JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 38, XXX-XXX (2022)
DOI: 10.6688/JISE.20220X_38(X).00XX

AD-FGP: Industrial Multivariate Time-Series Anomaly Detection via Fusion of Generative and Predictive Models

YONG JIN[1], YANG-HUA GAO[1], WEI-DONG LOU[1], ZE-LIANG ZHENG[2], AND SHENG-DUO GAN[2]

[1]*Information Center*
China Tobacco Zhejiang Industrial Co., Ltd.
Hangzhou, 311500, P.R. China
E-mail: 21051048@zju.edu.cn; gaoyh@zjtobacco.com; louwd@zjtobacco.com

[2]*College of Computer Science and Technology*
Zhejiang University of Technology
Hangzhou, 310023, P.R. China
E-mail: 1292239882@qq.com; 111122120021@zjut.edu.cn

Anomaly detection on industrial multivariate time-series data is an important research topic for industrial control systems. Due to the high dimensionality of industrial multivariate time-series and the lack of labeled anomaly samples, deep neural networks with the ability of learning temporal patterns in an unsupervised way have become the mainstream techniques, but there is still remaining limitations. First, they have not explicitly modeled the complex correlations between different dimensions. Second, they cannot make a balance between pattern deviation anomalies and single metric anomalies. Aiming at these limitations, this paper proposes AD-FGP, a framework for industrial multivariate time-series anomaly detection. AD-FGP has two novel features. First, it explicitly learns the correlations between different dimensions using a graph neural network. Second, it fuses a generative model and a predictive model to detect both pattern deviation anomalies and single metric anomalies effectively. We conducted extensive experiments based on both real-world and public datasets. Experiment results show that AD-FGP has a best overall anomaly detection performance by increasing the F1-score 5% to 40% as compared to the baseline methods.

Keywords: anomaly detection, multivariate time-series, industrial control system, graph neural network

1. INTRODUCTION

Industrial Control System (ICS) refers to the technology used to monitor and control industrial processes. These systems can range from simple machines that control a single function, to complex networks of interconnected devices that control entire factories [1]. As ICSs become more complex and interconnected, there is a growing need for anomaly detection in ICSs. Anomaly detection is the identification of deviations from expected behavior, which can indicate a malfunction, an attack, or other types of unexpected events [2]. Early detection of anomalies can prevent equipment failures, improve process efficiency, and enhance overall performance.

To establish information exchange between humans, devices, and systems, ICSs connect a large number of monitoring devices and sensors and collect data that can reflect the process and status of production. These data are in the form of multivariate time-series [3]. Therefore, multivariate time-series analysis has become the main tech-

nique for anomaly detection in ICSs [3, 4]. However, industrial multivariate time-series anomaly detection is a challenging task due to the following reasons.

High dimensionality: The data of ICSs are continuously collected from a large number of monitoring devices and sensors, resulting in high-dimensional time series data, making it challenging to extract effective features. For example, we have collected real ICS data from the production line of a tobacco factory, which has more than 128 monitoring devices and sensors.

Complex correlations: The different dimensions of ICSs (i.e., the data collected from different monitoring devices or sensors) might have potential correlations. For example, the data of one sensor could affect the data of another sensor, and a specific state of the ICS is jointly influenced by the data from multiple monitoring devices.

Lack of labeled data: The operating processes and collected data of ICSs are highly complex, making it difficult to understand the abnormal states. Therefore, labeling the abnormal states in the collected data of ICSs requires in-depth domain knowledge and is of high cost. In practice, it is usually difficult or impossible to obtain abnormal samples from real ICSs.

Most state-of-the-art industrial anomaly detection methods apply data-driven techniques, including traditional machine learning and deep learning. Due to the lack of labeling data, most previous studies focus on unsupervised learning techniques [5, 6]. In traditional machine learning techniques, existing studies typically involve two steps. They firstly extract a large number of features from industrial multivariate time-series data, and then discover anomalies by using outlier detection algorithms (e.g., clustering [7], One-Class SVM [8], iForest [9], etc.). However, feature engineering relies on domain knowledge, and the high-dimensional and dynamic nature of ICS data makes it difficult to design effective features. In recent years, thanks to the ability of automatically learning features from high-dimensional data, deep learning has been increasingly applied to the ICS anomaly detection tasks. Since univariate time-series anomaly detection methods (e.g., threshold-based methods) cannot detect pattern deviation anomalies [5], most existing multivariate time-series anomaly detection studies design the deep neural networks based on the generative encoder-decoder framework [10, 11]. Specifically, they use an encoder to learn the low-dimensional semantic features of the multivariate time-series data, and use a decoder to reconstruct the normal data. Then, the reconstruction errors are used to detect anomalies. In addition, due to the temporal nature of ICS data, most existing studies design the encoders and decoders based on RNN (Recurrent Neural Networks) [10, 12].

However, the generative "RNN + encoder-decoder" based ICS anomaly detection methods still have limitations. First, "RNN + encoder-decoder" cannot effectively learn the complex correlations between different dimensions, since it has not explicitly modeled the relationships between different monitoring devices and sensors. Second, it is difficult for "RNN + encoder-decoder" to make a balance between single metric anomalies and pattern deviation anomalies. The "RNN + encoder-decoder" usually computes the cumulative reconstruction errors by reconstructing a window spanning a period of time, and thus the sudden change of a metric in a single point of time could be ignored. If we reduce the size of the window to fit the single metric anomalies, it might easily miss the pattern deviation anomalies that span a longer duration.

To address these limitations, we propose AD-FGP, a novel framework for industrial

multivariate time-series anomaly detection via the fusion of both generative and predictive models. AD-FGP uses the following approaches to overcome the above mentioned two limitations. First, AD-FGP uses graphs to explicitly model the correlations between different monitoring devices and sensors, and applies GNNs (Graph Neural Networks) to learn the correlation strengths to integrate into the anomaly detection models. Second, AD-FGP fuses a generative model and a predictive model to make a balance between single metric anomalies and pattern deviation anomalies. Specifically, the generative model is used to detect pattern deviation anomalies based on the encoder-decoder architecture. The predictive model is used to detect single metric anomalies by evaluating the prediction error of each single point of time.

In summary, the contributions of this paper are as follows.

First, we propose a hybrid industrial multivariate time-series anomaly detection framework by fusing generative model and predictive model, to effectively detect both pattern deviation anomalies and single metric anomalies.

Second, we design the generative model and the predictive model by using a GNN to explicitly learn the complex correlations between different monitoring devices and sensors, and integrate the correlations into the generative and predictive models.

Third, we conducted extensive experiments on both real-world production line datasets and public datasets. The results show that AD-FGP increases the F1-score of detecting anomalies by approximately 5% to 40% as compared to the existing methods.

2. RELATED WORK

ICS data are typical multivariate time-series data, which have the characteristics of large volume, high dimensions, and strong dynamics, and thus the traditional univariate anomaly detection methods [5, 6, 7, 8] could not work effectively.

In order to adapt to the complex multivariate time-series data, most existing studies utilize learning based techniques (including machine learning and deep learning), which can be roughly divided into two categories, i.e., supervised learning techniques and unsupervised learning techniques. In terms of supervised learning techniques, Griffin *et al*. [13] proposed an anomaly detection method based on neural networks and decision trees to detect anomalies in multiple processing processes. Nanduri *et al*. [12] proposed the use of recurrent neural networks to detect anomalous events that may reduce flight safety factors. Janssens *et al*. [14] proposed a CNN-based feature learning system for detecting fault states of rotating machinery.

Although supervised learning techniques can better distinguish anomalies by explicitly learning the latent patterns from anomaly samples, it is difficult to implement in practice due to the extremely lack of labeled anomaly training data. Since normal ICS data can be easily obtained in a massive scale, unsupervised learning techniques are mostly applied for the industrial multivariate time-series anomaly detection task. In early stage, traditional unsupervised machine learning such as clustering, One-Class SVM, and iForest are applied. For example, Amruthnath and Gupta [15] applied multiple clustering algorithms for anomaly detection (e.g., K-Means, fuzzy C-Means). The anomaly detection of these algorithms can be defined as the process of identifying behaviors that deviate from the standard behavior. Diez-Olivan *et al*. [16] proposed an anomaly detection

method based on One-Class SVM to detect anomalies in sensor data by obtaining anomaly scores based on the distance between samples and the separating hyperplane. Joshi *et al*. [17] conducted anomaly detection based on HMM, which builds classifier by extracting features and calculating the anomaly probability in the state sequence generated by the model. Li *et al*. [18] proposed an anomaly detection method based on One-Class SVM to detect anomalies in image by building a generative one-class classifier on self-supervised deep representations.

However, the performance of traditional machine learning techniques heavily relies on feature engineering, while it is difficult to design effective features for the ICS data due to the high dimensionality and high dynamicity. In response to this problem, deep learning techniques have been extensively exploited for ICS anomaly detection in recent years, where the generative encoder-decoder deep neural networks are the mostly exploited strategy. For example, Kingma *et al*. [19] applied a VAE (Variational AutoEncoder) to detect anomalies by reconstructing the data and analyzing the residuals of the reconstructed data and the source data. Lu *et al*. [20] detected anomalies in rotating mechanical components by using a stacked autoencoder. Zhang *et al*. [21] proposed MSCRED, which uses ConvLSTM autoencoder to learn the multiple levels of system operation patterns characterized by multi-scale signature matrices in different time steps. Yin *et al*. [22] integrated the CNN and recurrent autoencoder for detecting anomalies in IoT systems. Specifically, they used a two-stage sliding window strategy to design the encoder for better feature extraction. Muneer *et al*. [23] proposed a hybrid model based on Deep Autoencoder Neural Network (DANN) with 5 layers for detecting anomalies in a real-world gas turbine dataset. Although the generative encoder-decoder deep neural networks have achieved promised results for ICS anomaly detection, there is still room for improvement if we can explicitly learn the relationships between different dimensions of time-series and appropriately making a balance between pattern deviation anomalies and single metric anomalies.

3. METHODOLOGY

3.1 Preliminaries

In this section, we firstly define the concepts and problem, and then present the architecture of AD-FGP.

Definition 1 (*Industrial Multivariate Time-Series*): The industrial multivariate time-series data are sampled from multiple sensors over time, denoted as $X \in R^{N \times T}$, where N is the number of sensors and T is the length of sampling time. Note that a real sensor might generate multiple readings at each time slot, e.g., an accelerometer generates three readings at each time slot. In this paper, we treat each univariate time-series in X as sampled from an individual virtual sensor, for convenience of presentation.

Definition 2 (*ICS Sample*): We use a sliding window of length W to segment X into a large number of ICS samples. Then, the ICS sample of time slot t is $X_t \in R^{N \times W}$, which is a subsequence of X during the time period of (t-W, t]. Before inputting to the anomaly detection model, the ICS samples are preprocessed based on min-max normalization and linear interpolation.

Problem Definition: The anomaly detection problem in this paper is defined as learning from a large number of normal ICS samples to obtain a function *f*, which takes X_t as input and produces an output value $y_t \in \{0,1\}$, denotes whether the ICS sample S_t is an anomaly.

Architecture of AD-FGP: Fig. 1 shows the architecture of AD-FGP, which utilizes two models (i.e., a generative model and a predictive model) to jointly detect anomalies. The generative model is trained to reconstruct the input X_t (we denote the reconstruction result as $\tilde{X}_t$). It is a deep neural network based on the encoder-decoder framework, which consists of three layers, i.e., the GAT (Graph Attention Network) layer, the encoder layer, and the decoder layer. The predictive model is trained to predict the sensor reading vector at time slot *t* given $X_t[t\text{-}W\text{+}1{:}t\text{-}1]$, i.e., the subsequence of X_t during the time period of (*t*-*W*, *t*-1] (we denote the prediction result as $\tilde{x}_t$). It is a deep spatiotemporal neural network that consists of three layers, i.e., the graph learning layer, the convolution layer, and the recurrent layer. Here, the GAT layer, the graph learning layer, and the convolution layer are used to learn the correlations between different sensors by using graph model. The encoder layer and the recurrent layer are used to learn the temporal patterns of the ICS samples. After training the two models, the anomaly score is calculated by jointly considering the reconstruction error of the generative model (i.e., the error between X_t and $\tilde{X}_t$) and the prediction error of the predictive model (i.e., the error between x_t and $\tilde{x}_t$).

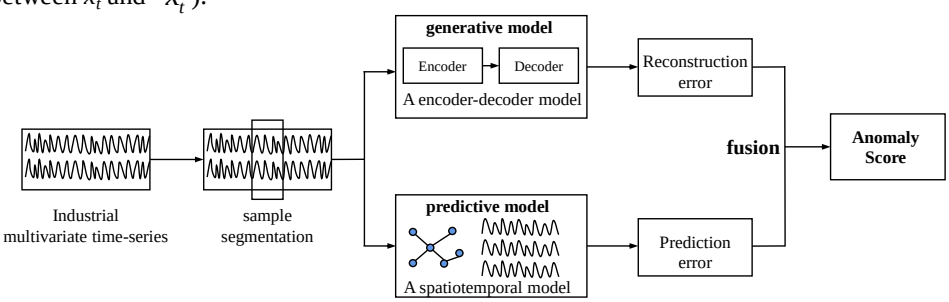

Fig. 1. The architecture of AD-FGP.

3.2 The Generative Model

The generative model is composed of three layers, i.e., the GAT layer, the encoder layer, and the decoder layer.

3.2.1 The GAT Layer

The industrial time-series data are usually collected from a large number of sensors, resulting in high-dimensional characteristics. First, these dimensions that represent different sensors have potential correlations and cross-effect. For example, the detection of some anomalies requires the consideration of the data from multiple sensors simultaneously. Second, the data from different sensors might have different levels of impact on the anomaly detection tasks. For example, pressure data in a hydraulic system are usually more important for anomaly detection than the data from other sensors. Aiming at these

issues, the GAT layer uses a graph to capture the correlations between different dimensions of the industrial time-series data (Step 1), and then the correlation strengths are learnt using a GAT subnetwork (Step 2).

Step 1 (Sensor graph construction): A sensor graph is represented as $G = (V, E, F)$, where each node $v_i \in V$ represents a dimension of the industrial time-series data (corresponding to a sensor), each edge $e_{ij} \in E$ represents a correlation between sensor v_i and v_j, and F represents the set of features with each element f_i represents the original feature of node v_i (i.e., the univariate time-series data collected from sensor v_i). To construct the sensor graph, whether two sensors have correlation can be decided based on domain knowledge. However, since domain knowledge is usually difficult to obtain, we design G as a fully connected graph if the domain knowledge is unavailable.

Step 2 (Correlation strength learning): We apply GAT to learn the correlation strength between each pair of node in G. As shown in Figure 2, for each edge e_{ij} in G, it correlation strength w_{ij} is calculated based on Equation 1, where q is a learnable parameter vector, $\sigma(\ldots)$ is a nonlinear activation function, $\oplus$ is the concatenation operation, and L is the number of nodes in the 1st-order neighbors of node v_i (including v_i itself). Then, the embedding vector of each node v_i (denoted as g_i) is updated according to Equation 3, where f_k is the original feature of node v_k.

$$w_{ij} = \frac{\exp(a_{ij})}{\sum_{k=1}^{L} \exp(a_{ik})} \tag{1}$$

$$a_{ij} = \sigma(q^{\mathrm{T}} \cdot (f_i \oplus f_j)) \tag{2}$$

$$g_i = \sigma(\sum_{k=1}^{L} w_{ik} f_k) \tag{3}$$

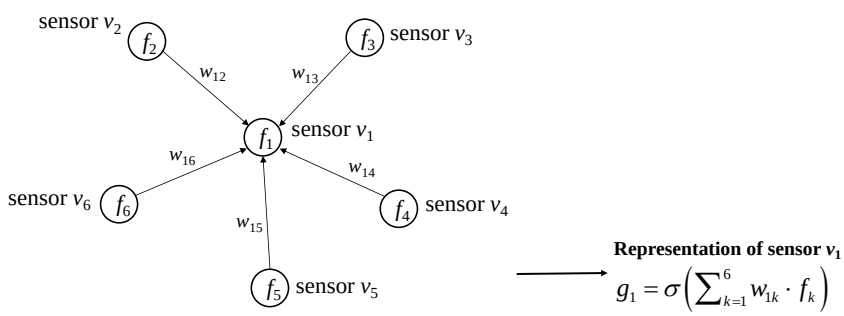

Fig. 2. The correlation strength learning based on GAT.

After the two steps, each ICS sample X_t could be represented as a feature matrix $Y_t \in R^{N \times W}$, where the i-th row of Y_t represents the embedding vector of node v_i.

3.2.2 The Encoder Layer

In this paper, the encoder-decoder subnetwork is essentially an autoencoder, which maps the original ICS samples into a lower-dimensional latent feature space using an encoder and then reconstructs the latent features into the original sample space using a decoder. The autoencoder is trained by gradually reducing the errors between original samples and reconstructed samples through backpropagation.

Since the latent features have lower dimension that that of the original samples, the latent features can be considered as the extracted main patterns of the original samples. For the anomaly detection task, the autoencoder is trained based on normal samples (or mostly normal samples), so the latent features of the trained autoencoder can represent the main patterns of normal samples. If a reconstructed sample greatly deviates from the original sample, it means that the original sample does not conform to the main patterns of normal samples, indicating an anomaly.

Considering the temporal characteristics of the ICS samples, we use a LSTM as the encoder. As shown in Figure 3, given a ICS sample $X_t = [x_{t-W}, x_{t-W+1}, \ldots, x_t]$ ($x_t \in R^{N\times 1}$ is the sensor readings at time slot t), we firstly input X_t into the GAT layer and obtain the updated feature matrix $Y_t = [y_{t-W}, y_{t-W+1}, \ldots, y_t]$. Then, we input Y_t into a LSTM subnetwork and obtain W hidden state vectors $h^e_{t-W}, h^e_{t-W+1}, ..., h^e_t$ based on Equation 4. We take the final hidden state vector h^e_t as the output of the encoder layer.

$$h^e_t = \mathrm{LSTM}(h^e_{t-1}, y_t) \tag{4}$$

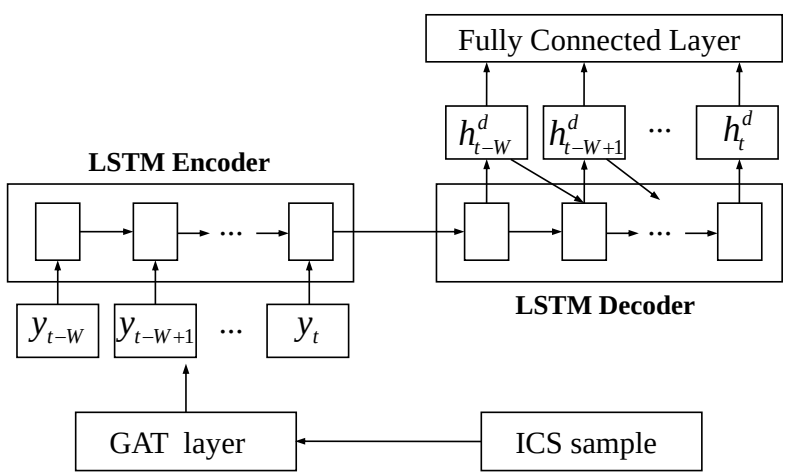

Fig. 3. The architecture of the LSTM autoencoder.

3.2.3 The Decoder Layer

As shown in Figure 3, the decoder takes h^e_t as input and also uses a LSTM subnetwork to output W hidden state vectors $h^d_{t-W}, h^d_{t-W+1}, ..., h^d_t$ based on Equation 5. After that, a fully connected subnetwork is used to reconstruct $h^d_{t-W}, h^d_{t-W+1}, ..., h^d_t$ into the same dimensions of the original samples $\tilde{X}_t \in R^{N\times W}$.

$$h_t^d = \text{LSTM}(h_{t-1}^d, h_t^e) \tag{5}$$

To train the three-layered generative model, we define the loss function as the error between the original input sample X_t and the reconstructed sample $\tilde{X}_t$, as Equation 6.

$$Loss_g = \sqrt{\frac{\sum_{i=0}^{N}\sum_{j=0}^{W}(X_t[i,j]-\tilde{X}_t[i,j])^2}{NW}} \tag{6}$$

3.3 The Predictive Model

The key capability requirement for the predictive model is to capture the spatial and temporal correlations in the multivariate time-series data. The spatial correlation indicates that the future data evolvement of a dimension could be affected by the data of other dimensions. The temporal correlation means that the future data evolvement could be affected by recent or periodic historical signals. Taking this requirement into account, the predictive model is designed to be composed of three layers, i.e., the graph learning layer, the graph convolution layer, and the recurrent layer. The graph learning layer and the graph convolution layer are used to capture the spatial correlation, and the recurrent layer is used to capture the temporal correlation.

3.3.1 The Graph Learning Layer

We also use a graph to represent the correlations between different dimensions of the industrial time-series data for the predictive model. The graph learning layer is designed to automatically learn the correlation strengths, i.e., the graph adjacency matrix. Most existing distance metrics for computing correlation strengths are symmetric. However, in the multivariate time series prediction task, we expect that the correlation strengths are asymmetric, i.e., the influence of node v_i on node v_j might be different from that of node v_j on node v_i. Therefore, we try to learn the graph adjacency matrix A based on Equation 8, where E_1 and E_2 are two randomly initialized node embedding matrices, Θ_1 and Θ_2 are learnable parameter matrices, α is a hyper-parameter to control the saturation rate of the activation function. In Equation 8, the subtractive term and the ReLU activation function regularize A in such a way that if $A[i, j]$ has a positive value, $A[j, i]$ will be zero, so as to force A to be asymmetric.

$$M_1 = \tanh(\alpha E_1\Theta_1), M_2 = \tanh(\alpha E_2\Theta_2) \tag{7}$$

$$A = \text{ReLU}(\tanh(\alpha(M_1M_2^{\text{T}} - M_2M_1^{\text{T}}))) \tag{8}$$

The graph adjacency matrix A computed based on Equation 8 represents a fully connected graph, which could result in high computation complexity. Aiming at this problem, we prune the graph by removing the edges whose correlation strength is lower than a threshold.

3.3.2 The Graph Convolution Layer

The graph convolution layer is used to update the features of each node by fusing its neighbors' information based on the graph adjacency matrix *A*. Then, the prediction for each targeted sensor can take into account other sensors that have strong influence on the targeted sensor. We apply a GCN (Graph Convolutional Network) to achieve this goal. Specifically, given the ICS sample X_t and the corresponding graph adjacency matrix *A*, the GCN updates the node feature matrix based on Equation 9, where $\tilde{A} = A + I$ represents the graph adjacent matrix with self-connections added, *I* is an identity matrix, $\tilde{D}$ is a diagonal matrix such that $\tilde{D}[i,i] = \sum_j \tilde{A}[i, j]$, and $W^{(l)}$ is the trainable weight matrix of the *l*-th layer. We can stack more GCN layers to model higher order interactions of neighbors. The input to the graph convolution layer is $H^{(0)} = X_t[t\text{-}W\text{+}1{:}t\text{-}1]$, and the output is $Z_t = H^{(L)}$, where *L* is the number of the GCN layers.

$$H^{(l+1)} = \mathrm{ReLU}(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{9}$$

3.3.3 The Recurrent Layer

The recurrent layer is utilized to capture the temporal correlation in the output of the graph convolution layer. We also use LSTM to implement the recurrent layer. Given a ICS sample $X_t = [x_{t-W}, x_{t-W+1}, \ldots, x_t]$, we firstly input $X_t[t\text{-}W\text{+}1{:}t\text{-}1]$ into the graph learning layer and the graph convolution layer, and obtain the output $Z_t = [z_{t-W}, z_{t-W+1}, \ldots, z_{t-1}]$. Then, $z_{t-W}, z_{t-W+1}, \ldots, z_{t-1}$ are chronologically fed into a LSTM subnetwork, and *W* hidden state vectors are generated, i.e., $h_{t-W}, h_{t-W+1}, \ldots, h_{t-1}$.

$$h_t = \mathrm{LSTM}(h_{t-1}, z_t) \tag{10}$$

After that, we stack a fully connected layer upon h_{t-1} to map it into the prediction result vector $\tilde{x}_t$. The predictive model is trained to minimize the loss function in Equation 11.

$$Loss_p = \sqrt{\frac{\sum_{i=0}^{N}(x_t[i] - \tilde{x}_t[i])^2}{N}} \tag{11}$$

3.4 Model fusion

The advantage of generative model is that it can accurately reconstruct the whole ICS sample by capturing the overall patterns of the multivariate time-series data. Thus, the generative model is capable of discovering the pattern deviation anomalies. On the other hand, the advantage of predictive model is that it can precisely predict the signals of a single time slot by capturing the evolving trend of the multivariate time-series data. Hence, the predictive model is capable of discovering the single metric anomalies. The fusion model combines the advantages of both predictive and generative models. Specifically, in the training phase, these two models are trained jointly by combining their loss functions as in Equation 12.

$$Loss = Loss_g + Loss_p \tag{11}$$

In the detection phase, given a ICS sample $X_t \in R^{N \times W}$, we calculate its anomaly score by using the fusion model based on Equation 12, where the left term is the anomaly score from the generative model, the right term is the anomaly score from the predictive model, and λ is a weight parameter to adjust the importance of the two models. Here, we use the summation operation for the generative model in order to detect anomalies with no sudden deviation at a single time slot, and we use the maximum operation for the predictive model in order to detect anomalies with significant deviation at a single time slot. Finally, we determine X_t as an anomaly ICS sample, if $score(X_t) > \delta$, a pre-defined threshold.

$$score(X_t) = \lambda \frac{\sum_{i=0}^{N} \sum_{j=0}^{W} (X_t[i, j] - \tilde{X}_t[i, j])}{NW} + (1-\lambda) \max_{i \in [1,N]} \{(x_t[i] - \tilde{x}_t[i])\} \tag{12}$$

4. EXPERIMENT

4.1 Experiment Setup

4.1.1 Datasets

We evaluate our method based on the following three ICS datasets.

ZJT datasets: It was collected from the production line of China Tobacco Zhejiang Industrial Company. The data was sampled every two seconds for a week from 162 sensors deployed on a variety of production devices (e.g., paper cutting wheel, power supply, etc.). Since ZJT is a dataset from real-world production line, it does not contain serious anomalies from accidents or attacks. Thus, we treat the states of transforming between different producing modes as anomalies. The ratio of normal states to abnormal states is 4:1.

HAI dataset[1]: It was collected from a realistic ICS testbed augmented with a HIL (Hardware-In-the-Loop) simulator that emulates steam-turbine power generation and pumped-storage hydropower generation. The dataset collection spans 11 days. It contains data collected every second from 84 sensors and actuators. The anomalies are generated from 50 cyber-attacks. The training set and the testing set are explicitly separated in HAI, where the training set is collected without anomalies and the testing set contains 1/40 abnormal samples.

PS datasets[2]: It was collected by Mississippi State University and Oak Ridge National Laboratory. It involves five types of anomalies, including short-circuit fault, line maintenance, remote tripping command injection, relay setting change, and data injection. The PS dataset was collected from 128 sensors. The ratio of normal states to abnormal states is 6:4.

4.1.2 Evaluation strategies

[1] https://github.com/icsdataset/hai.

[2] http://www.ece.uah.edu/~thm0009/icsdatasets/binaryAllNaturalPlusNormalVsAttacks.7z

Due to the extremely low frequency of abnormal events, the datasets are seriously unbalanced. Only considering accuracy makes no sense, since the model can have a high accuracy even when it cannot detect abnormal events. Hence, in addition to *Accuracy*, we use another two evaluation metrics, i.e., *Precision*, *Recall*, and F1-Score specifically for anomaly detection, as shown in Equation 13 and 14, where *TP* is the number of abnormal samples that are accurately detected, *FP* is the number of normal samples that are mistakenly identified as abnormal, and FN is the number of abnormal samples that are mistakenly identified as normal.

$$Precision = \frac{TP}{TP + FP} \tag{13}$$

$$Recall = \frac{TP}{TP + FN} \tag{14}$$

$$F1 = \frac{2 \times Precition \times Recall}{Precision + Recall} \tag{15}$$

For the training and testing set construction, the HAI dataset has explicitly provided training set and testing set, and we split the datasets in a ratio of 4:1 to construct training set and testing set for the ZJT and PS datasets.

4.2 Experiment 1: Parameter Tuning Experiment

The most important parameter of AD-FGP is δ, the threshold of the anomaly score. We increase δ from 0.1 to 1.5, and the experiment results for the three datasets are shown in Figure 4. It can be found that Precision and Recall have opposite evolving patterns. By increasing δ, the model would become stricter for anomaly detection, and thus there would be less anomalies being detected. As a result, the Precision shows a stable increasing phase and the Recall shows a stable decreasing phase. In the three datasets, the best overall performance can be achieved by setting δ in a narrow range of (0.1, 0.3). Consider that the model is trained in an unsupervised way, we cannot tune a specific value of δ for each dataset in practice, and thus we set δ = 0.2 for all the three datasets.

4.3 Experiment 2: Comparison Experiment

To evaluate the competitive performance of AD-FGP, we compare it with the following six baseline methods. All the methods work in the unsupervised learning style without abnormal samples, and they have been optimized to output their best performance.

Threshold: It firstly calculates the value range for each dimension in the normal samples (i.e., the maximum value and minimum value). Then, given a real-time ICS sample, if the value of any of its dimension exceeds the corresponding normal value range, it would be identified as abnormal.

OC-SVM: It refers to the One-Class SVM anomaly detection model [8]. Specifically, it firstly learns a boundary that encapsulates the normal samples in a high-dimensional feature space, and then detects abnormal samples that fall outside this boundary. We directly use the original sensor readings in the ICS samples as features.

iForest: It refers to the iForest anomaly detection model by leveraging an ensemble of trees [9]. Specifically, it firstly splits the samples according to the features using multiple decision trees, and then considers samples with lower average path lengths as anomalies. We also directly use the original sensor readings in the ICS samples as features.

LSTM-AE: It refers to the generative anomaly detection model that uses LSTM as encoder and decoder [24]. Specifically, the encoder projects the input multivariate time-series data into a latent space, and then the decoder reconstructs the input from the latent representation. The anomalies are identified based on the reconstruction errors. We utilize a stacked architecture with two LSTM layers.

GAT-AE: It refers to the generative anomaly detection model that uses GAT as encoder and LSTM as decoder [25]. Specifically, it firstly uses GAT to learn the correlations between different dimensions, and then works like LSTM-AE.

MTGNN-AD: It refers to a predictive anomaly detection model that applies the MTGNN model proposed in [26]. Specifically, it firstly trains a multivariate time-series prediction model based on MTGNN, and then detects anomalies by comparing the predicted values and the true values.

The comparison results are shown in Table 1, and the following tendencies could be discerned from the results.

First, although Threshold is the mostly applied strategy in engineering practice, it has very low performance on detecting anomalies in the three datasets, especially the pattern deviation anomalies that do not have sudden changes in sensor signals. Based on the analysis on the experiment results, we find that Threshold can detect simple anomalies such as equipment shutdown, continuous low/high temperature readings or control system failure. However, most anomalies in the three datasets are caused by complex events (e.g., producing mode switches, cyber-attacks). For example, producing mode switches would usually result in a signal pattern change instead of significant increase or decrease to the sensor signal readings. Cyber-attacks are stealthy events, which would also try not to significantly change the sensor signal readings.

Second, deep learning based methods (i.e., LSTM-AE, GAT-AE, MTGNN-AD, and AD-FGP) have a much better performance than traditional machine learning based methods (i.e., OC-SVM and iForest). This is because these deep learning based methods can capture the spatial temporal correlations of the multivariate ICS data, while the traditional machine learning based methods only extract very superficial features.

Third, GAT-AE outperforms LSTM-AE. It shows that graphs can capture the spatial correlations between different dimensions in the multivariate time-series data, which are effective in the anomaly detection tasks.

Fourth, GAT-AE generally has better performance than MTGNN-AD, except for the PS dataset. Since GAT-AE is a generative model and MTGNN-AD is a predictive model, it indicates that there are more pattern deviation anomalies than single metric anomalies in these datasets. It also shows that the complex events and stealthy attacks tend to generate pattern deviation anomalies, which cannot be effectively detected based on simple sensor signal changes.

Fifth, the Accuracy is insensitive to the methods, because these datasets are label imbalance, i.e., the majority of samples are benign. AD-FGP has the best overall performance on detecting anomalies. It increases the F1-score by approximately 5% to 40% as

compared to the best baseline method. It shows that it could achieve a better performance by fusing generative model and predictive model.

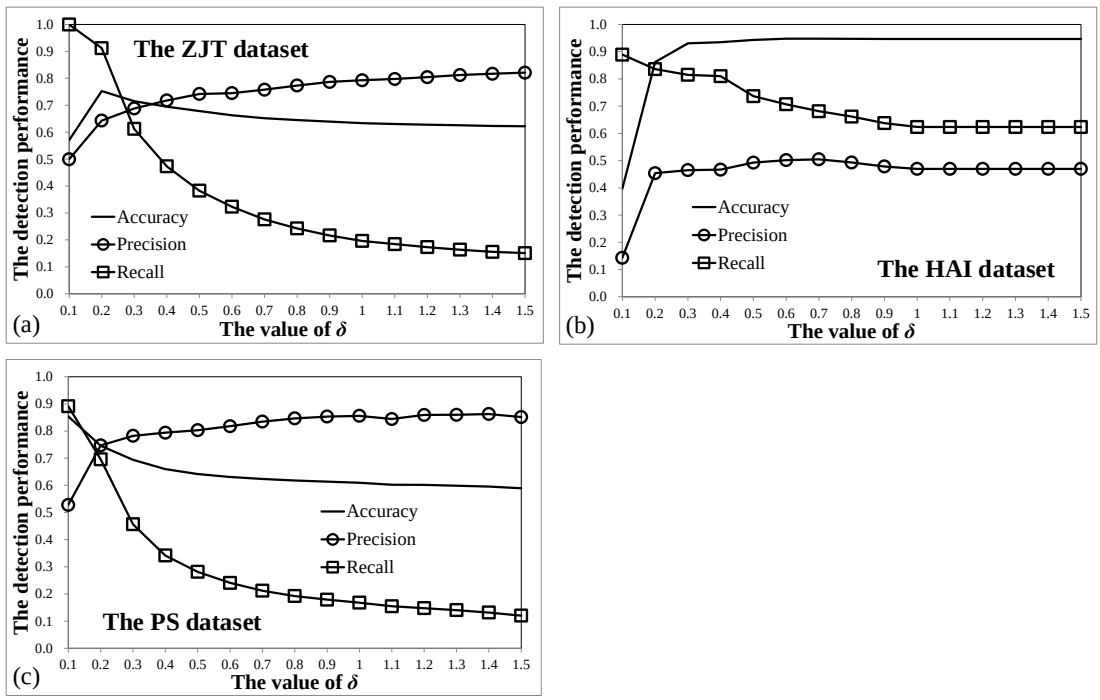

Fig. 4. The effect of parameter δ.

Table 1. The comparison experiment results.

	Accuracy	Precision	Recall	F1
ZJT-PD dataset:				
Threshold	0.563	0.135	0.097	0.113
OC-SVM	0.426	0.973	0.180	0.303
iForest	0.505	0.313	0.400	0.351
LSTM-AE	0.645	0.984	0.426	0.595
GAT-AE	0.750	0.648	0.782	0.709
MTGNN-AD	0.528	0.428	0.914	0.582
AD-FGP	0.753	0.643	0.912	0.754
HAI dataset:				

Threshold Learning	0.985	0.000	0.000	0.000
OC-SVM	0.951	0.084	0.091	0.087
iForest	0.824	0.117	0.089	0.101
LSTM-AE	0.947	0.172	0.698	0.276
GAT-AE	0.992	0.267	0.840	0.405
MTGNN-AD	0.932	0.166	0.682	0.267
AD-FGP	0.931	0.454	0.836	0.589
PS dataset:				
Threshold Learning	0.584	0.664	0.117	0.199
OC-SVM	0.546	0.652	0.153	0.248
iForest	0.608	0.775	0.173	0.283
LSTM-AE	0.572	0.720	0.299	0.422
GAT-AE	0.587	0.665	0.618	0.641
MTGNN-AD	0.745	0.644	0.685	0.664
AD-FGP	0.746	0.747	0.696	0.721

4.3 Experiment 3: Case Study

In this section, we use two cases to demonstrate the effectiveness of AD-FGP. In the first case, we extract a two-minutes data segment from the ZJT dataset, as shown in Figure 5(a). The data segment encompasses five different monitoring devices, i.e., paper cutter, industrial fan, F.R.L (air source unit), air distribution box, and soldering iron. The anomaly event occurs during the 40-seconds interval from 7:02:56 to 7:03:36. Based on the experiment results, the threshold-based method and the predictive model fail to detect this anomaly event, since the deviation of the time-series data is not sudden and significant but smooth and persistent. The generative model and AD-FGP successfully identify this anomaly event.

In the second case, we extract another 104-seconds data segment from the ZJT dataset, as shown in Figure 5(b). the data segment exhibits a sudden and significant deviation at 8:20:48. Based on the experiment results, the generative model fails to detect this anomaly event, while the threshold-based method, the predictive model, and AD-FGP successfully identify this anomaly event.

The two cases demonstrate that AD-FGP is capable of simultaneously capturing pattern deviation anomalies and single-metric anomalies, thereby enabling more accurate anomaly detection.

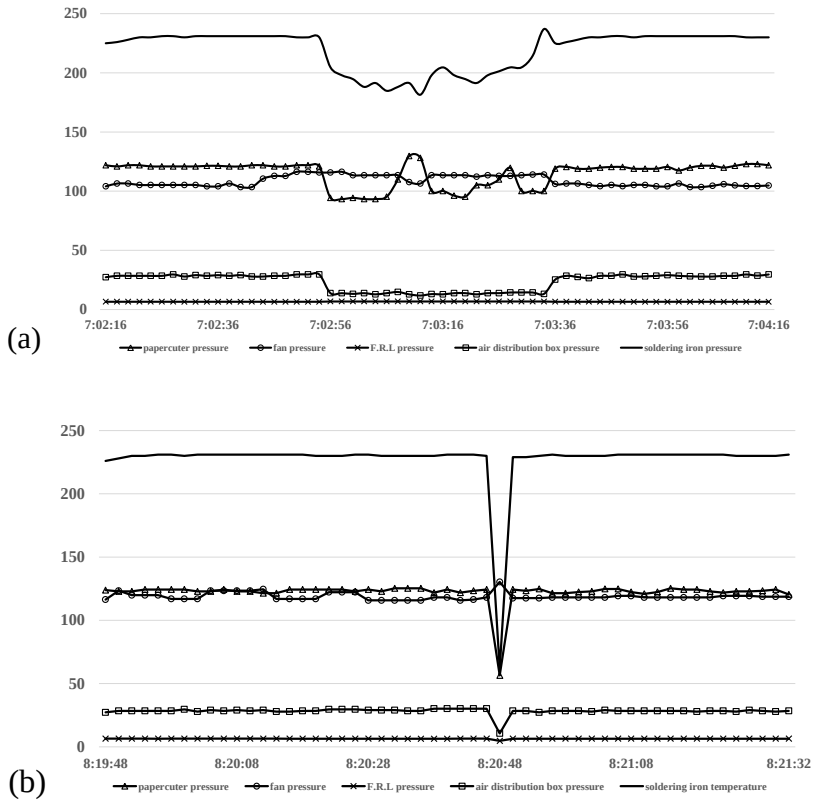

Fig. 5. A case study of AD-FGP.

5. CONCLUSIONS

In this paper, we investigate the anomaly detection in industrial control systems. We propose AD-FGP, a novel deep learning model that is trained in an unsupervised way to identify anomalies from the multivariate time-series data in industrial control systems. By combining the strengths of generative and predictive models and learning the sensor-wide correlations and temporal relations of multivariate time-series data, AD-FGP outperforms other state-of-the-art models on three datasets consistently. Specially, AD-FGP is effective at detecting both pattern deviation anomalies and single metric anomalies.

Future works could come from two aspects. First, AD-FGP can only detect anomaly events without knowing what types of anomaly events they are. Hence, enhancing AD-FGP with the ability of anomaly classification can benefit the response to these anomaly events. Second, how to diagnose the detected anomaly events and trace the sources are also worth studying.

REFERENCES

1. S. Keith, J. Falco and K. Scarfone, "Guide to industrial control systems (ICS) security," *NIST special publication*, vol. 800, no.82, pp. 16-16, 2011.
2. G. Pang, C. Shen, L. Cao et al., "Deep learning for anomaly detection: A review," ACM computing surveys (CSUR), vol. 54, no.2, pp. 1-38, 2021.
3. Z. Che, S. Purushotham, K. Cho et al., "Recurrent neural networks for multivariate time series with missing values," Scientific Reports, vol. 8, no. 1, pp. 6085, 2018.
4. X. Chen and L. Sun, "Bayesian temporal factorization for multidimensional time series prediction," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 9, pp. 4659 – 4673, 2021.
5. S. Omar, A. Ngadi, and H. H. Jebur, "Machine learning techniques for anomaly detection: an overview," International Journal of Computer Applications, vol. 79, no. 2, 2013.
6. J. Audibert, P. Michiardi, F. Guyard, et al., "Usad: Unsupervised anomaly detection on multivariate time series," in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 3395 – 3404, 2020.
7. A. Singhal and D. E. Seborg, "Clustering multivariate time-series data," Journal of Chemometrics: A Journal of the Chemometrics Society, vol. 19, no. 8, pp. 427 – 438, 2005.
8. S. M. Erfani, S. Rajasegarar, S. Karunasekera, et al., "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning," Pattern Recognition, vol. 58, pp. 121 – 134, 2016.
9. G. Thornton and P. B. Zadeh, "An investigation into Unmanned Aerial System (UAS) forensics: Data extraction & analysis," Forensic Science International: Digital Investigation, vol. 41, pp. 301379, 2022.
10. P. Malhotra, A. Ramakrishnan, G. Anand, et al., "LSTM-based encoder-decoder for multi-sensor anomaly detection," arXiv preprint arXiv, vol. 1607, no. 00148, 2016.
11. S. Akçay, A. Atapour-Abarghouei, and T. P. Breckon, "Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection," in 2019 International Joint Conference on Neural Networks (IJCNN), IEEE, pp. 1 – 8, 2019.
12. A. Nanduri and L. Sherry, "Anomaly detection in aircraft data using Recurrent Neural Networks (RNN)," in 2016 Integrated Communications Navigation and Surveillance (ICNS), IEEE, pp. 5C2-1-5C2-8, 2016.
13. J. M. Griffin, A. J. Doberti, V. Hernández, et al., "Multiple classification of the force and acceleration signals extracted during multiple machine processes: part 1 intelligent classification from an anomaly perspective," The International Journal of Advanced Manufacturing Technology, vol. 93, pp. 811 – 823, 2017.
14. O. Janssens, V. Slavkovikj, B. Vervisch, et al., "Convolutional neural network based fault detection for rotating machinery," Journal of Sound and Vibration, vol. 377, pp. 331 – 345, 2016.
15. N. Amruthnath and T. Gupta, "A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance," in 2018 5th International Conference on Industrial Engineering and Applications (ICIEA), IEEE, pp. 355 – 361, 2018.

16. A. Diez-Olivan, J. A. Pagan, N. L. D. Khoa, et al., "Kernel-based support vector machines for automated health status assessment in monitoring sensor data," The International Journal of Advanced Manufacturing Technology, vol. 95, pp. 327 – 340, 2018.
17. S. S. Joshi and V. V. Phoha, "Investigating hidden Markov models capabilities in anomaly detection," in Proceedings of the 43rd Annual Southeast Regional Conference-Volume 1, pp. 98 – 103, 2005.
18. C. L. Li, K. Sohn, J. Yoon, et al., "CutPaste: Self-supervised learning for anomaly detection and localization," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9664 – 9674, 2021.
19. D. P. Kingma and M. Welling, "Auto-encoding variational bayes," arXiv preprint arXiv, pp.1312.6114, 2013.
20. C. Lu, Z. Y. Wang, W. L. Qin, et al., "Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification," Signal Processing, vol. 130, pp. 377 – 388, 2017.
21. C. Zhang, D. Song, Y. Chen, et al., "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 1409 – 1416, 2019.
22. C. Yin, S. Zhang, J. Wang, et al., "Anomaly detection based on convolutional recurrent autoencoder for IoT time series," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 52, no. 1, pp. 112 – 122, 2020.
23. A. Muneer, S. M. Taib, S. M. Fati, et al., "A Hybrid deep learning-based unsupervised anomaly detection in high dimensional data," Computers, Materials and Continua, vol. 70, no. 3, pp. 6073 – 6088, 2022.
24. M. Said Elsayed, N. A. Le-Khac, S. Dev, et al., "Network anomaly detection using LSTM based autoencoder," in Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks, pp. 37 – 45, 2020.
25. C. Ding, S. Sun, and J. Zhao, "MST-GAT: A multimodal spatial – temporal graph attention network for time series anomaly detection," Information Fusion, vol. 89, pp. 527 – 536, 2023.
26. Z. Wu, S. Pan, G. Long, et al., "Connecting the dots: Multivariate time series forecasting with graph neural networks," in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 753 – 763, 2020.