An Intrusion Detection Method for Industrial Control System Based on Gate Recurrent Unit

Tusheng Chen, Peng Lin and Jie Ling

Faculty of Computer, Guangdong University of Technology, 100 West Ring Road, Higher Education Mega Centre, Panyu District, Guangzhou, Guangdong , China.
Email: cts930923@163.com

Abstract. We proposed an industrial control system intrusion detection method based on gate recurrent unit neural network to handle the problem that the intrusion detection method based on traditional machine learning algorithm such as SVM, decision tree, NN etc., cannot effectively deal with massive, high-dimensional, time related network traffic data in industrial control system. We used the update gate and the reset gate of GRU to save the information of the data in the time dimension. Its deep structure can fully learn the data features, and we used Adam algorithm to optimize the gradient training process of the neural network. Comparison experiments were conducted with the intrusion detection method based on machine learning algorithms such as SVM, decision tree, NN, RNN, LSTM etc. The results show that the proposed method has higher classification accuracy than SVM, decision tree, NN and RNN, and the accuracy is basically the same as LSTM but the training time is greatly reduced.

1. Introduction

In recent years, information security incidents in industrial control systems around the world have occurred frequently, causing very bad effects and serious consequences. For example, in 2010, the Stuxnet worm [1] infects more than 45,000 networks worldwide, and its targeted invasion of industrial control systems directly caused delayed power generation in Iranian nuclear power plants. The Duqu virus [2] appeared in 2011, its purpose was to lurk in the industrial control system to steal data information. Once the long-distance access capability is obtained, it will make future cyber-attacks easier. In 2012, the Flame virus [3] can receive remote commands from multiple places, and hackers can use this feature to remotely control the SCADA system, which is very harmful. Havex virus [4], which emerged in 2014, attacked industrial control systems through mail and phishing. Faced with the frequent occurrence of industrial control network security incidents, intrusion detection technology as an important defensive line of industrial control information security has become a hot research topic at home and abroad.

Traditional machine learning algorithms have been applied in intrusion detection and have achieved some results, but most of them belong to shallow learning. In real industrial environments, continuous working control systems generate large amounts of high-dimensional, nonlinear, and time related data. This makes the traditional shallow learning method have limitations [5]. For example, there is a lot of noise in the industrial control network data, the decision tree [6] is easily affected by the over-fitting and the classification accuracy is reduced. The support vector machine (SVM) [7]-[9] consumes a large amount of computational resources when processing large amounts of data. Artificial neural networks (NN) [10], [11] require many parameters to train and cannot represent the temporal relationship of data. Therefore, the traditional abnormal traffic detection method can not effectively solve the classification problem of a large number of complex data in industrial control networks. In
contrast, deep learning can fully learn data features through its deep structure, and has outstanding performance in massive high-dimensional data analysis. The deep learning concept first proposed by Hinton [12] in 2006 has made breakthroughs in the fields of speech recognition [13], image classification [14], and process modelling [15]. In recent years, some researchers have applied deep learning to the field of intrusion. Tang [16] proposed a deep neural network (DNN) model, which achieved better classification results using only six basic features in the NSL-KDD data set. Yin [5] applied RNN to classify network data sets into two-categories and multi-categories in intrusion detection, but standard RNNs have problems with gradient disappearance and insufficient memory for past time information. Yu [17] proposes an intrusion detection method based on long short term memory (LSTM), which improves problems of gradient disappearance and insufficient time memory of RNN, but LSTM requires more training parameters and time complexity is greater.

This paper presents an intrusion detection method for industrial control system based on gated recurrent unit neural network (GRU-IDS). It can fully learn the data characteristics through the deep structure of GRU, and use update gate and reset gate to preserve the information of data in time dimension. It solves the problems of gradient disappearance of RNN, and reduces the training parameters and time complexity compared with LSTM. And the gradient training process of the neural network is optimized by using the Adam algorithm. The GRU-IDS is validated by using the standard data set of supervisory control and data acquisition (SCADA) system proposed by the Key Infrastructure Protection Centre of Mississippi State University [18]. Due to the large difference of maximum values between different features in the original data set, normalization method is used to pre-process, and then the number of hidden layer nodes, training method and learning rate of GRU model are adjusted through many experiments to get the better model. Finally, the GRU-IDS is compared with traditional machine learning methods such as SVM [8], NN [10], DNN [16], and algorithms with time scale such as RNN [5] and LSTM [17]. The results show that the proposed GRU-IDS can deal with the intrusion detection problem of industrial control system more effectively.

2. Relevant Work

2.1. Intrusion Detection Based on Recurrent Neural Network

In [5], recurrent neural network (RNN) is applied to classify network data sets into two-categories and multi-categories in intrusion detection, and achieves higher classification accuracy than SVM, KNN, NN and random forest. RNN consists of input layer, hidden layer and output layer. But unlike feed-forward neural networks, its hidden layer output is not only related to the current input, but also affected by the hidden layer output of the previous step. Figure 1 is an expansive diagram of the recurrent neural network in time step. Among them, \( x \) is the input layer vector, \( U \) is the weight matrix from the input layer to the hidden layer, \( s \) is the hidden layer vector, \( W \) is the weight matrix from the previous hidden layer to the current hidden layer, \( V \) is the weight matrix from the hidden layer to the output layer and \( o \) is the output layer vector.

It can be seen from the graph that the value \( s_t \) of RNN hidden layer at time \( t \) not only depends on the current input \( x_t \), but also is affected by the value \( s_{t-1} \) of hidden layer at time \( t-1 \), so recurrent neural network can 'memory' previous input information. The following formulas can be used to represent the calculation process of recurrent neural network:

\[
o_t = g(V \ast s_t) \quad (1)
\]

\[
s_t = f(U \ast x_t + W \ast s_{t-1}) \quad (2)
\]

Formula (1) is the calculation formula of the output layer, which is a fully connected layer and \( g \) is an activation function. Formula (2) is the calculation formula of the hidden layer, which is a recurrent layer and \( f \) is an activation function.

2.2. Intrusion Detection Based on Long Short Term Memory

Because the hidden layer of the original RNN has only one state, it is very sensitive to short-term input and cannot handle long-distance dependencies. To solve this problem, Hochreiter and Schmidhuber
[19] proposed Long Short Term Memory (LSTM), which added cell state c to save the long-term state, as shown in Figure 2.

**Figure 1.** The network structure of RNN expanded

**Figure 2.** Hidden state comparison chart between RNN and LSTM

LSTM controls discarding or adding information through gate to realize the function of forgetting or memory. The gate is actually a fully connected layer consisting of a sigmoid function and a matrix point multiplication operation. Its input is a vector and its output is a real vector between 0 and 1. Among them, 0 represents total abandonment and 1 represents total adoption. An LSTM memory unit consists of three gates: forget gate, input gate and output gate. The forget gate determines how much of the state of the unit at the previous moment is reserved to the current moment; the input gate determines how much of the input at the current moment is saved to the state of the unit; and the output gate determines how much of the state of the cell unit is output to the final output value of the memory unit. This mechanism of LSTM keeps the error at a more constant level. When the error is transmitted backward along time and the upper layer, the problems of gradient disappearance or gradient explosion are avoided. Figure 3 shows the basic structure of LSTM memory unit.

**Figure 3.** Logic diagram of LSTM memory unit

In Figure 3, \(x_t\) is the input at time t, \(h_{t-1}\) and \(C_{t-1}\) are the output and cell states of the previous time step, respectively. \([\cdot, \cdot]\) is the matrix splicing operation, \(W_f, W_i, W_c, W_o\) are weight matrices, \(\sigma\)
is the sigmoid activation function, and the forget gate $f_t$, the input gate $i_t$, the output gate $o_t$, and the input candidate matrix $C'_t$ are respectively:

$$
f_t = \sigma(W_f \ast [h_{t-1}, x_t] + b_f) \quad (3)$$

$$
i_t = \sigma(W_i \ast [h_{t-1}, x_t] + b_i) \quad (4)$$

$$
C'_t = \tanh(W_c \ast [h_{t-1}, x_t] + b_c) \quad (5)$$

$$
o_t = \sigma(W_o \ast [h_{t-1}, x_t] + b_o) \quad (6)$$

Using the forget gate $f_t$ the input gate $i_t$ and the input candidate matrix $C'_t$, the cell state of the memory unit can be updated according to the following formula:

$$
C_t = f_t \cdot C_{t-1} + i_t \cdot C'_t \quad (7)
$$

Finally, the cell state $C_t$ is activated by the tanh function, and the output gate $o_t$ controls the degree of filtering of cell state $C_t$, thereby obtaining the final output $h_t$.

$$
h_t = o_t \cdot \tanh(C_t) \quad (8)$$

Literature [17] applied LSTM to industrial control system intrusion detection, and verified that the classification accuracy of LSTM is higher than that of RNN-based intrusion detection method on industrial control standard data set.

### 3. Proposed Methodologies

#### 3.1. Data Preprocessing

Because different features in the original industrial control standard data set have different dimensions and range of values, in order to eliminate the dimension effect between features, it is necessary to normalize the data. In this paper, Min-Max standardization is used to transform the original data linearly so that the result values are mapped to [0, 1]. The conversion function is as follows:

$$
x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (9)
$$

$max(x)$ and $min(x)$ are the maximum and minimum values of sample features respectively.

#### 3.2. Intrusion Detection Algorithms Based on GRU

GRU is an improvement of LSTM. It keeps the effect of LSTM and simplifies the structure, reduces the number of training parameters and reduces the time complexity. The memory module of LSTM has a complex structure. It implements three gates: forget gate, input gate and output gate. GRU only uses two gates, update gate and reset gate, which are the $z_t$ and $r_t$ of Figure 4. The update gate is used to control the extent to which the previous state information is brought into the current state, and the reset gate is used to control the degree of ignoring the state information of the previous time step.
Figure 4. Logic diagram of GRU memory unit

In Figure 4, $x_t$ is the input of time t and $h_{t-1}$ is the output of the previous time step. The iterative formula of output $h_t$ of memory unit can be obtained as follows:

$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$  \hspace{1cm} (10)
$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$  \hspace{1cm} (11)
$h'_t = tanh(W_{h'} \cdot [r_t \cdot h_{t-1}, x_t])$  \hspace{1cm} (12)
$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot h'_t$  \hspace{1cm} (13)

Among them, $W_r, W_z, W_{h'}$ are weight matrix, $z_t$ and $r_t$ are output of update gate and reset gate respectively, and $h'_t$ is input candidate matrix.

In addition, in order to speed up the training of weight matrix, the gradient training algorithm uses the Adam algorithm in [20]. Combining the advantages of AdaGrad [21] and RMSProp [22], this algorithm takes into account the first Moment Estimation (mean of gradient) and the second Moment Estimation (non-centralized variance of gradient). It has the following remarkable advantages:

- Simple implementation, high computational efficiency, and less memory requirement;
- Parameter update is not affected by gradient scaling;
- It’s suitable for scenarios of large-scale data and parameter;
- It’s suitable for gradient sparse or gradient with large noise.

The principle of the algorithm can be expressed as follows:

$g_t = \nabla J(W_{t-1})$  \hspace{1cm} (14)
$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$  \hspace{1cm} (15)
$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$  \hspace{1cm} (16)
$m'_t = \frac{m_t}{1 - \beta_1^t}$  \hspace{1cm} (17)
$v'_t = \frac{v_t}{1 - \beta_2^t}$  \hspace{1cm} (18)
$W_{t+1} = W_t - \frac{\alpha}{\sqrt{v'_t} + \epsilon} \cdot m'_t$  \hspace{1cm} (19)

Where $m_t$ and $v_t$ are first-order momentum terms and second-order momentum terms respectively, $\beta_1$ and $\beta_2$ are the exponential decay rates of first-order moment estimates and second-order moment estimates respectively, which are generally 0.9 and 0.999 respectively, $m'_t$ and $v'_t$ is the respective bias-corrected value. $W_t$ represents the weight matrix of the time-step t. $g_t$ represents the gradient of
cost function at time step t; \( \alpha \) is the learning rate, and \( \epsilon \) is a small value (generally 1e-8), the purpose is to avoid the denominator is 0.

The whole process of industrial control intrusion detection algorithm based on GRU is shown in Figure 5.

![Figure 5. Flowchart of intrusion detection mode based on GRU](image)

### 3.3 Simulation parameters and evaluation indexes

The experimental data consisted of 4000 training data and 1000 test data. The classification effect was evaluated in the sense of 5 fold cross validation. This experiment uses a Python-based deep learning framework Keras, simulation platform: Win10 64 bit operating system, Intel i3-3120M CPU 2.50GHz, 8GB memory, without GPU acceleration. Experimental parameters: input vector dimension is 26, output vector dimension is 8, time series length T is 200, hidden layer is 1 layer, gradient training algorithm uses Adam algorithm, maximum number of iterations is 100, sample number per batch is 200 and initial value of learning rate \( \alpha \) is 0.001. In each epoch, formula (20) is used to update learning rate, exponential attenuation rates of first-order moment estimation and second-order moment estimation are 0.99 and 0.999, respectively. In order to obtain the optimal model, the number of hidden layer units is selected 20, 50, 80 and 100 to carry out experiments respectively.

\[
\alpha = \alpha / \sqrt{\text{epoch}} \quad (20)
\]

For the industrial control intrusion detection algorithm, the typical evaluation indicators are the accuracy rate ACC, the recall rate TPR and the false positive rate FPR. Table 2 shows the confusion matrix, where TP indicates the number of samples correctly identified as the attack class, FP is the number of samples misidentified as the attack class, TN is the number of samples correctly identified as normal, and FN is the number of samples misidentified as normal. The definition of accuracy ACC, recall rate TPR and false positive rate FPR can be obtained from the confusion matrix.

\[
\text{ACC} = \frac{TP+TN}{TP+FP+TN+FN} \quad (21)
\]
\[
\text{TPR} = \frac{TP}{TP+FN} \quad (22)
\]
\[
\text{FPR} = \frac{FP}{FP+TN} \quad (23)
\]
Table 1. Data Description

| Attack categories | Tag | Attack description                        |
|-------------------|-----|-------------------------------------------|
| Normal            | 0   | Instance not part of an attack            |
| NMRI              | 1   | Naive malicious response injection attack |
| CMRI              | 2   | Complex malicious response injection attack|
| MSCI              | 3   | Malicious state command injection attack  |
| MPCI              | 4   | Malicious parameter command injection attack|
| MFCI              | 5   | Malicious function command injection attack|
| DOS               | 6   | Denial-of-service attack                  |
| Reconnaissance    | 7   | Reconnaissance attack                     |

3.4. Experimental Results and Analysis

3.4.1. GRU-IDS model. In order to determine the number of nodes in the hidden layer of the GRU neural network, the accuracy of the model, the recall rate, the false positive rate and the training time are calculated through multiple sets of comparison experiments. Parameters of best classification effect are the final parameters of the GRU neural network. Table 3 shows the experimental results, so when the hidden layer node is selected as 80; the classification accuracy is relatively optimal.

Table 2. Confusion matrix

| Actual Class | Predicted Class |
|--------------|-----------------|
|              | Attack | Normal |
| Attack       | TP     | FN     |
| Normal       | FP     | TN     |

Table 3. Effect of different hidden nodes

| Hidden Nodes | ACC/% | TPR/% | FPR/% | Training Time/s |
|--------------|-------|-------|-------|-----------------|
| 20           | 95.38 | 95.44 | 5.13  | 10.63           |
| 50           | 96.64 | 96.28 | 4.45  | 13.24           |
| 80           | 97.87 | 96.84 | 3.68  | 15.53           |
| 100          | 96.50 | 97.12 | 3.52  | 19.06           |

In order to verify the superiority of GRU-IDS in the field of industrial control system intrusion detection, this paper compares with RNN and LSTM which have time scale. The number of iterations of RNN and LSTM and the number of samples per batch are the same as GRU. Other parameters are adjusted to relative optimum through multiple combinatorial comparison experiments. In addition, in order to more comprehensively investigate the effect of the proposed algorithm in solving intrusion detection problems of industrial control system, the traditional machine learning algorithms which don’t have time scale, such as SVM, decision tree C4.5, ANN and DNN, are also simulated. The comparison is made under the accuracy rate, recall rate, false positive rate and training time. The results are shown in Table 4.
Table 4. Classification results of different algorithms

| Classification         | Algorithm | ACC/% | TPR/% | FPR/% | Training Time /s |
|------------------------|-----------|-------|-------|-------|------------------|
| Algorithms with Time Scale | GRU       | 97.85 | 96.73 | 2.18  | 14.09            |
|                        | LSTM      | 97.87 | 96.75 | 2.17  | 18.84            |
|                        | RNN       | 95.34 | 96.08 | 2.65  | 12.43            |
| Algorithms without Time Scale | C4.5      | 91.64 | 94.64 | 2.85  | 0.04             |
|                        | SVM       | 90.83 | 91.30 | 3.64  | 1.20             |
|                        | NN        | 94.98 | 95.60 | 2.82  | 8.30             |
|                        | DNN       | 97.50 | 96.18 | 2.04  | 14.75            |

As can be seen from Table 4, the accuracy of GRU and LSTM is almost the same, but the training time of GRU is significantly less than that of LSTM. This is because the number of gates of GRU is less than that of LSTM, which reduces matrix multiplication. When the training set is large, GRU will save a lot of time. In addition, the accuracy of GRU is significantly higher than that of RNN, because when the time step is large, there is a problem of gradient disappearance or gradient explosion of RNN, which results in limited time memory ability of RNN. At the same time, it can be seen from Table 4 that GRU has the best effect compared with the algorithm which don’t have time scale, and its accuracy is significantly larger than traditional shallow machine learning algorithms such as C4.5, SVM, NN and so on. Recall rates of C4.5, SVM and NN are relatively low, and their recognition abilities of attack type data is relatively poor. DNN is a traditional deep neural network with an accuracy rate close to GRU, but its recognition ability for attack type data is relatively poor, and the recall rate is significantly smaller than that of GRU.

The accuracy of the above algorithm for various categories of data is shown in Figure 6. It can be seen that the accuracy of GRU in various categories of data is high, especially the classification effect of DOS and NMRI is significantly better than other algorithms. All the algorithms have a poor detection effect on MFCI, but the classification accuracy of normal data, MPCI and RECO is very high
4. Conclusions
Faced with large-scale, high-latitude, and time related network traffic data in industrial control systems, traditional machine learning algorithms such as decision trees C4.5, SVM, NN, etc. cannot effectively extract the feature information of data. Therefore, an industrial control intrusion method based on GRU is proposed in this paper, the deep structure of GRU neural network is used to fully study the data features, and the update gate and reset gate are used to save the information of the data in the time dimension, which solves the problem of limited time memory ability of RNN. At the same time, compared with LSTM, the memory cell structure is simplified, the calculation amount is reduced while the effect is maintained, the training time is reduced. And the gradient training process of the neural network is optimized by using the Adam algorithm. Finally, a relatively optimal model is constructed through experiments, comparing it with RNN and LSTM which have time scale, and traditional machine learning algorithms such as LSTM, RNN and C4.5, SVM, NN and DNN etc., the overall detection effect (accuracy, recall rate, false positive rate, average training time) and the detection effect of different types of data are analysed. The results show that GRU-IDS has the best effect, and the overall classification accuracy is significantly larger than RNN and traditional shallow machine learning algorithms such as C4.5, SVM and NN. The accuracy of GRU-IDS is basically the same as that of LSTM but the training time is greatly reduced. In addition, GRU-IDS has higher classification accuracy on various types of data, especially the classification effect of DOS and NMRI is obviously better than other algorithms. Therefore, the intrusion detection method based on GRU is very suitable for the intrusion detection problem of massive high-dimensional and time related data of industrial control system; we provide a new method for intrusion detection of industrial control systems.

Figure 6. Detection results of normal data and various type of attack data
5. Acknowledgement

This work is supported by the science and technology project of Guangdong Province (No.2017B090906003) and the project of Guangzhou Science and Technology (No.201802010043, 201807010058).

6. References

[1] Farwell, J P and Rohozinski, R 2011 Stuxnet and the future of cyber war Survival 53(1) pp 23-40
[2] Bencsáth, B, Pék G, Buttyán, L and Félegyházi, M 2012 April Duqu: Analysis, detection, and lessons learned ACM European Workshop on System Security (EuroSec) Vol. 2012
[3] Naughton, J 2012 How the Flame virus has changed everything for online security firms The Guardian.
[4] Constantin, L 2014 New Havex malware variants target industrial control system and SCADA users PC World
[5] Yin, C, Zhu, Y, Fei, J and He, X 2017 A deep learning approach for intrusion detection using recurrent neural networks IEEE Access 5 pp 21954-61
[6] Beaver, J M, Borges-Hink, R C and Buckner, M A 2013 December An evaluation of machine learning methods to detect malicious SCADA communications Machine Learning and Applications (ICMLA), 2013 12th International Conference on IEEE Vol. 2 pp 54-59
[7] Kuang, F, Xu, W and Zhang, S 2014 Applied Soft Computing 18 pp 178-184
[8] Reddy, R R, Ramadevi, Y and Sunitha, K N 2016, September Effective discriminant function for intrusion detection using SVM Advances in Computing, Communications and Informatics (ICACCI), 2016 International Conference on IEEE pp 1148-53
[9] Maglaras, L A, Jiang, J and Cruz, T 2014 Electronics Letters 50(25) pp 1935-36
[10] Ingre, B and Yadav, A 2015, January Performance analysis of NSL-KDD dataset using ANN Signal Processing And Communication Engineering Systems (SPACES), 2015 International Conference on IEEE pp 92-96
[11] Shitharth, S 2017 Computers & Security 70 pp 16-26
[12] Hinton, G E and Salakhutdinov, R R 2006 Reducing the dimensionality of data with neural networks science 313(5786) pp 504-507
[13] Chen, X, Liu, X, Wang, Y, Gales, M J and Woodland, P C 2016 Efficient training and evaluation of recurrent neural network language models for automatic speech recognition. IEEE/ACM Transactions on Audio, Speech, and Language Processing 24(11) pp 2146-57
[14] Yu, S, Jia, S and Xu, C 2017 Convolutional neural networks for hyperspectral image classification Neurocomputing 219 pp 88-98
[15] Li J and Sang H 2017 Elman recurrent neural network method based on SCKF algorithm and its application to soft sensor modelling Information and Control 46(3) pp 342-349
[16] Tang, T A, Mhamdi, L, McLernon, D, Zaidi, S A R. and Ghogho, M 2016, October Deep learning approach for network intrusion detection in software defined networking Wireless Networks and Mobile Communications (WINCOM), 2016 International Conference on IEEE pp 258-263
[17] Yu, B B, Wang H Z and Yan C Y 2018 Intrusion Detection of Industrial Control System Based on Long Short Term Memory Information and Control 47(1) pp 54-59
[18] Morris, T and Gao, W 2014 March Industrial control system traffic data sets for intrusion detection research International Conference on Critical Infrastructure Protection Springer Berlin Heidelberg pp 65-78
[19] Hochreiter, S and Schmidhuber, J 1997 Long short-term memory Neural computation 9(8) pp 1735-80
[20] Kingma, D P and Ba, J 2014 Adam: A method for stochastic optimization. arXiv preprint arXiv 1412 p 6980
[21] Hagedu, A T, Nigam, A and Diaz-Aviles, E. 2015, October Large-scale learning with AdaGrad on Spark Big Data(Big Data), 2015 IEEE International Conference on IEEE pp 2828-30
[22] Tieleman, T and Hinton, G 2012 Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude COURSERA: Neural networks for machine learning 4(2) pp 26-31