IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 21, 2018, accepted August 8, 2018, date of publication August 28, 2018, date of current version September 21, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2867564

An Intrusion Detection System Using a Deep Neural Network With Gated Recurrent Units

CONGYUAN XU, (Student Member, IEEE), JIZHONG SHEN, XIN DU, AND FAN ZHANG, (Member, IEEE)

College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China

Corresponding author: Jizhong Shen (jzshen@zju.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61471314 and Grant 61432357, in part by the Welfare Technology Research Project of Zhejiang Province under Grant LGG18F010003, and in part by the China Scholarship Council under Grant CSC201606325012.

ABSTRACT To improve the performance of network intrusion detection systems (IDS), we applied deep learning theory to intrusion detection and developed a deep network model with automatic feature extraction. In this paper, we consider the characteristics of the time-related intrusion and propose a novel IDS that consists of a recurrent neural network with gated recurrent units (GRU), multilayer perceptron (MLP), and softmax module. Experiments on the well-known KDD 99 and NSL-KDD data sets show that the system has leading performance. The overall detection rate was 99.42% using KDD 99 and 99.31% using NSL-KDD with false positive rates as low as 0.05% and 0.84%, respectively. In particular, for detecting the denial of service attacks, the system achieved detection rates of 99.98% and 99.55%, respectively. Comparative experiments showed that the GRU is more suitable as a memory unit for IDS than LSTM, and proved that it is an effective simplification and improvement of LSTM. Moreover, the bidirectional GRU can reach the best performance compared with the recently published methods.

INDEX TERMS Intrusion detection, deep learning, recurrent neural network, gated recurrent unit.

I. INTRODUCTION

As computer networks have become widely used in all aspects of our lives, network security has become increasingly important. Network security includes mainly the confidentiality, integrity and availability (CIA) of its bearer's information. Any activity attempting to compromise CIA or to bypass the security mechanisms of a network can be considered a network intrusion [1]. An intrusion detection system (IDS) is a type of security management system used to detect intrusion on the network, and is an indispensable part of network security systems nowadays [2]. An IDS usually checks all incoming and outgoing packets of a particular network to determine whether each packet has signs of an intrusion. A well-designed IDS can identify the characteristics of most intrusion activities and automatically respond to them by writing to security logs or issuing warnings.

IDSs can be divided into two categories according to the main detection technology: misuse detection and anomaly detection [3]. Misuse detection is a knowledge-based detection technology. A misuse detection system needs to clearly define the features of the intrusion, then identify the intrusion by matching the rules. Misuse detection can achieve a high accuracy and low false alarm rate. However, it needs to build a feature library and cannot detect unknown attacks. In contrast, anomaly detection is a behavior-based detection technology. First, it needs to define the normal activities of a network, and then check whether the actual behavior has deviated from the normal activities. Anomaly detection needs only to define a normal state of a specific network, without prior knowledge of intrusion. Thus, it can detect unknown attacks, although there may be a high false alarm rate. At present, network structure is becoming more and more complicated, and intrusion methods are following the trend of diversification and complication, creating more challenges for IDSs.

Many studies on machine learning have developed intrusion detection technologies with machine intelligence. For instance, support vector machine (SVM), artificial neural networks (ANNs), and genetic algorithms (GAs) have achieved good results in the field of intrusion detection. However, the simple machine learning method suffers from many limitations, while intrusion is becoming more complicated and diversified. Better learning methods are needed, especially in the automatic extraction of intrusion features and analysis.

After its introduction by Hinton *et al.* [4], deep learning has been widely studied and has achieved great success in natural language processing, image recognition and weather forecasting. The models involved in deep learning have a high degree of non-linear structure which shows outstanding learning ability for the processing of complex data. The rapid development of parallel computing in recent years also has provided a hardware foundation for deep learning algorithms.

The recurrent neural network (RNN) has failed to become a mainstream network model in the past few years due to difficulties in training and computational complexity. In recent years, with the development of deep learning theory, RNN began to enter a rapid development period. Currently, RNN has already been applied successfully to handwriting [5] and speech recognition [6]. The main feature of RNN is that it circulates information in a hidden layer which can remember information processed previously, leading to a structural advantage for the processing of time series information. Correspondingly, many intrusion behaviors can be abstracted as specific time series of events from the underlying network. So, RNN is considered suitable for building an IDS.

To improve the learning ability of IDS and improve its detection performance, we propose an IDS model based on a deep neural network. Specifically, we propose a novel IDS that consists of a recurrent neural network with gated recurrent units (GRUs), multilayer perceptron and softmax module. The main contributions of this paper are as follows:

1) A new deep neural network based IDS is proposed which uses gated recurrent units as the main memory units, combined with a multilayer perceptron to identify network intrusion.
2) The proposed system was evaluated in detail on the KDD 99 and NSL-KDD datasets. These two datasets have been widely used in previous work, giving us the opportunity to directly compare performance. The experimental results showed that GRU is more suitable as the memory unit of RNN than long short term memory (LSTM) in intrusion detection.
3) With the help of the deep neural network, the proposed system does not require manual feature selection. It significantly reduces the workload of network experts and has a positive outcome in today's ever-changing network environment.

The rest of this paper is organized as follows. Section II outlines related studies. Section III introduces the components which were used to construct the proposed system and the overall architecture. Section IV presents the experimental details and results. Sections V and VI discuss our results and draw conclusions, respectively.

II. RELATED WORKS

Since Denning proposed the first intrusion detection model [7], scholars have applied a variety of methods for intrusion detection. In recent years, methods ranging from relatively simple statistical methods to advanced machine learning and data mining methods have been used in attempts to extract specific patterns from network intrusions. Thereby, we can distinguish attack traffic from normal traffic, and finally build IDSs.

The use of a single machine learning algorithm has inherent limitations. In recent years, different learning algorithms have been combined for a better performance.

Ibrahim *et al.* [8] used an unsupervised ANN to construct an IDS based on anomaly detection. The system employed self-organization map (SOM) ANNs for detection and to distinguish attack traffic from normal traffic. The detection rate can reach 92.37% on the KDD 99 dataset and 75.49% on the NSL-KDD dataset. SOMs are more powerful than static networks because dynamic networks have memory, which can be trained to learn sequential or time-varying patterns. A classifier called GPSVM based on SVM and genetic programming (GP) was proposed by Pozi *et al.* [9] to improve the rare attack detection rate. Experimental results showed GPSVM can produce a more balanced classification accuracy on the NSL-KDD dataset without the need for resampling or feature selection techniques. A hybrid method combining SVM and genetic algorithm (GA) was proposed by Aslahi-Shahri *et al.* [10]. The hybrid algorithm was used to reduce the number of features from 45 to 10, and the GA algorithm assigned these features into three priorities. As a result, it showed an outstanding true positive value and low false positive value on the KDD 99 dataset. Hussain *et al.* [11] proposed a two-stage hybrid classification method. In the first stage, SVM was used for anomaly detection, while in the second stage, ANN was used for misuse detection. The main idea was to combine the advantages of each method to improve classification accuracy. Simulation results based on the NSL-KDD dataset demonstrated that this method outperforms individual classification of SVM and ANN algorithms. Bamakan *et al.* [12] proposed a time-varying chaos particle swarm optimization (TVCPSO) to set parameters and select features simultaneously for multiple criteria linear programming (MCLP) and SVM. Empirical results showed that this method achieves a high detection rate and a low false alarm rate. Feng *et al.* [13] proposed an SVM method with clustering based on self-organized ant colony network (CSOACN) to combine the advantages of both methods and obtained a better classification rate and run-time efficiency. Evaluation on the KDD 99 dataset showed this algorithm outperforms SVM alone or CSOACN alone in terms of both classification accuracy and run-time efficiency. Aburomman and Reaz [14] proposed an ensemble construction method which combines opinions from several experts into one model. Weighted majority voting (WMV) was used to improve accuracy and the best results were obtained with particle swarm optimization (PSO). However, such classifiers are based on binary classification methods which can distinguish between only two states. An ensemble method based on bat algorithm (BA) was proposed in [15] which applied an ELM as the base classifier and the BA to optimize the original ensemble, then applying it to intrusion detection.

Although the performance of the ELM is unstable, the method combining different ELMs into ensembles achieved better performance than using a single ELM.

From the above work we conclude that the idea of combining different learning algorithms has achieved better results in improving the performance of the classifier. However, the classic classification algorithms need to preprocess the data and extract the features manually, and cannot adjust the parameters autonomously. To complete the goal of learning and classification, experts are needed to be involved from time to time.

As a new hotspot in the study of neural networks, deep learning has attracted much attention in academia and industry as it has significant advantages in many areas. In the intrusion detection area, deep learning has achieved some good results.

Ma *et al.* [16] adopted spectral clustering (SC) to extract the features from the network traffic and used a multilayer deep neural network (DNN) to detect attack types. Experimental results showed that SC-DNN performs better than SVM, back propagation neural network (BPNN), random forest (RF) and Bayesian methods, with the best accuracy rates. However, the weight parameters and thresholds of each DNN layer need to be determined empirically rather than through rigorous mathematical theory. Kang and Kang [17] proposed an efficient IDS based on DNN for in-vehicle networks. The system uses DNN to provide the probability of each class discriminating normal and attack packets in a controller area network (CAN) bus. To take advantage of deep learning, the system initializes the parameters through pre-training deep belief networks (DBN), resulting in an improvement in detection accuracy. Erfani *et al.* [18] proposed a hybrid model that coupled a deep belief network (DBN) with a one-class SVM. An unsupervised DBN was trained to extract generic underlying features, while a one-class SVM was trained from the features learned by the DBN. This model provided an efficient, accurate and scalable anomaly detection approach that is suitable for large-scale and high-dimensional domains. A deep learning technique called self-taught learning (STL) was used by Javaid *et al.* [19] to build a network intrusion detection system (NIDS). Sparse autoencoder and softmax regression based NIDS were implemented. Experiments on the NSL-KDD dataset showed that the performance of STL was comparable to the best results achieved in several previous studies. Staudemeyer [20] claimed that, for the first time, a long short term memory (LSTM) recurrent neural network had been applied to intrusion detection, and he confirmed its effectiveness. LSTM can learn to look back in time and discover associations from a time perspective. The proposed classifier was effective in the detection of denial of service (DOS) attacks and network probes, both of which had a distinctive time series of events. Experiments showed that LSTM outperformed the winning entries of the KDD Cup 99 challenge, with 93.82% accuracy. Kim *et al.* [21] also used the LSTM structure to construct an IDS model trained with the KDD 99 dataset. Experimental results showed that the LSTM structure is effective for intrusion detection, and a feasible approach to reduce the false positive rate was proposed. This provided further evidence that the LSTM structure is suitable for intrusion detection. Mohammed *et al.* [22] proposed a developed learning model for fast learning network (FLN) based on particle swarm optimization (PSO) and applied to intrusion detection problem. This model has outperformed other learning approaches in the testing accuracy of the learning, but also counter the problem of less accuracy for a certain number of class due to the limited of training data for the class.

Deep learning techniques can automatically extract the features of a specific problem without the need for strong prior knowledge, which is hugely beneficial for intrusion detection. In particular, RNN has shown a strong advantage in this area. However, DNNs have more parameters and a higher degree of non-linearity, which means a higher design requirement. The most recent methods are based on RNN with LSTM. This is a good structure and preliminary research has proved its applicability to intrusion detection. However, performance measures such as the detection rate have not improved far beyond those of classic machine learning algorithms, which means that further improvements are needed. Also, LSTM is a relatively complex structure, which is detrimental to a real-time processing IDS.

III. SYSTEM COMPONENTS

A. RECURRENT NEURAL NETWORK

RNNs have a variety of structures, including the simple structure proposed by Elman [23]. An RNN is a development of a traditional feed-forward neural network. In the traditional neural network model, the data flow is unidirectional, that is, from the input layer to the hidden layer, and finally to the output layer. However, the RNN is different: it can remember the information processed at time t for the calculation at the subsequent time $(t+1, t+2\ldots)$. Therefore, the input of a hidden layer includes not only the output of the upper layer, but also the output of the same layer at the last time point.

An RNN can be unfolded in time and form a full network. The input set can be denoted as $\{i_0, i_1, \ldots i_{t-1}, i_t, i_{t+1}, \ldots\}$ and the output set as $\{o_0, o_1, \ldots o_{t-1}, o_t, o_{t+1}, \ldots\}$. For each hidden layer, the input set can be denoted as $\{x_0, x_1, \ldots x_{t-1}, x_t, x_{t+1}, \ldots\}$ and the output set as $\{h_0, h_1, \ldots h_{t-1}, h_t, h_{t+1}, \ldots\}$. U, V, W are weight matrices from the input layer to the hidden layer, the hidden layer to the output layer and inside the hidden layer, respectively. In an RNN, the hidden units complete the most important job. The information of the input layer flows unidirectionally to the hidden layer, but the hidden nodes are self-connected and interconnected in order to fully exchange the information.

An important extension to the RNN is the bidirectional RNN (BRNN) of Shuster and Paliwal [24] (Fig. 1). Its basic

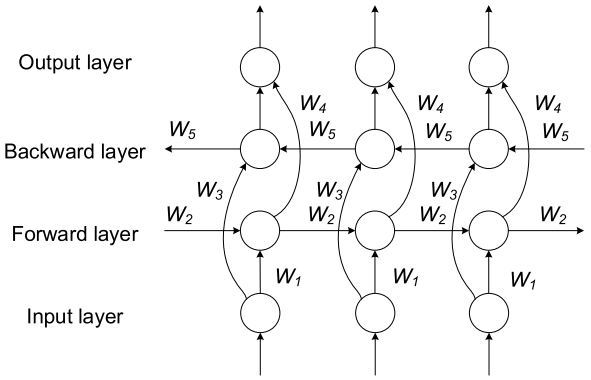

FIGURE 1. Bidirectional recurrent neural network.

idea is to put two opposite RNNs together, but share the same input and output layers. In this way, the trained data can be associated with both past and future information. The salient feature of a BRNN is two representative hidden layers: the forward and backward layers (Fig. 1). Other components are similar to those of general RNNs. There are five weight matrices in BRNN denoted as $W_1, W_2 \ldots W_5$. Note that there is no connection between the two hidden layers, so there are no loops in the unfolded network.

Training an RNN is similar to training a traditional ANN. Basically, the back propagation (BP) algorithm is used, but there are some differences. The full network unfolded from an RNN is in the time dimension, so the parameters are shared in space, which is quite different from a traditional neural network. For instance, Weight matrices $W_1, W_2 \ldots W_5$ are shared three times in Fig. 1. Parameter sharing significantly reduces the number of parameters and is one of the advantages of an RNN. In the running of the gradient descent algorithm, the output at each moment is based not only on the input of the same time, but also on the state of the previous time. This training algorithm is called back propagation through time (BPTT) [25].

Traditional RNNs encounter gradient vanishing or explosion problems [26]. To alleviate these problems, some specific RNN structures are proposed. LSTM and GRU are two such structures, which use a number of gates to control the memory and prevent the gradient vanishing.

B. GATED RECURRENT UNIT

A GRU proposed in [27] is a novel memory cell that has been proven to be effective in a variety of applications. A GRU can be seen as a simplification and improvement of LSTM and can be comparable in performance to LSTM [28]. To describe a GRU clearly, we briefly introduce LSTM first.

In an RNN, the hidden unit is the most important component as it is responsible for remembering or forgetting particular information. The LSTM proposed by Hochreiter and Schmidhuber [29] is a good implementation, and has many improved variants. A diagram of the structure of a common LSTM with a "peephole" [30] is shown in Fig. 2.

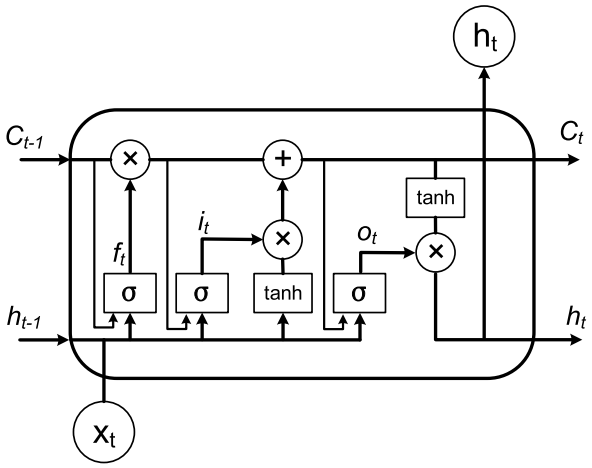

FIGURE 2. Structure of the LSTM.

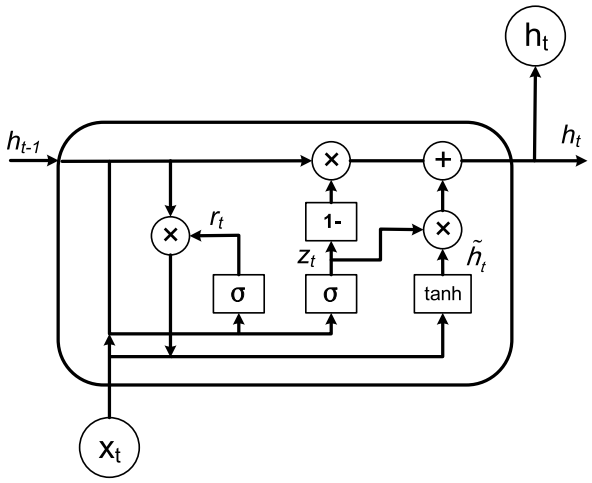

FIGURE 3. Structure of the GRU.

The connection relationship in Fig. 2 is given by Eq. (1).

$$\begin{cases} f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}C_{t-1}) \\ i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}C_{t-1}) \\ C_t = f_t \odot C_{t-1} + i_t \odot tanh(W_{xc}x_t + W_{hc}h_{t-1}) \\ o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}C_{t-1}) \\ h_t = o_t \odot tanh(C_t) \end{cases} \quad (1)$$

In Eq. (1), x is the input vector, h is the output vector and C is the cell state. The subscript t represents the current time and $t-1$ is the last time. σ is a sigmoid function, $\odot$ is the Hadamard product and W represents undetermined parameters. In Eq. (1), f is the forget gate that decides what information needs to be discarded from the cell state. i is the input gate that decides what information needs to stored in the cell state. o is the output gate that decides what information to output.

Compared with LSTM, a GRU includes some simplifications. A diagram of the structure of the GRU is shown in Fig. 3. The connection relationship in Fig. 3 is given

by Eq. (2).

$$\begin{cases} r_t = \sigma(W_r x_t + U_r h_{t-1}) \\ z_t = \sigma(W_z x_t + U_z h_{t-1}) \\ \tilde{h}_t = tanh(W_h x_t + U(r_t \odot h_{t-1})) \\ h_t = (1 - z_t)h_{t-1} + z_t \tilde{h}_t \end{cases} \tag{2}$$

In Eq. (2), x is the input vector, h is the output vector and $\tilde{h}$ is the candidate output. The other symbols are the same as before. The GRU has two gates: r is denoted as the reset gate and z as the update gate.

Compared to LSTM, the GRU has fewer gates. This is because the GRU has no cell state and combines the input and forget gates into a single gate, the update gate z. So, the GRU is much simpler than the LSTM in its structure and has fewer parameters, which gives it a great advantage in terms of performance and convergence. In subsequent experiments, GRU also showed a great advantage.

C. MULTILAYER PERCEPTRON

A multilayer perceptron (MLP) is a unidirectional ANN composed of multiple layers [31]. By using a non-linear activation function, MLP can identify linearly indivisible data. An MLP is characterized by signal forward propagation, error back propagation and is trained by a BP algorithm. The standard BP algorithm is a classic learning algorithm that calculates the difference between the actual output and the expected output, reverses the difference back to each layer, thereby adjusting the parameters of each layer to achieve the goal of learning.

A typical MLP consists of three major components: an input layer, several hidden layers and an output layer. Adjacent layers are fully connected, but nodes in the same layer are independent of each other. The activation function used by MLP is required to be continuous and monotonically increasing, such as the sigmoid function.

D. SOFTMAX REGRESSION

Softmax regression is a generalization of logistic regression which can produce a K-dimensional vector $\sigma(x)$ in the range (0, 1) from a K-dimensional vector x [32]. The equation is given by Eq. (3).

$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}} \quad j = 1, \ldots, K \tag{3}$$

The types of network attacks are diverse, so each particular record should be attributed to one of them (or normal). An IDS using a multi-classifier will be more appropriate. To build a multi-classifier, for a given input x, a hypothesis function is needed to estimate the probability $P(y = j|x)$ for each class j. That is, we need to estimate the probability of every possible classification output. Specifically, the hypothesis function should output a K-dimension vector (the sum of the vector elements is 1) to represent the estimated probability. The form of the hypothesis function is shown in Eq. (4).

$$\begin{aligned} h_\theta(x^{(i)}) &= \begin{bmatrix} P(y^{(i)} = 0|x^{(i)}; \theta) \\ P(y^{(i)} = 1|x^{(i)}; \theta) \\ \vdots \\ P(y^{(i)} = k-1|x^{(i)}; \theta) \end{bmatrix} \\ &= \frac{1}{\sum_{j=0}^{k-1} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_0^T x^{(i)}} \\ e^{\theta_1^T x^{(i)}} \\ \vdots \\ e^{\theta_{k-1}^T x^{(i)}} \end{bmatrix} \end{aligned} \tag{4}$$

In Eq. (4), the hypothesis function is denoted as $h_\theta(x^{(i)})$, $\theta_0, \theta_1, \ldots, \theta_{k-1}$ are the parameters to be determined, and $\frac{1}{\sum_{j=0}^{k-1} e^{\theta_j^T x^{(i)}}}$ is the normalization factor for the hypothetical function. In addition, if $\theta \to \infty$, then softmax becomes the maximum function. When taking different finite values, softmax can be considered a parameterized and softened version of the maximization function.

E. OVERALL ARCHITECTURE

To construct a DNN, every component should be regarded as a layer and cascaded together. The proposed IDS structure is shown in Fig. 4. The system consists of a preprocessing module, a GRU module, an MLP module and an output module. The preprocessing module processes the data into a normalized value suitable for the input neural network without changing the dimension of the data. The preprocessing algorithm for the experimental dataset is given below. The GRU module consists of one or more GRU (or bidirectional GRU) layer(s), which are used to extract and store features. This is the core of the system. The MLP module is an n-layer perceptron model, carrying out non-linear mapping from the output of the GRU module which makes a non-linear classification decision. The output module is a softmax layer, which normalizes the classification probability and outputs it as a final result.

Among these components, the GRU and MLP modules are significant for performance. The two modules are two different types of neural networks. The GRU has memory, but has a more complex structure and larger calculation. The MLP has a simple structure, a fast calculation and is easy to stack. The combination of the two constitutes a deep network which can achieve a more optimized result.

IV. EXPERIMENTS AND ANALYSIS

A. BENCHMARK DATASETS

The best way to evaluate an IDS is to use a common dataset for testing, so that a fair comparison of different systems can be made.

The DARPA/KDD Cup 99 dataset (usually abbreviated as KDD 99) has been widely used for many years in the

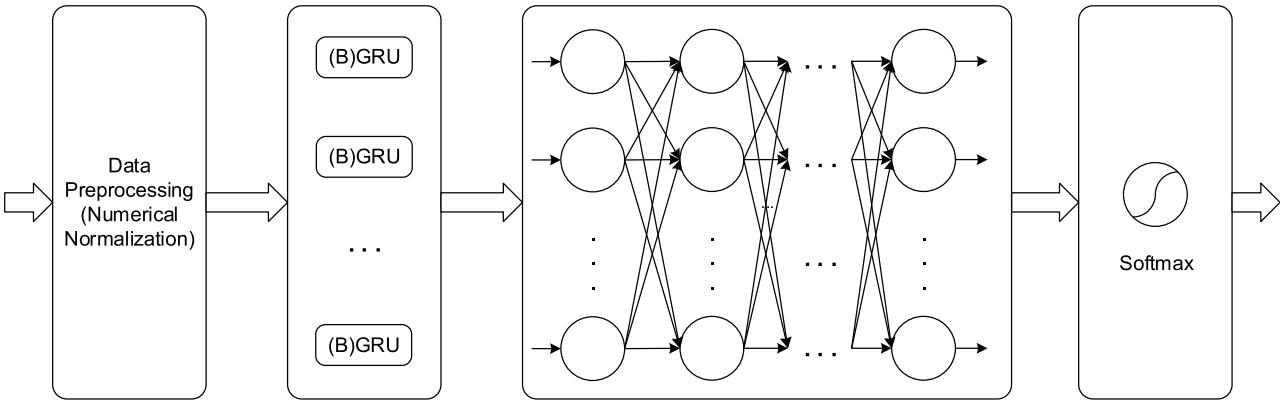

FIGURE 4. Proposed system structure.

evaluation of IDSs and has become the de-facto standard for benchmarking [33]. The KDD 99 dataset was built based on the data captured in the DARPA'98 IDS evaluation program by MIT Lincoln Labs [34]. It contains seven weeks of training data and two weeks of test data. This dataset includes 39 types of attacks: 22 are in the training set, and 17 are appear only in the test set as unknown attack types for testing the generalization performance of the algorithm. All these attacks can be divided into four categories:

1) DOS: denial-of-service, to prevent users from accessing a service, e.g. syn flood;
2) R2L: unauthorized access from a remote machine, e.g. guessing password;
3) U2R: unauthorized access to local root privileges, e.g. buffer overflow;
4) PROBING: surveillance and other probing, e.g. port scanning.

In addition, taking NORMAL (no attacks) into consideration, each record is is assigned to one of those five categories.

In KDD 99, each connection record has 41 features: 34 continuous and 7 discrete-valued. All the features can be divided into four broad categories:

1) Basic features of individual TCP connections. These features are directly extracted from the header of the packets;
2) Content-based derived features. Content features within a connection suggested by domain knowledge;
3) "same host" features. These features examine only the connections in the past 2 seconds which have the same destination host as the current connection;
4) "same service" features. These features examine only the connections in the past 2 seconds that have the same service as the current connection.

"Same host" and "same service" features are together called time-based traffic features of the connection records. More details about this dataset can be found in [33].

Tavallaee *et al.* [35] proposed a revised version of the KDD 99 dataset named NSL-KDD. The NSL-KDD dataset overcomes some shortcomings found in KDD 99. For instance, it does not include redundant or duplicate records. The number of selected records from each difficulty level is more appropriate, which makes it more efficient for obtaining a fair evaluation. The total number of records is reasonable, which makes it possible to run algorithms on the complete dataset rather than on a small portion selected randomly. As a result, evaluations from different studies can be more easily compared.

In this study, both the KDD 99 and NSL-KDD datasets were exploited to evaluate the proposed IDS. In each dataset, we used the most common used data files for fair comparison.

B. EVALUATION METRICS

For classification problems, the result of a classification can be correct or incorrect, and all possible results can be divided into the following four conditions:

1) True Positive (TP): actual attacks are classified as attacks;
2) True Negative (TN): actual normal records are classified as normal;
3) False Positive (FP): actual normal records are classified as attacks. This condition is also knows as false alarms;
4) False Negative (FN): actual attacks are classified as normal records.

For simplicity, TP, TN, FP, FN are used to represent the numbers of the four conditions. On this basis, the accuracy, precision, detection rate, false positive rate and F-measure can be defined as shown in Eq. (5).

Accuracy is the number of correct classifications as a proportion of the total number of records. Precision is the number of actual attacks as a proportion of the number classified as attacks. The detection rate (DR) is the number classified as attacks as a proportion of the number of actual attacks. The false positive rate (FPR) is the number classified as attacks as

TABLE 1. Classification label coding.

Classification label	code
NORMAL	0
DOS	1
R2L	2
U2R	3
PROBE	4

a proportion of the number of all normal records.

$$\begin{cases} Accuracy = \dfrac{TP + TN}{TP + TN + FN + FP} \\ Precision = \dfrac{TP}{TP + FP} \\ Detection\ Rate\ (DR) = \dfrac{TP}{TP + FN} \\ False\ Positive\ Rate\ (FPR) = \dfrac{FP}{FP + TN} \\ F - measure = \dfrac{2(Precision * DR)}{Precision + DR} \end{cases} \tag{5}$$

On the one hand, from the point of view of a classifier, the precision and detection rate are a pair of contradictory metrics. Higher precision means fewer false positives, but a higher detection rate means fewer false negatives. For example, if more suspected attacks are classified as attacks (the extreme situation is that all records are classified as attacks), the detection rate will increase, but precision will decrease, and vice versa. So a single high precision or detection rate is meaningless. On the other hand, from the point of view of intrusion detection, especially in some strict environments, the tolerance for intrusion is very low, so a separate detection rate is also an important metric for consideration.

The F-measure is a comprehensive consideration of the precision and detection rate. It is based on their harmonic mean. A higher F-measure means a higher precision and detection rate.

C. DATA PREPROCESSING

The proposed system can accept only numeric inputs, so it is necessary to convert the non-numerical data in the dataset into numerical data. Since NSL-KDD is a revision of KDD 99, their data types are the same. In each record, only three features (protocol type, service and flag) are symbolic and need to be converted to numerical data. 1-to-N encoding is applied to accomplish this. Similarly, the classification results are represented numerically (0 to K-1), as shown in Table 1.

Next, min-max normalization was applied to scale the feature data linearly between 0 and 1. Therefore, Eq. (6) was applied to each feature in each record in the dataset:

$$f' = \frac{f - \min_j}{\max_j - \min_j} \tag{6}$$

In Eq. (6), f is the original value of the feature, f' is the normalized value, and $\max_j$ and $\min_j$ are the maximum and minimum values, respectively, of the *j-th* feature.

TABLE 2. Hyper-parameter configuration.

Hyper-parameter	Value
Batch Size	32
Epoch	20
Learning Rate	0.01
Decay	10^{-5}
Momentum	0.9
MLP Layers	3
MLP hidden nodes	48
RNN hidden units	128

TABLE 3. Experimental results on KDD 99 dataset.

System	Accuracy (%)	DR (%)	FPR (%)
BGRU+MLP	**99.84**	**99.42**	**0.05**
GRU+MLP	99.28	96.73	0.07
BLSTM+MLP	98.57	93.78	0.17
LSTM+MLP	98.51	94.77	0.53
GRU	92.28	71.77	0.13
LSTM	91.91	70.77	0.10
MLP	91.88	70.92	0.31

TABLE 4. Experimental results on NSL-KDD dataset.

System	Accuracy (%)	DR (%)	FPR (%)
BGRU+MLP	**99.24**	**99.31**	**0.84**
GRU+MLP	99.19	99.35	1.00
BLSTM+MLP	96.41	95.65	2.67
LSTM+MLP	95.22	93.97	3.24
GRU	94.94	94.76	4.84
LSTM	94.1	95.65	7.58
MLP	90.56	86.61	3.49

D. EXPERIMENTAL RESULTS

Our experiments were implemented under the following hardware and software platforms: Hardware: Intel Core i7 @ 3.4 GHz, 64 GB RAM, NVIDIA TESLA K40. Software: Ubuntu 16.04 LTS, CUDA 8.0, cuDNN 6.0, TensorFlow 1.4.1. All the software can be downloaded freely from the internet.

To evaluate system performance objectively, the following experiments were performed 10-fold for cross-validation on two datasets and test sets ware used to evaluate the performance. As it is a DNN, stochastic gradient descent (SGD) was used in the training phase [36]. To improve efficiency, the cross entropy was used as the cost function instead of the minimum squared error (MSE) function [37]. The learning rate and the number of iterations were determined by practical experience. The hyper-parameter configuration are shown in Table 2.

In the proposed system, the GRU and MLP modules are the most important, so the following experiments focused on verifying the validity and necessity of the two modules.

Experiment 1 evaluated GRU + MLP performance. As a reference, the LSTM module was replaced with the GRU module in the same experimental environment. In addition, bidirectional RNN experiments were added to determine

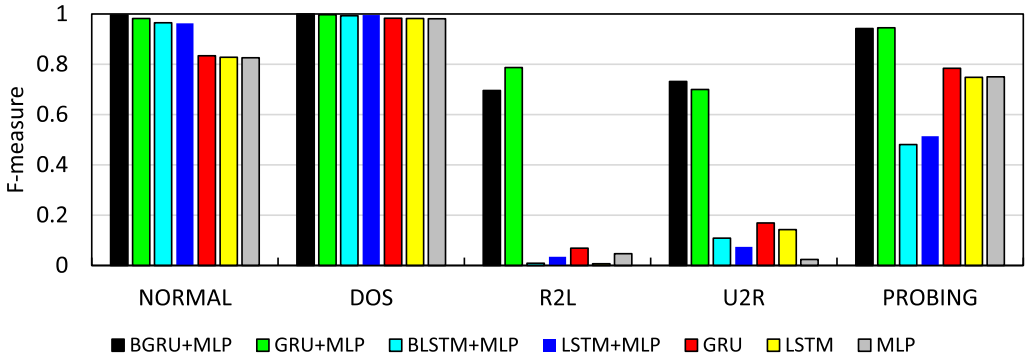

FIGURE 5. F-measure on KDD99.

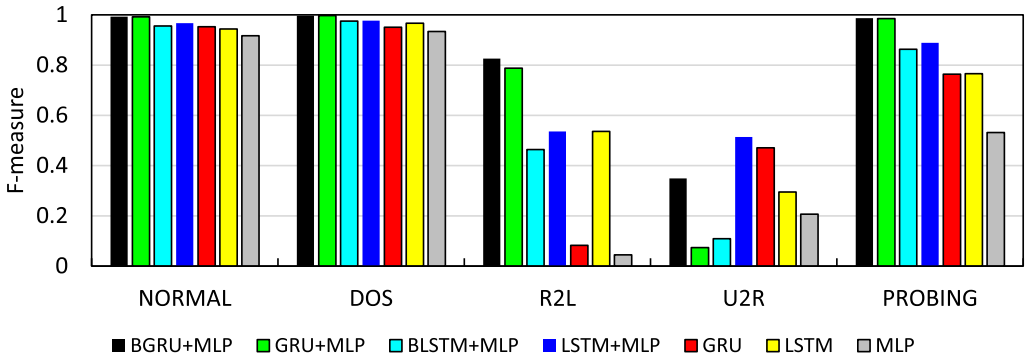

FIGURE 6. F-measure on NSL-KDD.

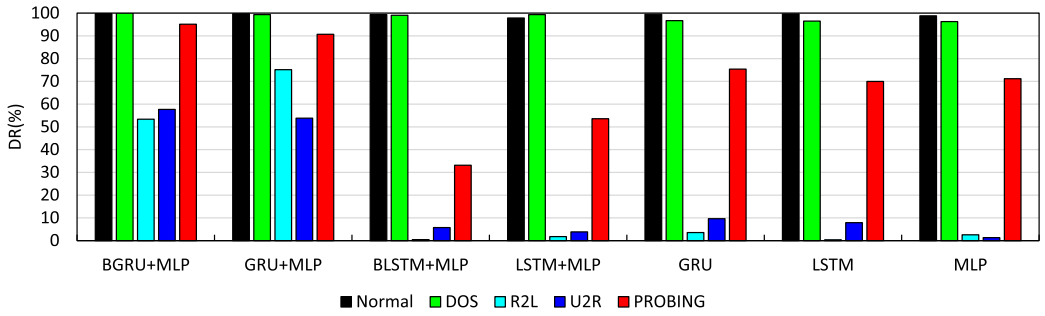

FIGURE 7. Detection rate on KDD 99.

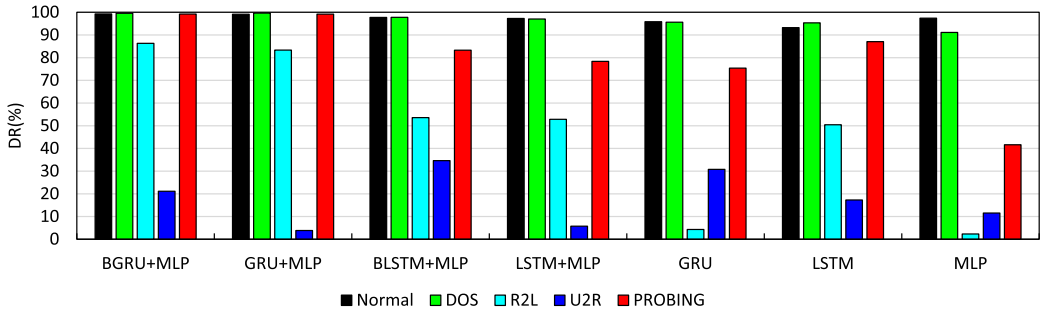

FIGURE 8. Detection rate on NSL-KDD.

whether it was necessary. In the results, the prefix "B" indicates that it is a bidirectional RNN.

Experiment 2 evaluated the situation when each module was operating independently. The RNN module or the

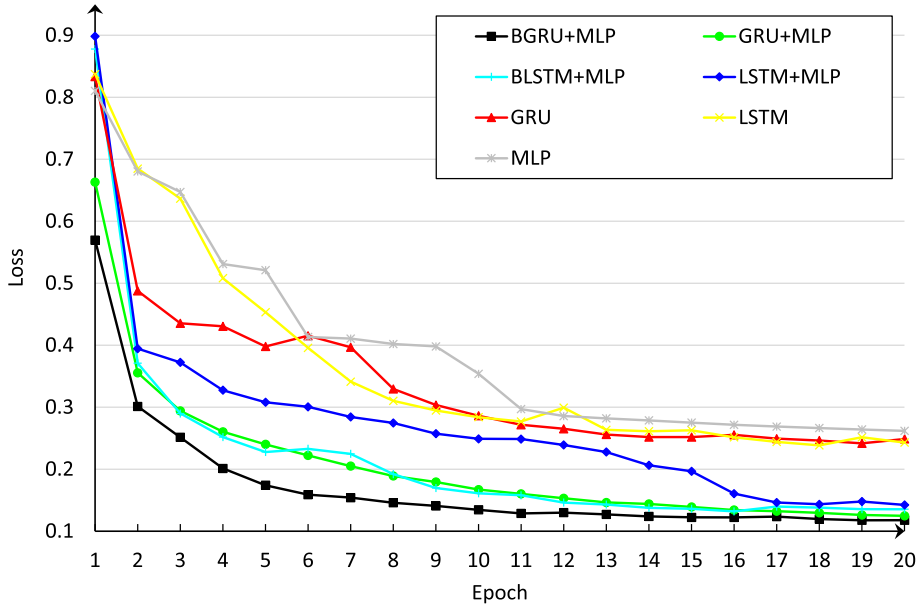

FIGURE 9. Comparison of convergence.

TABLE 5. Performance comparison.

System	Dataset	Accuracy (%)	DR (%)	FPR (%)
LSTM(2015) [20]	KDD 99	94.11	77.07	0.18
OS-ELM(2015) [38]	NSL-KDD	N/A	97.67	1.74
LSSVM+FMIFS(2016) [39]	KDD 99	99.79	99.46	0.13
LSSVM+FMIFS(2016) [39]	NSL-KDD	99.91	98.76	0.28
LSTM-RNN(2016) [21]	KDD 99	96.93	98.88	10.04
TVCPSO-MCLP(2016) [12]	NSL-KDD	N/A	97.23	2.41
Pruning VELM(2017) [15]	KDD 99	98.94	98.37	0.32
VELM(2017) [15]	NSL-KDD	97.58	97.69	2.22
GA+FLN(2018) [22]	KDD 99	99.69	N/A	N/A
PSO+FLN(2018) [22]	KDD 99	99.68	N/A	N/A
BGRU+MLP (proposed)	**KDD 99**	**99.84**	**99.42**	**0.05**
BGRU+MLP (proposed)	**NSL-KDD**	**99.24**	**99.31**	**0.84**

MLP module was removed before carrying out similar experiments.

Experiments 1 and 2 were performed on the KDD 99 and NSL-KDD datasets, respectively. The results are summarized in Tables 3 and 4, respectively. Fig. 5 and Fig. 6 show the F-measures of the different systems and attack types. Next, we calculated the detection rates and summarized by attack types. The results on the KDD 99 and NSL-KDD datasets are illustrated in Fig. 7 and Fig. 8, respectively.

From the above results, we conclude that the result of BGRU + MLP was the best for both the KDD 99 and the NSL-KDD datasets. Experiment 1 showed that GRU was better than LSTM, and that BGRU was better than GRU. A bidirectional RNN can further improve the performance of the RNN. Experiment 2 showed that the combination of RNN and MLP was effective, giving results better than those of the RNN (GRU and LSTM) or MLP alone. In terms of attack types, DOS and PROBING attack detection were significantly better than R2L and U2R attack detection.

Fig. 9 compares the convergence of the different algorithms. All experiments converged after about 17 epochs and 20 epochs are enough. Combined with the previous results, it can be seen that the system using (B)GRU + MLP has advantages not only in terms of accuracy, but also with a faster convergence rate.

V. DISCUSSION

To improve the analysis of the experimental results, we compared the results with those of previous studies (Table 5).

From Table 5, it can be found that BGRU + MLP performed well on both datasets. Using KDD 99, it achieved the best accuracy and FPR. Using NSL-KDD, it achieved the highest DR.

Note that such a comparison is only a reference rather than an absolute distinction. Different IDSs differ in their intrusion responses, and it is difficult to find a system that can achieve the best performance in every situation. In addition, due to a slight difference in the evaluation method, for example,

random sampling in datasets, the final results could be different. Nevertheless, compared with recent studies, we believe that our proposed system has significant advantages in terms of accuracy, DR and FAR.

The detection of R2L and U2R was not ideal. This is a common problem both in our proposed system and other systems. We think there are two main reasons. First, the recorded number of these two types of attacks was too small. In the KDD 99 dataset, the proportions of U2R and R2L were 0.01% and 0.23%, respectively. In the NSL-KDD dataset, as a revised version of KDD 99, the proportions of U2R and R2L increased to 0.04% and 0.79%, respectively, but this is an insignificant improvement. Such few records make extraction of the features by the learning algorithm ineffective, so that the classification accuracy is not as high as for other measures. Secondly, the RNN-based system has a greater advantage in dealing with time-series tasks. The DOS and PROBING attacks have more obvious timing characteristics than R2L and U2R attacks, enabling a higher detection performance to be achieved.

VI. CONCLUSION

In this study, we designed a new IDS. We propose a new DNN model which uses GRUs as the main memory unit, combined with MLP to identify network intrusions. Deep learning techniques were used for training and achieved good performance. Experiments on the well-known KDD 99 and NSL-KDD datasets showed that the system has leading performance. The overall detection rate was 99.42% on KDD 99 and 99.31% on NSL-KDD, with false positive rates as low as 0.05% and 0.84%, respectively. In particular, the detection rates for DOS attacks were 99.98% on KDD 99 and 99.55% on NSL-KDD. Compared experiments were done on LSTM and GRU with or without bidirectional connections. The combination of bidirectional GRUs and an MLP outperformed other recently published methods.

The system proposed in this paper relies mainly on theoretical verification. A lot of engineering work should be done to verify its practical application. The next step could be to optimize the system so that it can be applied to real network environments and be implemented more efficiently.

REFERENCES

[1] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.

[2] S. Sharma and R. K. Gupta, "Intrusion detection system: A review," *Int. J. Secur. Appl.*, vol. 9, no. 5, pp. 69–76, 2015.

[3] J. Allen, A. Christie, W. Fithen, J. McHugh, and J. Pickel, "State of the practice of intrusion detection technologies," Softw. Eng. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU/SEI-99-TR-028, 2000.

[4] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[5] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "DRAW: A recurrent neural network for image generation," in *Proc. ICML*, 2015, pp. 1462–1471.

[6] A. Hannun *et al.* (2014). "Deep speech: Scaling up end-to-end speech recognition." [Online]. Available: https://arxiv.org/abs/1412.5567

[7] D. E. Denning, "An intrusion-detection model," *IEEE Trans. Softw. Eng.*, vol. SE-13, no. 2, pp. 222–232, Feb. 1987.

[8] L. M. Ibrahim, D. T. Basheer, and M. S. Mahmod, "A comparison study for intrusion database (KDD99, NSL-KDD) based on self organization map (SOM) artificial neural network," *J. Eng. Sci. Technol.*, vol. 8, no. 1, pp. 107–119, 2013.

[9] M. S. M. Pozi, N. Sulaiman, N. Mustapha, and T. Perumal, "Improving anomalous rare attack detection rate for intrusion detection system using support vector machine and genetic programming," *Neural Process. Lett.*, vol. 44, no. 2, pp. 279–290, 2016.

[10] B. M. Aslahi-Shahri *et al.*, "A hybrid method consisting of GA and SVM for intrusion detection system," *Neural Comput. Appl.*, vol. 27, no. 6, pp. 1669–1676, 2016.

[11] J. Hussain, S. Lalmuanawma, and L. Chhakchhuak, "A two-stage hybrid classification technique for network intrusion detection system," *Int. J. Comput. Intell. Syst.*, vol. 9, no. 5, pp. 863–875, 2016.

[12] S. M. H. Bamakan, H. Wang, T. Yingjie, and Y. Shi, "An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization," *Neurocomputing*, vol. 199, pp. 90–102, Jul. 2016.

[13] W. Feng, Q. Zhang, G. Hu, and J. X. Huang, "Mining network data for intrusion detection through combining SVMs with ant colony networks," *Future Gener. Comput. Syst.*, vol. 37, pp. 127–140, Jul. 2014.

[14] A. A. Aburomman and M. B. I. Reaz, "A novel SVM-kNN-PSO ensemble method for intrusion detection system," *Appl. Soft Comput.*, vol. 38, pp. 360–372, Jan. 2016.

[15] Y. Shen, K. Zheng, C. Wu, M. Zhang, X. Niu, and Y. Yang, "An ensemble method based on selection using bat algorithm for intrusion detection," *Comput. J.*, vol. 61, no. 4, pp. 526–538, 2018.

[16] T. Ma, F. Wang, J. Cheng, Y. Yu, and X. Chen, "A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks," *Sensors*, vol. 16, no. 10, p. 1701, 2016.

[17] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLoS ONE*, vol. 11, no. 6, p. e0155781, 2016.

[18] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning," *Pattern Recognit.*, vol. 58, pp. 121–134, Oct. 2016.

[19] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proc. 9th EAI Int. Conf. Bio-Inspired Inf. Commun. Technol. (BIONETICS)*, New York, NY, USA, vol. 35, 2015, pp. 21–26.

[20] R. C. Staudemeyer, "Applying long short-term memory recurrent neural networks to intrusion detection," *South Afr. Comput. J.*, vol. 56, no. 1, pp. 136–154, 2015.

[21] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *Proc. Int. Conf. Platform Technol. Service (PlatCon)*, 2016, pp. 1–5.

[22] M. H. Ali, B. A. D. Al Mohammed, A. Ismail, and M. F. Zolkipli, "A new intrusion detection system based on fast learning network and particle swarm optimization," *IEEE Access*, vol. 6, pp. 20255–20261, 2018.

[23] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[24] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[25] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[26] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[27] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," in *Proc. 8th Workshop Syntax, Semantics Struct. Stat. Transl.*, 2014, p. 103.

[28] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proc. NIPS Deep Learn. Workshop*, Dec. 2014, p. 47.

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

[30] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Comput.*, vol. 12, no. 10, pp. 2451–2471, 2000.

[31] S. K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets, and classification," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 683–697, Sep. 1992.

[32] C. M. Bishop, "Probability distributions," in *Pattern Recognition and Machine Learning*, 2nd ed. New York, NY, USA: Springer, 2006, ch. 2, sec. 4, pp. 113–116.
[33] (1999). *KDD CUP 1999 Data*. Accessed: Dec. 21, 2016. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/
[34] MIT-Lincoln-Labs. (1999). *DARPA Intrusion Detection Data Sets*. Accessed: Dec. 25, 2016. [Online]. Available: https://www.ll.mit.edu/ideval/data/
[35] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl. (CISDA)*, Jul. 2009, pp. 1–6.
[36] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. COMPSTAT*. New York, NY, USA: Springer, 2010, pp. 177–186.
[37] J. Shore and R. Johnson, "Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy," *IEEE Trans. Inf. Theory*, vol. IT-26, no. 1, pp. 26–37, Jan. 1980.
[38] R. Singh, H. Kumar, and R. K. Singla, "An intrusion detection system using network traffic profiling and online sequential extreme learning machine," *Expert Syst. Appl.*, vol. 42, no. 22, pp. 8609–8624, 2015.
[39] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 2986–2998, Oct. 2016.

CONGYUAN XU (S'18) received the B.S. degree in applied physics from Xidian University, Xi'an, China, in 2013. He is currently pursuing the Ph.D. degree in Zhejiang University, Hangzhou, China. His current research focuses on machine intelligence and cyberspace security.

JIZHONG SHEN received the Ph.D. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2001. He is currently a Full Professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou. He has authored/co-authored over 180 refereed technical papers on low-power digital circuits design, brain–computer interface, and so on. His current research interests include digital integrated circuit & low power design, digital system security, and brain-computer interface.

XIN DU received the B.S. degree in electronic engineering from Hangzhou University, Hangzhou, China, in 1997, and the M.S. and Ph.D. degrees from Zhejiang University, Hangzhou, in 2000 and 2003, respectively. He is currently an Associate Professor with the College of Information Science and Electronic Engineering, Zhejiang University. His current research interests include image processing, data security, and machine intelligence.

FAN ZHANG (M'17) received the Ph.D. degree from the Department of Computer Science and Engineering, University of Connecticut, in 2012. He is currently with the College of Information Science and Electrical Engineering, Zhejiang University, China. His main interests include side channel analysis and fault analysis in cryptography, computer architecture, and security in wireless network.