Multivariate Abnormal Detection for Industrial Control Systems Using 1D CNN and GRU

XIN XIE, BIN WANG, TIANCHENG WAN, AND WENLIANG TANG

School of Information Engineering, East China Jiaotong University, Nanchang 330013, China
Corresponding author: Bin Wang (wangbin199702@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61762037, and in part by the Science and Technology Key Research and Development Program of Jiangxi Province under Grant 20192ACB50027.

ABSTRACT Currently, most anomaly detection approaches in industrial control systems (ICSs) use network event logs to build models, and current unsupervised machine learning methods rarely use spatiotemporal correlations and other dependencies between multiple variables (sensors/actuators) in a system to detect anomalies. Most of the existing anomaly detection technologies simply compare the current states with the predicted normal range. Due to the highly dynamic characteristic of industrial control systems, it is insufficient to simply compare the current states with the predicted normal range. As a result, these approaches have low detection rates for unknown or new types of attacks. In view of these shortcomings, this paper presents a network model for predicting sensor/controller parameters in industrial control systems. To predict the parameter values of the sensors and controllers more accurately, the 1D convolutional neural network (1D_CNN) and gated recurrent unit (GRU) are combined to fully learn the spatiotemporal correlation and other dependencies between the parameter values of the sensors and controllers at each moment. An abnormal state detection method based on the calculation of the statistical deviation is proposed to realize the anomaly detection of industrial control systems. The model is validated on the Secure Water Treatment (SWaT) dataset. The precision, recall and F1 scores are used to evaluate the effectiveness of this method in anomaly detection on the SWaT dataset. The experimental results show that the average precision and recall of this method are 0.99 and 0.85, respectively, and that the average F1 score is 0.91. The experimental results show that the proposed method can be successfully applied to anomaly detection systems in industrial control systems with lower false positive rates.

INDEX TERMS Auto-encoder, 1D convolutional neural network, gated recurrent unit, industrial control system, anomaly detection, SWaT dataset.

I. INTRODUCTION

Industrial control systems (ICSs), such as SCADA, DCS and PLC, have been widely used in industry, energy, transportation, water conservancy, municipal, and other national key infrastructure fields and are the core part of industrial automation [1]. In recent years, with the increasing degree of integration of industrialization and information, the number of interconnections between industrial equipment and the Internet has also increased sharply, and malicious attacks targeting industrial control systems through the Internet have followed [2]. A number of high-impact cyberattacks have been reported in recent years, including the infamous Stuxnet malware, which targeted nuclear centrifuges in Iran and appeared in 2010 [3], an attack on Ukraine’s power grid in 2015 that left millions of people spending Christmas in the dark [4], and, since 2016, a series of industrial control security incidents, such as the Israeli power system invasion [5] and Russian hackers’ public release of the SCADA Pass default password list of industrial control hardware and software equipment [6]. This series of events has sounded the alarm of industrial control system security.

Therefore, improving the detection ability of industrial control system attacks has become an urgent problem for academia and industry. One of the approaches is to learn from the traditional network-based intrusion detection system, collect and analyze the network traffic in the industrial control system, match it with the known abnormal traffic incidents, such as the Israeli power system invasion and Russian hackers’ public release of the SCADA Pass default password list of industrial control hardware and software equipment. Therefore, improving the detection ability of industrial control system attacks has become an urgent problem for academia and industry. One of the approaches is to learn from the traditional network-based intrusion detection system, collect and analyze the network traffic in the industrial control system, match it with the known abnormal traffic model (if the matching fails, it will be regarded as unknown abnormal traffic), and analyze it to determine whether there is a
suspicious intrusion or attack [7]. To provide intrusion detection rules governing how to prevent the invasion of detailed information to improve the ability of a system to carry out intrusion detection and prevention, Morris et al. [8] improved the requirements of the network intrusion detection systems of industrial control systems by analyzing four types of intrusion vulnerability, namely, denial of service, command injection, response injection and system reconnaissance. As increasingly more wireless smart devices are applied to ICS networks, conventional encryption methods and security patches cannot improve the security level of older devices in ICS networks. Shen et al. [9] used the simplicity of the program process and the stability of the hardware configuration to measure the interlayer data response processing time and analyze the network traffic, which enhanced the traditional intrusion detection mechanism in an ICS network. Another approach is to model the underlying physical process and predict the data, comparing it with the actual data to determine whether an exception is generated [10]. To prevent an attacker of an industrial control system from false data injection attacks, Lu and Feng [11] proposed a system state estimation method based on the particle filter estimate node status of the system, introduced a voting scheme to identify malicious nodes, estimated the actual system state accurately by building a local estimator and removing the voted malicious nodes, and finally used the prediction algorithm with the estimated actual status to predict the actual measured value of the damaged nodes. Kurt et al. [12] modeled a power grid system studied as a discrete time linear dynamic system and proposed a distributed dynamic state estimation algorithm based on a Kalman filter to estimate the system state, which greatly reduced the response time of the detector to detect structural and random attacks and improved the detection accuracy. Although these methods can greatly improve the accuracy of anomaly detection, it is difficult to build these models, which requires an in-depth understanding of the physical system and familiarity with the control process of the system [12].

With the development of machine learning and neural networks, some scholars have gradually combined deep learning with the intrusion detection of industrial control systems [13] and achieved certain results. Bangbing et al. [14], based on the industrial control intrusion detection technology of the long short-term memory (LSTM) network, selected the relatively optimal LSTM model through cross-validation and achieved higher accuracy than that of traditional intrusion detection methods. To further improve the detection accuracy of the anomaly model based on LSTM, Wang and Jin [15] combined LSTM with linear discriminant analysis (LDA), used LSTM to learn the context representation of the event log of the industrial control network, and used LDA to classify the features of the output of LSTM to achieve anomaly detection. However, the detection rates of unknown or new attacks are unsatisfactory, and it is more difficult to detect attacks based on physical layers. For this reason, Liu et al. [16] proposed a two-level anomaly detection framework. The first level extracts features through the convolutional neural network (CNN) model, and the second level takes the extracted features as the input of the process state transition algorithm to construct the normal state process transfer model of an ICS to realize the detection of unknown attacks or 0-day attacks. Kim et al. [17] applied the time series data anomaly detection to the edge calculation of the industrial Internet of things (IIoT) with a stacked compression variational autoencoder (SCVAE), which reduced the size of the model and the time spent in detection while ensuring a similar accuracy level. Because there are many controllers and sensors in industrial control systems, the data corresponding to different processes or scenarios are different. To effectively detect cross-scene network attacks, Wang et al. [18] proposed extracting nonlinear and nonstationary features from power load data with a stacked autoencoder and then using these features to improve the accuracy of power load data prediction to improve the accuracy of network attack detection.

Generally, when evaluating a model, we should consider not only the precision but also the recall rate. The F1 score [19] can reflect these two indicators at the same time. The higher the F1 score, the better the detection result of the model. Due to the low F1 score generated in the detection of multitype attacks with the above methods, when the attack types are diverse, better detection cannot be achieved. Therefore, we propose an intrusion detection system model based on the autoencoder, 1D_CNN and GRU. We use the model to detect cyberattacks on the SWaT dataset. The contributions of this paper are as follows:

- Anomaly detection method for multivariate industrial process time series data;
- A model for predicting sensor/controller parameters in industrial control systems (the spatiotemporal correlation and other dependencies between the values of the sensors and the controllers at each moment are fully learned by combining the 1D_CNN with the GRU);
- A higher F1 score for anomaly detection on the SWaT dataset when using our method.

II. CORRELATION PRINCIPLE OF AUTOENCODER

The autoencoder [20] is an unsupervised learning method that includes a feedforward symmetric network of multi layers of neurons; the structure is shown in figure 1. It is important to note that the number of hidden layers can be more than one. The process by which the network converts the input data \(v\) into the code \(h\) for representation is called encoding. To train the autoencoder, the code \(h\) obtained from the hidden layers needs to be mapped to the output layer through the decoding layers to obtain the reconstructed vector \(\hat{v}\).

The encoding process can be expressed as

\[
H = f_l(\theta_l, x) = f_l(W_l x + b_l), \quad \theta_l = \{W_l, b_l\}
\]  

and the reconstructing process can be expressed as

\[
I = f_l(\theta_l, H) = f_l(W_l H + b_l), \quad \theta_l = \{W_l, b_l\}
\]
where \( f_l \) is a nonlinear mapping of the \( l \)th hidden layer, which can be used as a sigmoid function [21], tanh function, and so on. For the first hidden layer, \( x = v \); for the first hidden layer of the reconstruction process, \( H = h \); and for the last hidden layer, \( I = \hat{v} \). The target of autoencoder is to reduce reconstruction errors.

\[
\theta^*, \theta'^* = \arg\min_{\theta, \theta'} \frac{1}{n} \sum_{i=1}^{n} L(v^i, \hat{v}^i)
\]

where \( L = \|v - \hat{v}\|^2 \). This makes the hidden layers learn a feature representation of the input data.

Current studies have found that the stacked denoising autoencoder (SDA) [22] has better performance in feature extraction. The input of the SDA is randomly damaged by randomly setting some input units to zero through random mapping \( \tilde{v} \sim qD(\tilde{v}|v) \). The SDA tries to reconstruct the original input from the damaged input version. So we can use the mean squared error as the loss function.

\[
L = \frac{1}{n} \sum_{i=1}^{n} (v^i - \tilde{v}^i)^2
\]

### III. ABNORMAL DETECTION METHOD

We propose an abnormal state detection method based on the calculation of the statistical deviation between the predicted value and the observed value. The main idea of the anomaly detection method is to predict the value of the next moment or sequence of the next time according to the existing flow in the previous period and judge whether the abnormality occurs because of the deviation between the predicted value and the actual value.

The steps are as follows: let \( X = (x_0, x_1, \ldots, x_{n-1}) \) and \( Y = (y_n, y_{n+1}, \ldots, y_{n+m}) \) be input flow sequence vectors and output flow sequence vectors at time \( i \), respectively. Since \( (y_n, y_{n+1}, \ldots, y_{n+m}) \) is the predicted value, there will be deviation from the actual observed value; thus, the deviation \( \varepsilon_i \) at time \( i \) can be set as

\[
\varepsilon_i = |\tilde{y} - y_a|
\]

where \( \tilde{y} \) is the value of a dimension’s feature in the actual observed value corresponding to the predicted value \( y_a \). Then, regularize \( \varepsilon_i \).

\[
\varepsilon = \frac{v_i - \mu_v}{\sigma_v}
\]

where \( \mu_v \) is the mean of the prediction error \( \varepsilon_i \) from time \( i \) to time \( t \) and the corresponding standard deviation is \( \sigma_v \). Calculate the mean and variance of each predicted characteristic value and observed value in \( Y \), and finally, set the threshold value \( T \) according to the deviation value. If

\[
\max \varepsilon > T
\]

then the state is regarded as abnormal; otherwise, it is normal.

### IV. ICS ANOMALY DETECTION MODEL BASED ON 1D CNN AND GRU

First, the features of the historical data are obtained through the SDA, and then the predicted features in each window are obtained through the neural network combined with the 1D-CNN and GRU. Finally, the statistical deviation between the predicted features and the observed features is calculated and compared with the set threshold to determine whether the system enters an abnormal state.

In the stage of data preprocessing, the one-hot encoder (OHE) [23] is used to convert the original flows into a vector. Since the dimensions of the vector are very high, an autoencoder is used to reduce the dimensions of the vector. To obtain a better feature vector with low dimensionality and give the model a certain anti-disturbance ability relative to an anomaly or an attack on the system, we use the SDA to extract the flow features and reduce the dimensions of the features; however, the dependencies between some features may be lost due to the dimensionality reduction.

To extract better features effectively, the network should learn higher-level and more-abstract operation and parameter features to learn industrial control system features from low-level common features. To add potential information about the feature interdependencies, we add an additional full connection layer before the CNN, which expands the number of features we provide to the CNNs, as shown in figure 2. We set the dimensions \( i \) of the extended vector to a value between the output dimensions \( m \) of the SDA and the dimensions \( n \) of the initial vector after being transformed by the OHE, i.e., \( m < i < n \).
has not only a strong functional learning ability but also its own defects. First, the gradient is calculated by chain back propagation. As the number of layers increases, an exponential decrease/increase in the gradient easily occurs. Therefore, deeper CNNs often encounter the problem of gradient disappearance or explosion, and their training becomes more difficult. Second, network degradation is another major problem, leading to an increase in the training errors of samples. Therefore, we use the 1D_CNN with the residual module to avoid these problems. The 1D_CNN introduces the idea of residual learning into the traditional 1D CNN. The 1D residual block can effectively address the deeper CNN training difficulty and performance degradation. In addition, the introduction of the pool layer further enhances the feature learning ability of CNNs in noisy environments. The performance of the network is further improved through the adoption of batch normalization.

The extended vectors are taken as the input of the 1D convolutional neural network with the residual module, as shown in figure 3.

![FIGURE 3. 1D_CNN with residual module.](image)

After the expanded vectors are normalized through batch normalization [24], the ReLU function [25] is adopted as the activation function for nonlinear mapping and then processed through a 1D convolution layer. After the above steps are carried out many times, the residual module is used for the output.

However, using only the 1D_CNN to learn the features and their dependencies will lead to deterioration of the accuracy and robustness of system prediction, which will result in a high false positive rate of multitype attack detection.

On the other hand, because the conventional RNN architecture is too sensitive to the interference of adjacent time periods, the problem of error flow disappearance still exists. Compared to the original RNN and LSTM, the gated recurrent unit overcomes the gradient disappearance problem by multiplicative gates. Generally, both the GRU and LSTM are suitable for processing time series data, but the GRU is more efficient at achieving convergence and updating weights, and the internal gate structure is more concise than LSTM. The experiments also show that the GRU is better than LSTM in this model.

By providing a series of data points to the above network structure, a single subsequent data point containing many features can be obtained. After we obtain the output vector of the 1D_CNN [26], we let it pass through a dropout layer [27] to increase the generalization capability of the network. Then, the output of the dropout layer is used as the input of the GRU neural network [28] (as shown in figure 4) to predict the subsequent flow features, and the predicted eigenvector is obtained symmetrically through a fully connected layer.

![FIGURE 4. GRU model.](image)

For the whole network mentioned above, the mean squared error (MSE) [29] is adopted as the model’s loss function to train our model.

\[
MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2
\]

where \(y_i\) is the predicted feature and \(\hat{y}_i\) is the actual feature.

We use the method that we proposed in section 3 for flow anomaly detection. First, the eigenvector \(Y'_\text{t}\) of the traffic at the next time \(t\) is obtained by using the above network model, and the eigenvector of the actual traffic at the time is \(Y'_\text{t}\). According to the formula, the deviation of the eigenvalue of dimension \(i\) in the eigenvector corresponding to the flow at time \(t\) is \(e'_t\), and the mean value \(\mu_e\) and standard deviation \(\sigma_e\) of the eigenvalue of dimension \(i\) are calculated. After regularization of the deviation, the deviation value of the eigenvalue of dimension \(i\) at time \(t\), \(\overline{Z'_t}\), is obtained. We use this method to calculate the eigenvalues of each dimension. Then, we set the threshold \(T\) and the minimum value of the time window \(w\) as hyperparameters, multiply the deviation value of each dimension to obtain an outlier, and then use the outlier to judge whether an abnormal state occurred:

\[
A_t = \prod_{e=t-w}^{t} \left( \arg \sum_{i=0}^{m-1} \overline{Z'_t} \right)
\]

where \(m\) represents the number of dimensions of the eigenvector. For the above outliers, we map them to a level between 0 and 10 to obtain the level of outliers at time \(t\).

\[
OS_t = \frac{A_t - \min (A_t)}{\max (A_t) - \min (A_t)} \times 10
\]
Finally, compared with the set threshold $T$, if $OS_t > T$, the system at time $t$ is abnormal and can be regarded as attacked; otherwise, it is normal.

**Algorithm 1** The Proposed Anomaly Detection Method

**Input:** training data, testing data

**Output:** anomaly score $OS_t$

1: Initialize weight parameters in model
2: Transform the normalized data into the initial vector by the OHE
3: for the $k$th epoch do
4: Input the data into the SDA
5: Update the weight parameters of the SDA by minimizing (descending) $L(v, \hat{v}) = \frac{1}{n} \sum_{i=1}^{n} L(v_i, \hat{v}_i)$
6: end for
7: Record parameters of the SDA and freeze layers
8: Use the SDA to extract the features of the training data and reduce the dimensions of the vectors
9: for the $m$th epoch do
10: Input the vectors into the 1D_CNN-GRU
11: Update the weight parameters of the 1D_CNN-GRU by minimizing MSE $= \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
12: end for
13: Input the testing data into the SDA-1D_CNN-GRU
14: Calculate the residuals of the prediction values $\varepsilon_t = \frac{|e_t - \mu_{e_t}|}{\sigma_{e_t}}$
15: Multiply the deviation value of each dimension to obtain an outlier, and then use the outlier to judge whether an abnormal state occurred

$$A_t = \prod_{n=t-w}^{t} \left( \arg \sum_{i=0}^{m-1} Z_{e_t}^i \right)$$

$$OS_t = \frac{A_t - \min(A_t)}{\max(A_t) - \min(A_t)} \times 10$$

**V. EXPERIMENT AND ANALYSIS**

**A. EVALUATION CRITERIA**

In this paper, the F1 score is adopted to evaluate the performance of the model, which is often used in multi-classification problems. The relevant definitions are as follows:

$$F1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (11)$$

$$\text{precision} = \frac{TP}{TP + FP} \quad (12)$$

$$\text{recall} = \frac{TP}{TP + FN} \quad (13)$$

where $TP$ (true positive) is the number of abnormal flows detected correctly, $FN$ (false negative) is the number of abnormal flows that have occurred but have not been detected, and $FP$ (false positive) is the number of normal flows that have been incorrectly marked as abnormal.

**B. DATASET**

This paper uses the SWaT dataset collected through Secure Water Treatment [30] designed by researchers in Singapore to test and verify the performance of the model.

The SWaT dataset was collected from a test-bed water treatment system. The test-bed water treatment system represents a smaller version of a large modern water treatment plant used in large cities. By codesigning the operational test platform with the relevant departments, the whole physical process and control system are very similar to the real system. As shown in figure 5, the water purification process consists of six subprocesses, i.e., P1 through P6 [31]. Each stage is controlled by a corresponding pair of programmable logic controllers (PLCs), one for the primary PLC and the other for the standby PLC. Between the different stages, the PLC communicates through a separate network called the L1 network. In addition, in the SWaT platform, all network communications between the PLCs, sensors, and actuators use industrial Ethernet/IP (ENIP) and common industrial protocol (CIP) stacks. The first process involves the raw water supply and storage, and P2 is pretreatment, in which the targets was to reduce the performance from a nominal level (for example, 5 gallons per minute) to a low value. This attack can be initiated by damaging the LIT401 sensor, which measures the water level in the P4 ROfeed tank. By attacking LIT401, the attacker lowered the horizontal level of the RO feed tank from 800 mm to 200 mm, causing PLC-4 to stop pumping P401, reducing the water pumped into P5. Finally, the attack sensor LIT401 had a negative effect on the output

![Network structure of SWaT platform](image-url)
significantly different from the KDDCUP99 dataset: in other studies. It is worth noting that the SWaT dataset is

dataset and the KDDCUP99 dataset, which is commonly used

period, the amount of treated water decreased.

Table 1 summarizes some general information about this
dataset and the KDDCUP99 dataset, which is commonly used
in other studies. It is worth noting that the SWaT dataset is
significantly different from the KDDCUP99 dataset:

water flow of the RO device (measured in FIT501 in P5).
According to system specifications, this flow must be main-
tained at approximately 1.2 cm/h, which results in nearly
5 gallons/min of treated water. Thus, during the observation
period, the amount of treated water decreased.

Table 1 summarizes some general information about this
dataset and the KDDCUP99 dataset, which is commonly used
in other studies. It is worth noting that the SWaT dataset is
significantly different from the KDDCUP99 dataset:

a) Depending on the scenario, different attacks may last
for different durations. In fact, some attacks did not take
effect immediately. The timing of system stabilization
also varies from attack to attack. Some simple attacks,
such as those aimed at changing traffic, can stabilize
the system in a short period of time, while attacks that
have a greater impact on system dynamics require more
time to stabilize the system.

b) Attacks on one sensor/actuator may affect the perform-
ance of other sensors/actuators or the entire system
after a certain time delay.

c) In addition, similar types of sensors/actuators tend to
respond to attacks in similar ways. For example, attacks
on the LIT101 sensor (a water-level sensor in phase P1)
and on both LIT101 and LIT301 (another water-level
sensor in phase P3) caused significant abnormal spikes,
but readings from other sensors and actuators (such as
flow sensors and electricity meters) had a relatively
small impact.

Therefore, compared with KDDCUP99, the SWaT dataset
is closer to a real industrial control system, and the spa-
tiotemporal relationship in the dataset is more complicated.
Therefore, it is very important to adopt a multivariate method
for anomaly detection in system modeling because the over-
all performance changes of different subprocesses can help
better identify attacks. In other words, the potential asso-
ciation between the sensors and the actuators may help in
the detection of system behavior anomalies caused by an
attack.

C. SETUP
The model was trained and tested on two Intel i7-6700k
workstations (32 GB of RAM) using the NVIDIA 8 GB
1080 GPU. The model and anomaly detection algorithm were
implemented using Google’s TensorFlow framework [32].
We used low-level APIs to facilitate fine-grained control of
the network architecture.

D. DATA PREPROCESSING
Generally, in machine learning, the data are normalized to
a Gaussian distribution with a mean of 0 and a standard
deviation of 1. However, to ensure that the distribution of
most labels in the normal portion of the SWaT dataset is
not distorted or multiple spikes occur, we chose min-max
normalization, where the minimum and maximum values
of normalization are 0 and 1, respectively. The adoption of
min-max normalization can also give our model structure
generalizability for different types of data. Therefore, we nor-
malized the original data into standard data in 51 dimensions.
After normalization, the normal data in the dataset were used
for training, 80% of which were training data and 20% of
which were validation data, and the abnormal data of an
attack were used as test data.

E. SYSTEM ARCHITECTURE
For the model proposed in this paper, the first layer of the
1D_CNN network structure adopts kernel = 2 and 32 filters,
and the filter size of the next 1D convolutional layer is twice
that of the previous layer. GRU or LSTM networks with
depths of 3 and 100 internal hidden units were used. Since
the data in the SWaT dataset were recorded once per second,
we set the window length to T = 120 (that is, data are
collected within 2 minutes). To capture the relevant dynamics
of the data, the window (T = 120) was applied to the normal
dataset and the test dataset, where the slide length on the
normal dataset window was 10 and the slide length on the
test dataset was 120.

F. TRAINING
To avoid the occasional local minimum problem in train-
ing the neural network, we conducted multiple independent
trainings on the neural network and selected the best result
from the multiple training results. Since the neural network
was trained by the stochastic gradient descent method and
was mostly initialized randomly, the results obtained from
each operation were different. We iterated the batch_size
of 4096 each time during model training. The experimental
results in this paper are all based on the optimal results

G. EXPERIMENTAL RESULTS AND ANALYSIS
Experiment 1: Monolayer LSTM and GRU were combined
with a monolayer 1D_CNN.
First, we analyzed the difference between the mono-
layer GRU and LSTM in model training after they were
combined with the 1D_CNN. The results are shown
in figure 6 and figure 7. These analyses allowed us to obtain
the loss function and accuracy curve of the two models in the
training stage on the training set and verification set.

The experiment shows that in the monolayer case, the loss
function of the model using LSTM is not significantly dif-
f erent from that using the GRU in the first few rounds of
training, which means that the model using LSTM is not

| TABLE 1. General information about datasets. |
|------------------------------------------------|
| Dataset Item | SWaT | KDDCUP99 |
|----------------|--------------------------|
| Variables      | 51           | 34        |
| Attacks        | 36           | 2         |
| Attack durations (mins) | 2-25 | NA        |
| Normal data    | 496800       | 562387    |
| Data with attacks | 449919 | 494021    |
significantly different from that using the GRU in the first few rounds of training. With the increase in the number of training iterations, LSTM and the GRU almost reached convergence at the same time. Therefore, for the model with the single-layer GRU and LSTM combined with the 1D_CNN, the difference in training is not very large, and the GRU can achieve the same training effect as LSTM under the condition of multiple trainings.

Experiment 2: Multilayer LSTM and GRU were combined with a single-layer 1D_CNN

Then, we increased the layers of the GRU and LSTM to three layers to analyze the impact of the change in the number of layers on the model training. The results are shown in figure 8 and figure 9.

The experiment shows that the training speed of the multilayer model is faster than that of the single-layer model and that the multilayer model converges faster. The models with increasing GRU and LSTM loss function values were similar under optimal conditions, but using the GRU model helped to speed up the decline of its loss function compared with using the LSTM model. In the fourth round, to achieve relative stability, using the LSTM model is a poor choice, while use of the GRU is beneficial, as using the LSTM model during practice will cause a small wave. In addition, in terms of the accuracy of model training, the model using the GRU achieved better stability during training than that using LSTM and can achieve higher accuracy.

Experiment 3: Multilayer LSTM and GRU were combined with a multilayer 1D_CNN.

After analyzing the multilayer GRU and LSTM, we tried to increase the number of layers of the 1D_CNN to analyze whether the increase in the number of layers of the convolutional layer would affect the above results. We also wanted to know whether increasing the number of 1D_CNN layers would have a significant positive effect on the model. The results are shown in figure 10 and figure 11.

The above experimental results show that increasing the number of layers of the 1D_CNN can indeed improve the model. Specifically, it can enhance the stability of the model, and the loss function of the model becomes gentler during training. However, compared with LSTM, the GRU is still gentler, and its fluctuation is not very large. Considering that multilayer networks are usually used when designing models, GRU networks are more suitable for this paper during the training phase.

Experiment 4: Selection of the 1D_CNN layers

As shown in the above experiment, choosing the multilayer 1D_CNN to build an anomaly detection model can yield better results than choosing the single-layer 1D_CNN. Therefore, we further determined how many layers of the 1D_CNN
should be set to build the detection model in this paper. The results are shown in figure 12 and figure 13.

When we adopt a 2-layer 1D_CNN, although the loss function declines more stably, the convergence effect of the model is significantly worse than that of a 4-layer 1D_CNN and 8-layer 1D_CNN. Then, it can be seen that the loss function and accuracy of the model with the 8-layer 1D_CNN achieves the best results, but the stability of the model is worse than that of the model with the 4-layer 1D_CNN, and the loss function and accuracy fluctuate significantly. In the training phase, the performance of using the model with the 4-layer 1D_CNN is not much different from that of the 8-layer 1D_CNN. Therefore, this paper used a 4-layer 1D_CNN to build the abnormal state detection model of an industrial control system.

**Experiment 5: Experimental results for different types of networks**

We tested the current popular industrial control system anomaly detection methods based on the CNN network on the SWaT dataset and compared the time spent by different models in training and testing and separately on corresponding datasets for different processing stages. Tests were performed to compare different models to detect the F1 scores that occurred during different stages of processing on SWaT.

Looking at figure 14 and figure 15, we can see that the model using the 1D_CNN combined with the GRU requires more training time and testing time than does the 1D_CNN model alone. However, the testing times are not very different. The testing time of the model using the 4-layer 1D_CNN combined with a 3-layer GRU is less than 20 minutes longer than that of the model using the 4-layer 1D_CNN alone.
TABLE 2. F1 scores for different models at each stage of the process.

| Stage | P1 | P2 | P3 | P4 | P5 |
|-------|----|----|----|----|----|
| 2-layer convolution | 0.798 | 0.469 | 0.852 | 0.871 | 0.653 |
| 4-layer convolution | 0.887 | 0.578 | 0.854 | 0.903 | 0.729 |
| 8-layer convolution | 0.923 | 0.591 | 0.92 | 0.9 | 0.796 |
| 4-layer convolution + 3 GRU layers | 0.87 | 0.678 | 0.961 | 0.91 | 0.805 |

TABLE 3. Comparison of attack detection effects.

| Methods | Precision | Recall | F1 | Testing Time (s) |
|---------|-----------|--------|----|-----------------|
| SVM     | 0.92603   | 0.69894| 0.79662 | - |
| DNN     | 0.98325   | 0.68102| 0.80469 | - |
| TABOR [33] | 0.86832   | 0.79011| 0.82737 | - |
| MLP [34] | 0.96802   | 0.70214| 0.81392 | - |
| Sequence-to-Sequence Neural Networks [35] | 0.97613 | 0.79218 | 0.87459 | - |
| SDA+1D CNN+LSTM | 0.98863 | 0.84931 | 0.91369 | 5457 |
| SDA+1D CNN+GRU | 0.99657 | 0.85346 | 0.91948 | 4812 |

Experiment 6: Results of the model in the testing stage

To test the model proposed in this paper, we calculated the precision, recall rate and F1 score of the model when it was applied to the attack detection of industrial control systems and compared it with several traditional abnormal state detection methods of industrial control systems. The results are shown in Table 3.

The experiment shows that the model based on the SDA and 1D_CNN has a great improvement in precision and recall rate compared with the traditional SVM, the precision up to 99.6% with the recall rate of 85.3%; thus, the F1 score is also increased by 11.7% compared with the SVM. Compared with the model based on the DNN, the precision of the model based on the SDA and 1D_CNN is also improved, the precision is increased by 1.3% and the recall rate is also much better than that of the model based on the DNN, the recall is increased by 17%, so the F1 score of the model is nearly 11% different. For the TABOR and the MLP, there is always an improvement and a decrease in precision and recall, so the F1 score of them is not improved obvious. Compared with the sequence-to-sequence neural networks proposed in [35], the precision, recall rate and F1 score increases by at least 2%, 6% and 4% respectively. In addition, based on the model with the SDA and 1D_CNN, the GRU is adopted to predict the traffic, and compared with LSTM, the precision and recall rate are improved. In terms of time consumption, the model using the GRU is also superior to the model using LSTM.

We also estimate the AUC of our anomaly detection method according to each record detection metrics. The reason for using each record label for the AUC is to define the number of negative instances, which is problematic in the case of each attack detection (it is impossible to measure how many non-attacks there are). It can be seen from Figure 16 that the AUC of the anomaly detection method is high which reaches 0.976.

Table 4 shows that the detection performances of the models based on the SVM, DNN, TABOR or MLP for some 2-layer and 4-layer 1D_CNN. However, in the P2, P3, P4 and P5 stages, the F1 scores are lower than those in the model combining the 4-layer 1D_CNN and 3-layer GRU, and the detection effect is not as good as that of the model combining the 4-layer 1D_CNN and 3-layer GRU. Only in the P1 stage the performance is better than that of the model combining the 4-layer 1D_CNN and 3-layer GRU. As for the attacks in the P2 stage, there are only four such attacks numbers 6, 24, 29 and 30 in the dataset. The attack numbers 29 that did not achieve the desired effect and another attack numbers 24 that did not have the expected impact according to the description. Thus we can conclude that the relatively low F1 score for the P2 stage is caused by the test setup and our model successfully detected the attacks which were actually carried out.
### TABLE 4. Recall rate of each attack.

| No. | No. Scenario | Description of Attack | SVM | DNN | TABOR [33] | MLP [34] | StoS [35] | Ours |
|-----|--------------|-----------------------|-----|-----|------------|----------|----------|------|
| 1   | 1            | Open MV101            | 0   | 0   | 0.050      | 0        | 0        | 0.999| 0.999|
| 2   | 2            | Turn on P102          | 0   | 0   | 0.931      | 0.765    | 0.999    | 0.999|
| 3   | 3            | Increase LIT101 by 1 mm every second | 0   | 0   | 0         | 0.042    | 0.653    | 0.910|
| 4   | 4            | Open MV504            | 0.034 | 0   | 0.328     | 0        | 0        | 0     |
| 5   | 6            | Set value of FIT401 as 6 | 0.721 | 0.715 | 0.996   | 0.953 | 0.999 | 0.999|
| 6   | 7            | Increase water level LIT301 above HH | 0.889 | 0   | 0         | 0.911    | 0.999    | 0.999|
| 7   | 8            | Set value of DPIT as <40 kPa | 0.92 | 0.926 | 0.612    | 0.986    | 0.999    | 0.999|
| 8   | 10           | Set value of FIT401 as >0.7 | 0.435 | 0.999 | 0.994   | 0.979    | 0.999    | 0.999|
| 9   | 11           | Set value of FIT401 as 0 | 0.999 | 0.979 | 0.998    | 0.992    | 0.999    | 0.999|
| 10  | 13           | Close MV304           | 0   | 0   | 0         | 0        | 0        | 0     |
| 11  | 14           | Do not let MV303 open | 0   | 0   | 0         | 0        | 0        | 0     |
| 12  | 16           | Decrease water level LIT301 by 1 mm each second | 0   | 0   | 0.606     | 0.999    | 0.999    | 0.999|
| 13  | 17           | Do not let MV303 open | 0   | 0   | 0.597     | 0        | 0        | 0.999| 0.999|
| 14  | 19           | Set value of AIT504 to 16 uS/cm | 0.132 | 0.124 | 0.004   | 0.973    | 0.152   | 0.999|
| 15  | 20           | Set value of AIT504 to 255 uS/cm | 0.848 | 0.844 | 0.997   | 0        | 0.999    | 0.999|
| 16  | 21           | Keep MV101 on continuously; set value of LIT101 as 700 mm | 0.017 | 0   | 0.085     | 0.981    | 0.347   | 0.999|
| 17  | 22           | Stop UV401; set value of AIT502 as 150; force P501 to remain on | 0.999 | 0.997 | 0.998    | 0.978    | 0.999    | 0.999|
| 18  | 23           | Set value of DPIT301 to >0.4 bar; keep MV302 open; keep P602 closed | 0.875 | 0.867 | 0        | 0.713    | 0.999   | 0.999|
| 19  | 24           | Turn off P203 and P205 | 0   | 0   | 0         | 0.921    | 0.285   | 0.999|
| 20  | 25           | Set value of LIT401 as 1000; keep P402 on | 0.01 | 0   | 0        | 0.294    | 0.201   | 0.211|
| 21  | 26           | Turn P101 on continuously; set value of LIT301 as 801 mm | 0   | 0   | 0.999     | 0.999    | 0.999    | 0.999|
| 22  | 27           | Keep P502 on continuously; set value of LIT401 as 600 mm until 1:26:01 | 0   | 0   | 0.196     | 0        | 0.999    | 0.999|
| 23  | 28           | Close P502 | 0.94 | 0.934 | 0.999    | 0.033    | 0.999   | 0.999|
| 24  | 29           | Turn on P201; turn on P203; turn on P205 | 0   | 0   | 0         | 0.875    | 0        | 0     |
| 25  | 30           | Turn P101 on continuously; turn MV101 on continuously; set value of LIT101 as 700 mm; P102 starts itself because the level of LIT301 becomes low | 0.003 | 0   | 0.999     | 0.834    | 0.999    | 0.999|
| 26  | 31           | Set LIT401 to less than L | 0   | 0   | 0         | 0.786    | 0.501    | 0.208|
| 27  | 32           | Set LIT301 to above HH | 0.904 | 0   | 0        | 0.604    | 0.999    | 0.999|
| 28  | 33           | Set LIT101 to above H | 0   | 0   | 0.891     | 0.941    | 0.999    | 0.999|
| 29  | 34           | Turn P101 off | 0   | 0   | 0.992     | 0.808    | 0.999    | 0.999|
| 30  | 35           | Turn P101 off; keep P102 off | 0   | 0   | 0.261     | 0.842    | 0.294   | 0.201|
| 31  | 36           | Set LIT101 to less than LL | 0.118 | 0   | 0.901     | 0.821    | 0.999    | 0.999|
| 32  | 37           | Close P501; set value of FIT502 to 1.29 at 11:18:36 | 0.999 | 0.999 | 0.999    | 0.842    | 0.999    | 0.999|
| 33  | 38           | Set value of AIT402 as 260; set value of AIT502 to 260 | 0.928 | 0.924 | 0.997    | 0.784    | 0.999    | 0.999|
| 34  | 39           | Set value of FIT401 as 0.5; set value of AIT502 as 140 mV | 0   | 0.941 | 0.369     | 0.844    | 0.999    | 0.999|
| 35  | 40           | Set value of FIT401 as 0 | 0.928 | 0.932 | 0.998    | 0.801    | 0.999    | 0.999|
| 36  | 41           | Decrease LIT301 value by 0.5 mm per second | 0.356 | 0   | 0        | 0        | 0.999    | 0.999|
attacks (such as attack 1, attack 3, attack 12, etc.) are very poor, while our method achieves better performances for most attack detection tasks. The recall rates of our method and the method proposed in [35] are higher than other methods in detecting most attacks. But our method performs better in detecting some attacks, such as attack 3, attack 14, attack 16, and attack 19. The water level sensors (e.g. LIT-101, LIT-301 and LIT401) have the best anomaly detection performance. This is because other than the direct attacks on those sensors, most of attacks applied to the other sensors/actuators also affected the water levels indirectly. For the attack 14, due to the chemical accumulation in P5 is relevant for anomaly detection after the chlorine has been added to the system in P3 and P4, so the recall rate of the attack 14 is higher. For the attack 16, the target of attack 16 is MV101 and LIT101. For the attack 19, although the target is P203 and P205, it impacts the MV101 and LIT401. We can detect them through the MV101. In particular, a high recall rate is achieved for attacks with a low recall rate when using SVM- or DNN-based model detection (such as attack 1, attack 2, attack 3, etc.), thus improving the recall rate of multitype attack detection.

We discovered that only four attacks were consistently undetected: attack numbers 4, 10, 11 and 24. When reviewing the description of these tests we found that these attacks failed to have their expected impact on the system; thus, it is reasonable that they were undetected. For the attack 4, this attack opens MV-504 that does not exist in the dataset. The description in SWaT of attacks indicates that this attack has no impact. For the attack 10, this attack attempted to close MV-304 but MV-304 was closed later than when this attack occurred. In the SWaT dataset, MV-304 did not change. According to the description of the dataset, the attack 11 failed because Tank 301 was already full. For the attack 24, it is said that P-201, P-203, and P-205 did not start because of mechanical interlocks. In the dataset, nothing was changed around. Thus we can conclude that the low recall for some particular attacks (No. 4, 10, 11, 24) is caused by the test setup and our model detected the attacks which were actually carried out.

The SDA is used to achieve better learning and dimensionality reduction processing of the eigenvectors, and some features are eliminated. In addition, the 1D_CNN performs well in processing time series. Combined with LSTM or the GRU, features can be predicted more accurately. However, compared with the complex gate structure in the LSTM model, the structure of the GRU is simpler, achieving better results than LSTM in this paper. Therefore, the model based on the combination of the SDA, 1D_CNN and GRU can also be used in the abnormal state detection of industrial control systems and can achieve a better detection effect. Compared with the traditional methods, it greatly improves the precision and recall rate in the abnormal state detection of industrial control systems.

VI. CONCLUSIONS

This paper studied the use of autoencoders and neural networks to detect attack behaviors in an ICS and tested the effect of a series of model structure changes. To fully use the spatiotemporal correlations and other dependencies between multiple variables (sensors/actuators) in a system to detect anomalies, a model based on the SDA combined with the 1D_CNN and GRU was introduced, which was applied to the abnormal state detection of the SWaT dataset, successfully realized the detection of multiple types of attacks on the physical control level of the industrial control system, and achieved a higher precision and recall rate, thus improving the F1 score of the detection system. After using the SDA to conduct feature learning on the data, on the one hand, the data features were fully learned, and some irrelevant features were eliminated, and on the other hand, the eigenvectors originally obtained through the OHE were dimensionally reduced to avoid the dimension explosion caused by a large amount of data. To make the 1D_CNN fully learn the potential dependence between features, the eigenvector output by the SDA was extended through a fully connected layer. In the 1D_CNN structure, the 1D convolution layer was used to process each feature from the time dimension, and the features were predicted better by combining the 1D_CNN with the GRU.

Although the anomaly detection system for industrial control proposed in this paper achieves a good anomaly detection effect, the system proposed in this paper does not consider the situation in which the anomaly detection system may be deceived by an anti-attack. Therefore, this paper studies how to design an industrial control anomaly detection system that can not only achieve a better anomaly detection effect but also resist a higher degree of data disturbance and have a stronger optimization strategy to effectively resist an attack. This will be an interesting direction for future work. In addition, due to the use of the SDA, the number of model parameters was relatively large, which would result in training difficulty. Therefore, a better method or model for learning features could be found. For the optimization of data features, it was also possible to consider the relationship between networks in learning. For the structure obtained after learning the model, we could also try to interpret it more reasonably to design the model and rule-based detection more reasonably.
REFERENCES

[1] W. Yusheng, F. Kefeng, L. Yingxu, L. Zenghui, Z. Ruikang, Y. Xiangzhen, and L. Lin, “Intrusion detection of industrial control system based on Modbus TCP protocol,” in Proc. IEEE 13th Int. Symp. Auto. Decentralized Syst. (ISADS), Mar. 2017, pp. 156–162.

[2] D. Myers, K. Radke, and S. Suriadi, “Process discovery for industrial control system cyber attack detection,” in Proc. IFIP Int. Conf. IFT Systems Secur. Privacy Protection, 2017, pp. 61–75.

[3] T. M. Chen and S. Abu-Nimeh, “Lessons from stuxnet,” Computer, vol. 44, no. 4, pp. 91–93, Apr. 2011.

[4] Wired. (2016). Inside The Cunning, Unprecedented Hack Of Ukraine’s Power Grid. Accessed: Mar. 2018. [Online]. Available: https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/Last

[5] L. Zhongwei, T. Weiming, and J. Xianji, “Construction of cyber security defense hierarchy and cyber security testing system of Smart Grid: Thinking and enlightenment for network attack events to national power grid of Ukraine and Israel,” Autom. Electr. Power Syst., vol. 40, no. 8, pp. 147–151, 2016.

[26] D. Peng, Z. Liu, H. Wang, Y. Qin, and L. Jia, “A novel deeper one-dimensional CNN for intrusion detection in industrial control systems,” IEEE Trans. Ind. Inform., vol. 15, no. 8, pp. 2015–2030, Aug. 2018.

[27] N. Srivastava, G. Hinton, and A. Krizhevsky, “Dropout: A simple way to prevent neural networks from overfitting,” J. Mach. Learn. Res., vol. 15, no. 1, pp. 1929–1958, 2014.

[28] K. Cho, M. B. Van, and C. Gulcehre, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), pp. 1724–1734.

[29] M. Stone, “Cross-validatory choice and assessment of statistical predictions,” J. Roy. Stat. Soc. B, Methodol., vol. 36, no. 2, pp. 111–133, Jan. 1974.

[30] A. P. Mathur and N. O. Tippenhauw, “SWAT: A water treatment testbed for research and training on ICS security,” in Proc. Int. Workshop Cyber-Phys. Syst. Smart Water Netw. (CySWater), Apr. 2016, pp. 31–36.

[31] G. Jonathan, S. Adepu, K. N. Junejo, and A. Mathur, “A dataset to support research and training on ICS security,” in Proc. Int. Workshop Cyber-Phys. Syst. Smart Water Netw. (CySWater), Apr. 2016, pp. 603–608.

[32] D. Peng, Z. Liu, H. Wang, Y. Qin, and L. Jia, “A novel deeper one-dimensional CNN with residual learning for fault diagnosis of wheelset bearings in high-speed trains,” IEEE Access, vol. 7, pp. 10278–10293, 2019.

[33] N. Srivastava, G. Hinton, and A. Krizhevsky, “Dropout: A simple way to prevent neural networks from overfitting,” J. Mach. Learn. Res., vol. 15, no. 1, pp. 1929–1958, 2014.

[34] K. Cho, M. B. Van, and C. Gulcehre, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), pp. 1724–1734.

[35] M. Stone, “Cross-validatory choice and assessment of statistical predictions,” J. Roy. Stat. Soc. B, Methodol., vol. 36, no. 2, pp. 111–133, Jan. 1974.

[36] A. P. Mathur and N. O. Tippenhauw, “SWAT: A water treatment testbed for research and training on ICS security,” in Proc. Int. Workshop Cyber-Phys. Syst. Smart Water Netw. (CySWater), Apr. 2016, pp. 31–36.

[37] G. Jonathan, S. Adepu, K. N. Junejo, and A. Mathur, “A dataset to support research in the design of secure water treatment systems,” in Proc. Int. Conf. Crit. Inf. Infrastructures Secur. 2016, pp. 88–99.

[38] Google. (2016). TensorFlow, An Open-Source Machine Learning Framework for Everyone. [Online]. Available: https://www.tensorflow.org

[39] Q. Lin, S. Verwer, S. Adepu, and A. Mathur, “TABOR: A graphical model-based approach for anomaly detection in industrial control systems,” in Proc. ACM Asia Conf. Comput. Commun. Secur. (ASIACCS), 2018, pp. 525–536.

[40] D. Shalyaga, P. Filonov, and A. Lavrentyev, “Anomaly detection for water treatment system based on neural network with automatic architecture optimization,” 2018, arXiv:1807.07282. [Online]. Available: http://arxiv.org/abs/1807.07282

[41] J. Kim, J-H. Yun, and H. Chun Kim, “Anomaly detection for industrial control systems using Sequence-to-Sequence neural networks,” 2019, arXiv:1911.04831. [Online]. Available: http://arxiv.org/abs/1911.04831

VOLUME 8, 2020