, 2003.*, vol. 3. IEEE, 2003, pp. 1741–1745.

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

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

[10] Y. Guo, W. Liao, Q. Wang, L. Yu, T. Ji, and P. Li, "Multidimensional time series anomaly detection: A gru-based gaussian mixture variational autoencoder approach," in *Asian Conference on Machine Learning*. PMLR, 2018, pp. 97–112.

[11] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *International Conference on Learning Representations*, 2018.

[12] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. 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*, 2019, pp. 2828–2837.

[13] F. Xue, W. Yan, T. Wang, H. Huang, and B. Feng, "Deep anomaly detection for industrial systems: a case study," in *Annual Conference of the PHM Society*, ser. 1, vol. 12, Nov. 2020.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4]Disclaimer: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

[15] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[16] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

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

[18] D. Park, Y. Hoshi, and C. C. Kemp, "A Multimodal Anomaly Detector for Robot-Assisted Feeding Using an LSTM-Based Variational Autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, Jul. 2018.

[19] M. E. Villa-Pérez, M. Á. Álvarez-Carmona, O. Loyola-González, M. A. Medina-Pérez, J. C. Velazco-Rossell, and K.-K. R. Choo, "Semi-supervised anomaly detection algorithms: A comparative summary and future research directions," *Knowledge-Based Systems*, p. 106878, 2021.

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

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

[22] Y.-L. Zhang, L. Li, J. Zhou, X. Li, and Z.-H. Zhou, "Anomaly detection with partially observed anomalies," in *Companion Proceedings of the The Web Conference 2018*, ser. WWW '18. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2018, p. 639–646. [Online]. Available: https://doi.org/10.1145/3184558.3186580

[23] F. Gao, J. Li, R. Cheng, Y. Zhou, and Y. Ye, "Connet: Deep semi-supervised anomaly detection based on sparse positive samples," *IEEE Access*, vol. 9, pp. 67 249–67 258, 2021.

[24] G. Pang, C. Shen, L. Cao, and A. v. d. Hengel, "Deep learning for anomaly detection: A review," *arXiv preprint arXiv:2007.02500*, 2020.

[25] J. J. Downs and E. F. Vogel, "A plant-wide industrial process control problem," *Computers & chemical engineering*, vol. 17, no. 3, pp. 245–255, 1993.

[26] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, "A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark tennessee eastman process," *Journal of process control*, vol. 22, no. 9, pp. 1567–1581, 2012.

[27] W. Sun, A. R. Paiva, P. Xu, A. Sundaram, and R. D. Braatz, "Fault detection and identification using bayesian recurrent neural networks," *Computers & Chemical Engineering*, vol. 141, p. 106991, 2020.

[28] C. Rieth, B. Amsel, R. Tran, and M. Cook, "Additional tennessee eastman process simulation data for anomaly detection evaluation," 2017.

[29] H.-K. Shin, W. Lee, J.-H. Yun, and H. Kim, "{HAI} 1.0: Hil-based augmented {ICS} security dataset," in *13th {USENIX} Workshop on Cyber Security Experimentation and Test ({CSET} 20)*, 2020.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.