Received January 29, 2020, accepted February 18, 2020, date of publication February 24, 2020, date of current version March 3, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2975877

A Content-Based Method for Sybil Detection in Online Social Networks via Deep Learning

TIANYU GAO[1], JIN YANG[2], WENJUN PENG[1], LUYU JIANG[1], YIHAO SUN[1], AND FANGCHUAN LI[1]
[1]College of Software Engineering, Sichuan University, Chengdu 610207, China
[2]College of Cybersecurity, Sichuan University, Chengdu 610207, China

Corresponding author: Jin Yang (841025071@qq.com)

This work was supported in part by the Natural Science Foundation of China under Grant 61872254, in part by the Fundamental Research Funds for the Central Universities under Grant YJ201727, in part by the Application Foundation Project of Sichuan Province Science and Technology Department under Grant 2014JY0036 and Grant 2018JY0913, and in part by the Foundation for Training and Supporting Academic and Technological Leaders in Sichuan Province (2016).

ABSTRACT Online social networks (OSNs) are generally susceptible to Sybil attack, which causes a series of cybersecurity problems and privacy violations. Malicious attackers can create massive Sybils and further utilize those fake identities to launch various Sybil attacks. Therefore, Sybil detection in OSNs has become an urgent security research problem for both academia and industries. The existing content-based methods to detect Sybils base on the combination of manual-design features and machine learning algorithms, which requires lots of professional experiences and human effort. These methods divide the Sybil detection problem into two piece-wise sub-problems, which prevents us from getting the optimal solution. In this work, we propose a novel content-based method to detect Sybils. The proposed method is an end-to-end classification model that extracts features directly from the input data, and then output the classification results in a unified framework. The proposed method includes three main parts: first, the self-normalizing convolutional neural network (CNN) is adopted to extract lower features from the multi-dimensional input data; second, the bidirectional self-normalizing LSTM network (bi-SN-LSTM) is developed to extract higher features from the compressed feature map sequence; third, the dense layer and softmax classifier are stacked to output the classification results. Unlike the traditional bidirectional long short-term memory network (bi-LSTM), the proposed bi-SN-LSTM network utilizes SELU as the activation function of its recurrent step, which provides unbounded changes to the state value. Through the case study of the real-world dataset, the comparison experiments demonstrate that our method significantly outperforms other state-of-the-art methods.

INDEX TERMS Convolutional neural network (CNN), deep learning (DL), long short-term memory (LSTM), online social networks (OSNs), sybil detection.

I. INTRODUCTION

In recent years, online social networks (OSNs) have had an ever-increasing impact on human society, including the impact on people's daily life, business models and international communication methods, etc. Meanwhile, with the rapid development of OSNs, the number of users on those platforms (e.g., Twitter, Facebook, Instagram, etc.) is also rising rapidly. Take Twitter as an example, there are 145 million daily active Twitter users that create 500 million tweets per day [1]. With the increasing influence of OSNs, attackers appear, and they can launch Sybil attacks through a large number of malicious users, which are called Sybils. Examples of Sybils include, but not limited to, spammers, compromised normal users, and fake users [2]. Sybil attack takes advantage of its high proportion of Sybils among users of the entire OSN to increase the impact of multiple malicious activities, such as spamming [3], phishing attacks [4], illegal access to personal privacy information [5], malware distribution [6], etc. Attackers can further leverage the influence of Sybils to multiply the losses caused by the above malicious attack methods to the OSN. Moreover, Sybils can mimic benign users to bypass detection, which increases the difficulty of detecting malicious users. Therefore, the Sybil

The associate editor coordinating the review of this manuscript and approving it for publication was Fan-Hsun Tseng.

detection in OSNs is an urgent problem that needs to be addressed, and it has already become a focus issue for both academia and industries.

The existing Sybil detection methods can be summarized into two main categories, which are structure-based methods and content-based methods. In the above two types of methods, content-based methods represent each user using multi-dimensional features, which can be collected from user-profiles and the structure of users' local subgraph. Then, the feature vectors corresponding to labeled users are concatenated together to form the final training dataset. Finally, the binary classifier is trained to predict labels of users who are not used for model training. In conclusion, the core idea of the content-based method is to treat the Sybil detection problem as a binary classification problem based on machine learning classifiers. The general procedures of content-based methods are decomposed into two sub-problems, which are feature engineering and model training. The purpose of feature engineering is to manually design features and select effective features. To improve the performance of the classifier, the above sub-problem greatly requires not only professional experiences to design high-quality features, but also human effort to verify the contribution of selected features. Moreover, the model training is to obtain the classification model (e.g., the random forest classifier) by utilizing the selected features, which also needs human effort to select proper classification models to output convincing results carefully. Both of these sub-problems can directly affect the classification performance of content-based methods. In conclusion, traditional content-based methods depend on human intervention heavily, which leads to an increase in classification errors and high consumption of human effort.

To address these problems, we propose an end-to-end classification model, which is a novel content-based method for Sybil detection in OSNs. The purpose of designing an end-to-end model is to combine the feature engineering and model training together as a unified model. Our model can automatically extract and learn features directly from the raw input data. In other words, the proposed model significantly saves human effort for manually designing, selecting, and verifying features. The major architecture of the proposed model is composed of three main parts. Firstly, our model automatically extracts lower features of input data (i.e., the multi-dimensional input feature information) by a self-normalizing CNN. The extraction result of lower features forms a feature map sequence, where each feature map corresponds to a different user in the original dataset. Because RNN-based methods are very suitable for the analysis of sequence data, so we utilize the bidirectional SN-LSTM (bi-SN-LSTM) network to further extract hidden features in the following structure of the proposed model. The proposed bi-SN-LSTM network summarizes the feature map sequence with both directions (i.e., forward and backward) to extract higher features, e.g., the correlation information between users. Utilizing SELU as the activation function for the recurrent step, the proposed bi-SN-LSTM outperforms other commonly utilized RNN-based methods as demonstrated in Section V, which also proves the efficiency of the proposed method. Finally, the extracted feature information is fed into a dense layer and a softmax classifier to obtain classification results.

In conclusion, the features for classification can be extracted and learned automatically from the raw input data by the end-to-end model. As demonstrated in Section V, the lower features and higher features jointly improve the classification performance. The hierarchical architecture of the proposed model not only provides our model with excellent feature extraction capability but also increases the generalization of the classification model. To the best of our knowledge, our work is the first attempt to apply hierarchical deep learning (DL)-based method for Sybil detection in OSNs. Moreover, it is worthy to note that we utilize the combination of SELU and alpha dropout for preserving the self-normalizing property along the training process. With the same effect as normalization techniques (e.g., batch normalization, weight normalization, layer normalization, etc.), the self-normalizing property enables our model to robustly learn many layers without adding any normalization layers in the proposed method, which simplifies the model structure and enables our model to reach promising classification results when fed with large datasets from the real-world OSNs.

The main contributions of this paper are summarized as follows:

1. We propose a content-based end-to-end classification model to detect Sybils in OSNs. The proposed model utilizes self-normalizing CNN and bidirectional SN-LSTM (bi-SN-LSTM) to automatically extract lower and higher features from the input data, respectively. The end-to-end architecture of our model significantly saves human effort and improves classification results.
2. A bidirectional SN-LSTM network (bi-SN-LSTM) is proposed to extract and generate higher features from the feature map sequence. The experimental results prove that bi-SN-LSTM is more effective than other commonly utilized RNN-based methods such as bi-GRU and bi-LSTM, etc.
3. The proposed method is evaluated on the MIB dataset, which is a real-world dataset for the Sybil detection in OSNs. Experiment results show that our model outperforms several state-of-the-art content-based methods.

The rest of the paper is organized as follows. Section II discusses related work. Section III describes the problem definition and other preliminaries. Section IV formally describes the architecture of the proposed model in detail. The general introduction of the datasets, the detail of comparison experiments, experimental results, ablation analysis and further analysis on the proposed model are presented in Section V. Finally, the conclusion and the future works are drawn in Section VI.

II. RELATED WORK

Researches on OSN's Sybil detection in an endless stream. In this section, we introduce structure-based methods and content-based methods in this field.

A. STRUCTURE-BASED METHODS

Structure-based methods are mainly based on the structure of the social graph. Those methods distinguish Sybils and human users by analyzing the edges and links of the graph. Yu *et al.*. proposed a decentralized protocol SybilGuard [7] to identify the Sybil nodes and limit the influence of the Sybil attack by utilizing random routes. On the basis of SybilGuard, Yu *et al.* proposed a decentralized protocol using the same insight as SybilGuard but with different random walk-based (RW-based) methods in work [8]. Danezis and Mittal proposed a centralized Sybil detection algorithm, and they obtained each node's degree of certainty by calculating their probability to become Sybil with a Bayesian inference approach [9]. Tran designed Sybil-resilient to improve Sybil-Limit under the assumption of large-scale social networks considered as random expander graphs [10]. Yang proposed a new tool that relies on social graph properties to rank users base on their perceived likelihood of being Sybil by work [11]. Furthermore, work [12] further thwart Sybils by reducing the social edges on users that have received negative feedback, and it reached better detection accuracy compared to work [11]. Wei *et al.* proposed a mechanism that using the network topologies and RW-based methods to defend against Sybil attack in OSNs [13].

Xue *et al.* proposed a system that further leverages user interactions information [14], they evaluated whether the user is a Sybil by utilizing trust-based vote assignment and global vote aggregation. Furthermore, the system outperformed multiple existing ranking system on the real-world OSN environment. Yang *et al.* further utilized user-level activities in work [15], they proposed a voting-based Sybil detection method to identify Sybils and detect Sybil communities to find some more related Sybils. Boshmaf *et al.* designed a scalable defense system with better use of each node's features to detect automated Sybils in work [16], they achieved a precision(95%) that is over two times higher than SybilRank (43%). To lower the running time complexity and reduce dependency on the proper choice of known trusted nodes, Misra *et al.* proposed a Sybil community detection algorithm based on the communities' apparent likelihood of being a Sybil [17].

Bansal et al. (2016) modified multiple structure-based methods by making use of the trust value of each user, it showed at around 14% decrease in false-positive rate comparing with [7] and [18] in work [19]. Wang *et al.* [20] utilized some advantages of both loop belief propagation-based (LBP-based) and RW-based methods to make their orders of magnitude more scalable than a semi-supervised learning method [21]. Zhang *et al.* utilized users' activities and detect Sybils by coupling three RW-based algorithms in work [22]. Wang *et al.* (2019) proposed a structure-based model to detect Sybils in work [2], it takes the advantages from both RW-based and LBP-based methods to outperforms some existing methods, i.e., SybilRank [11] and Sybilbelief [21].

B. CONTENT-BASED METHODS

The content-based mechanisms are mainly utilizing Machine Learning (ML) methods. ML has been well applied to many frontier research areas. Researchers focus on using ML classifiers to better utilize the feature information of users in OSNs. There are several attempts to apply ML to the Sybil detection problem in OSNs. Wang *et al.* proposed a server-side clickstream model to group user clickstreams that are close to each other into behavioral clusters [23]. However, the false-negative rate raised when testing their model with an unbalanced training dataset with the number of Sybils in the training dataset get lower. Alsaleh *et al.* built some classification models based on user features from Twitter, they utilized four different ML algorithms: decision tree (C4.5), random forest, SVM, and multilayer neural network [24]. Kang *et al.* combined the features of different users and the network reliability [25], they constructed a user discriminant formula to identify Sybils in OSNs with the false-positive rate fluctuating between 3.74% and 14.96%. Xia *et al.* proposed an attribute credibility-based Sybil detection approach in work [26], they calculated the user's credibility by using the Euclidean distance between the center of Sybils' attribute and users' attribute. Furthermore, they utilized the credibility as a key parameter to classify Sybils. After analyzing human user trajectories and the behavior of Sybil attack, Xu *et al.* designed a Bloom filter-based Sybil detection method, which better leveraged the user location data [27]. In work [28], Mulamba *et al.* leveraged multi-dimensional feature information that summarized from the structure of the OSN graph to build up ML classifiers, e.g., AdaBoost and KNN.

Al-Qurishi *et al.* further utilized feature information of users and built up a prediction model by utilizing a deep neural network, it is also the first time that deep learning (DL) method ever applied to the field of Sybil detection [29]. Since the OSNs are growing rapidly, the lack of robustness of traditional methods will let Sybils to bypass the detection by mimicking human users. In this paper, we attempt to leverage the hierarchical DL network structure to better utilize the multi-dimensional feature information of users and minimize detection errors for Sybils. The proposed method takes advantage of the well-designed feature extractors to improve the accuracy of Sybil detection problems.

III. PRELIMINARIES

In this section, we first give the definition of the Sybil detection problem in OSNs. Then, the convolutional neural network, vanilla long-short term memory (LSTM) network, scaled exponential linear unit (SELU), and alpha dropout are introduced briefly.

A. PROBLEM DEFINITION

The Sybil detection problem in this paper is to accurately detect the Sybils in the OSNs by end-to-end binary classifier based on deep learning. Every user is represented by the data composed of multi-dimensional features. Assume that there are a total number of M users and each user has N different features. We utilize different feature vectors to represent different users and let the $p-$ th user be $P_i = [K_1^{(i)}, K_2^{(i)}, \ldots, K_N^{(i)}]$, where $K_p^{(i)}$ is the value of $p-$th feature of P_i. The real label of P_i is denoted as T_i, we use $T_i = 1$ when P_i is Sybil and $T_i = 0$ when P_i is a human (i.e., benign user). To solve this classification problem, we build an end-to-end hierarchical model $\tau(P_i)$ to predict a label $\hat{T}_i$ that is exactly the real label T_i.

B. CNN MODEL

The Convolutional Neural Network (CNN) is one of the most popular neural networks that have been well applied to many different research areas, such as image data augmentation [30], object detection [31] and text classification [32]. Features are the critical components for any machine learning model, especially for deep learning methods. The CNN model's performance is greatly affected by the extraction results of the feature information. Therefore, the CNN model adopts multiple strategies (i.e., convolution, max pooling, etc.) to extract features and also enhance the computing efficiency. In the following part of Section III-B, we summarize the commonly utilized 2-dimensional (2D) convolution operation and max pooling, respectively.

In order to obtain the output feature map by utilizing convolution operation, the learnable kernels from the current layer are used to convolve with the previous layer's feature map and then put through the activation function with an additive bias is given. Moreover, each output feature map may result from the sum of distinct kernels convolution with multiple input feature maps. The convolution operation for each feature map is defined as follows,

$$m^t = f(\sum_{i,j} m_{i,j} * k_{i,j}^t + b) \tag{1}$$

where $m_{i,j}$ represents the sub-region of the input feature map with the size of $i \times j$, m^t denotes the $t-th$ output feature map of the corresponding input, $f(\cdot)$ denotes the output activation function, k and b defines the trainable kernel and bias, respectively.

The max pooling can down-sample the input feature maps to accelerate the convergence rate and also reduce the computation complexity. Each input feature map from the previous layer is spatially divided into multiple subregions by some fixed-size max pooling windows. Then, the pooling operation takes the maximum value of each subregion to form the output feature map with lower dimensionality is shown in Eq. (2).

$$m_{i,j}^n = \max_{0 \le k \le H', 0 \le l \le W'-1} m_{i+k,j+l}^n \tag{2}$$

where $m_{i+k,j+l}^n$ represents the value at the $(i+k, j+l)$ position in the $n-th$ input 2D feature map, $m_{i,j}^n$ denotes the value at the (i, j) position in the $n-th$ output 2D feature map, H' and W' denotes the height and width of the max pooling windows, respectively.

C. VANILLA LSTM MODEL

The vanilla Long-Short Term Memory Network (vanilla LSTM) is one of the most popular variants of Recurrent Neural Network (RNN) to model sequences. It's capable of solving the vanishing gradient problem by extracting and utilizing long-term dependencies. The single unit of vanilla LSTM network contains three gates (i.e., input gate, output gate and forget gate) and a memory cell to control the flow of information. The forget gate enables the vanilla LSTM to reset its state as follows,

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3}$$

where W_f is the trainable input weight, U_f represents the recurrent weight and b_f is the bias weight to be learned, x_t denotes the input at time step t, h_{t-1} is the hidden state at the previous time step. $\sigma(x) = 1/(1+\exp(-x))$ is the sigmoid function used as the gate activation function with the output in [0,1]. The input gate determines which information should enter the long-term memory and its output at time step t is computed by

$$i_t = \sigma\ (W_i x_t + U_i h_{t-1} + b_i) \tag{4}$$

where W_i, U_i, and b_i denote trainable variables. Next, the tanh function is applied to create a vector which could be added to the cell state as follows,

$$l_t = tanh\ (W_l x_t + U_l h_{t-1} + b_l) \tag{5}$$

where W_l, U_l, and b_l are also learnable parameters, $tanh\ (x) = (\exp(x) - \exp(-x))/(\exp(x) + \exp(-x))$ is the hyperbolic function with the output in $[-1,1]$. The update of the previous cell state C_{t-1} can be computed from the three outputs above, that is:

$$C_t = f_t \odot C_{t-1} + i_t \odot l_t \tag{6}$$

where C_t denotes the updated cell state, $\odot$ represents the element-wise product. Hence, the hidden state can be updated as follows,

$$h_t = tanh\ (C_t) \odot o_t \tag{7}$$

where o_t is the output of the output gate at time step t, shown as below,

$$o_t = \sigma\ (W_o x_t + U_o h_{t-1} + b_o) \tag{8}$$

where W_o, U_o, and b_o are also trainable variables. Therefore, we describe the update process briefly as:

$$h_t = LSTM\ (x_t, h_{t-1}) \tag{9}$$

where h_t is the output of the LSTM cell at time step t, $LSTM(\cdot)$ is the summary of the above equations.

D. SELU

The Scaled Exponential Linear Unit (SELU) [33] is a continuous curve that contains both positive and negative values for controlling the mean, a slope larger than one to increase the variance when it is too small and saturation regions to reduce variance when it is too large. When applying SELU as the activation function of the hidden layer, it is capable to automatically normalize the output of the corresponding layer to a fixed mean and variance. The SELU activation function is expressed as:

$$SELU\ (x) = \lambda \begin{cases} x & if\ x > 0 \\ \alpha e^{x} - \alpha & if\ x \le 0 \end{cases} \tag{10}$$

where hyperparameters $\alpha = 1.6732632$ and $\lambda = 1.0507009$ are derived but not trained to keep the mean and variance of the output of the hidden layers at 0 and 1, respectively.

E. ALPHA DROPOUT

Alpha dropout [33] is a variant of dropout used to prevent over-fitting while training neural networks. It's a regularization technique that can keep the mean and variance of the inputs' distribution to their original values. With the dropout variable h obeys a binomial distribution $B(1, q)$, the mean and variance after alpha dropout are respectively expressed as Eq. (11) and (12).

$$E(a(xh + \alpha'(1-h)) + b) = 0 \tag{11}$$

$$Var(a(xh + \alpha'(1-h)) + b) = 1 \tag{12}$$

where a and b are two parameters that only related to the $1-Q$ and randomly negative saturation value α'.

IV. PROPOSED METHOD

Our proposed method is composed of data preprocessing and the end-to-end classification model as illustrated in Fig. 1. The raw input data is preprocessed and then fed into our proposed end-to-end model. The proposed model is a hierarchical model that consists of multiple layers to extract features and to output the classification results. The feature extraction part is composed of self-normalizing CNN and the proposed bi-SN-LSTM that can extract lower features and higher features, respectively. The output features of the bi-SN-LSTM network are further fed into the stacked dense layer to seek a higher-level representation. Finally, the softmax classifier is adopted to obtain the classification result of the corresponding user. In the following part of this section, we will present the detail of our proposed method.

A. DATA PREPROCESSING

In the real-world OSNs, the multi-dimensional data of users have the input of different scales. Those unequally weighted input features may lead to an unstable model with higher generalization error. Applying the proper feature scaling method can transfer the raw data into a common scale to avoid the above problem. Moreover, it can also speed up the learning and convergence of the proposed model. To enhance the property of self-normalizing, we adopt z-score normalization [34] in this paper to make the resulting distribution has a standard deviation of 1 and a mean of 0, expresses as below:

$$x'_{(m,n)} = \frac{x_{(m,n)} - \mu_n}{\sigma_n} \tag{13}$$

where n denote the $n-th$ input features, m is the $m-th$ user of the input data, $x_{(m,n)}$ denote raw input data, $x'_{(m,n)}$ represent the normalized raw input data, μ_n and σ_n denote the mean and standard deviation of the $n-th$ feature, respectively.

To better extract features from normalized raw input data by utilizing 2-dimensional (2D) convolution, we completed the transformation of different user's input data from 1D to 2D, expressed as below:

$$vec^{-1}_{H,W} \circ vec\ :\ \mathbb{R}^{1\times N} \rightarrow \mathbb{R}^{H\times W} \tag{14}$$

$$m'_i = vec^{-1}_{H,W}(vec(P_i)) \tag{15}$$

where Eq. (14) defines the function composition, the transformation step is denoted as Eq. (15), the result of $H \times W$ is equal to the total numbers of the input features, m'_i represents the input feature map corresponding to the $i-th$ user.

B. PROPOSED END-TO-END MODEL

The proposed end-to-end architecture is shown in Fig. 1. Our model is composed of three parts as follows:

1. utilizing the self-normalizing CNN to automatically extract lower features from normalized input data. The self-normalizing property of our model is implemented by adopting the combination of SELU and alpha dropout along the training process of our model (i.e., keep the mean and variance of the corresponding hidden layer's output to 0 and 1, respectively).
2. leveraging the proposed bi-SN-LSTM network to summarize feature map sequence in two directions (i.e., forward and backward) to extract higher features (e.g., the correlation between users, etc.).
3. stacking dense layer to learn the representation extracted in step 3 and then leveraging the softmax classifier to output the classification results.

In conclusion, the last dense layer function as the classification layer of our model, we utilize the softmax classifier to generate the classification of different users. The overall performance of our model is improved by taking advantage of both lower input feature information (i.e., multi-dimensional input features of different users) and higher feature information (i.e., the feature information of correlation among different users that hidden in the input data). The necessity and efficiency of utilizing both lower and higher features will be demonstrated in Section V. We adopt the convolutional layer and the max pooling layer to automatically extract feature information of every single user. The CNN structure is the suitable feature extractor for extracting lower

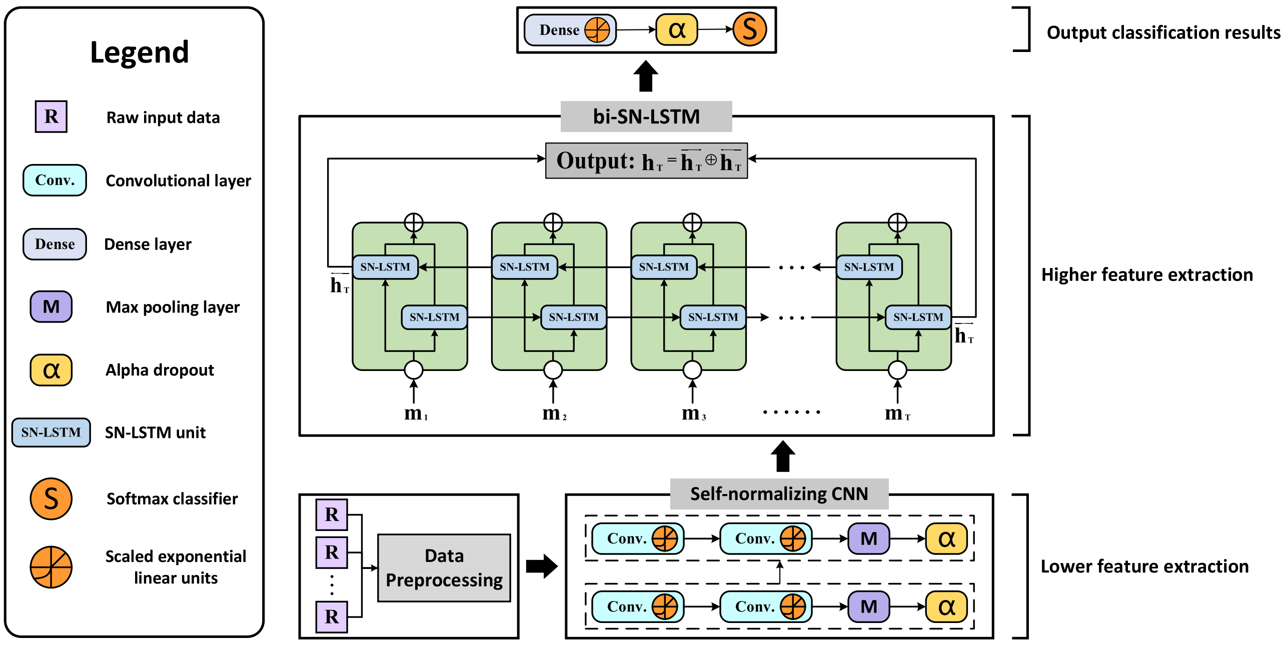

FIGURE 1. The architecture of the proposed model.

features because it can compress the multi-dimensional input data, and form the feature map sequence, which can be further fed into bi-SN-LSTM network to summarize higher features.

A large number of trainable parameters enables our model to output the desired classification results, but at the same time, it increases the risk of over-fitting. To address this problem, we utilize alpha dropout as the regularization method of the proposed model, instead of the standard dropout [35]. The standard dropout method prevents the corresponding layer in our model to keep both mean and variance to the desired value, and also, it cannot fit well with SELU because the default and low variance value is $\lim_{x \to -\infty} SELU(x) = -\lambda\alpha = \alpha'$ [33]. Moreover, alpha dropout fits well with SELU by randomly setting inputs to α' to preserve the self-normalizing property of the corresponding layer in our classification model.

In the first part of our proposed model, the inputs (i.e., $P = [m'_1, m'_2, m'_3, \ldots, m'_M]$) are fed into the CNN structure to automatically extract each user's feature information by utilizing the combination of convolutional layer and max pooling layer. To keep the mean and variance of input feature maps unchanged, we adopted SELU as the activation function in the convolutional layer to output the feature map as shown in Eq. (16). Moreover, SELU can efficiently stabilize the training process when working together with CNN without applying additional layers (e.g., Batch Normalization). Then, the max pooling layer is leveraged to summarize the features in different sub-regions of the output feature maps, which can improve the convergence speed of our model and also reduce the computational cost.

$$m^t = SELU\left(\sum_{i,j} m_{i,j} * k^t_{i,j} + b\right) \tag{16}$$

The output of the CNN structure constitutes a sequence of feature maps, and each of them represents the lower feature extraction result of the corresponding user. Then, we utilize the vanilla LSTM-based network to summarize higher features from the sequence of feature maps. The commonly utilized vanilla LSTM leverage the tanh as the activation function in the recurrent step to determine the candidate value that is added to the cell state. It has the bounded output of $[-1,1]$ to apply both positive and negative changes to the state value. Comparing with tanh, the ReLU activation function is less prone to the vanishing gradient problem, faster to execute and can also induce sparseness. However, ReLU has the unbounded positive output that may let the hidden state to grow exponentially large, and also, it cannot decrease the state because it cannot output the strictly negative values. Inspired by work [36] that opened up possibilities for replacing the tanh activation function with an unbounded activation function in the LSTM-based network, we leverage SELU as the activation function of the recurrent step in the proposed SN-LSTM model. It not only remains the advantages of ReLU but also provides a negative output to decrease the cell state. The exponentially grown hidden state provided by SELU improves the classification performance, which will be further demonstrated in Section V. In conclusion, the proposed unit architecture of SN-LSTM is shown in Fig. 2. It can

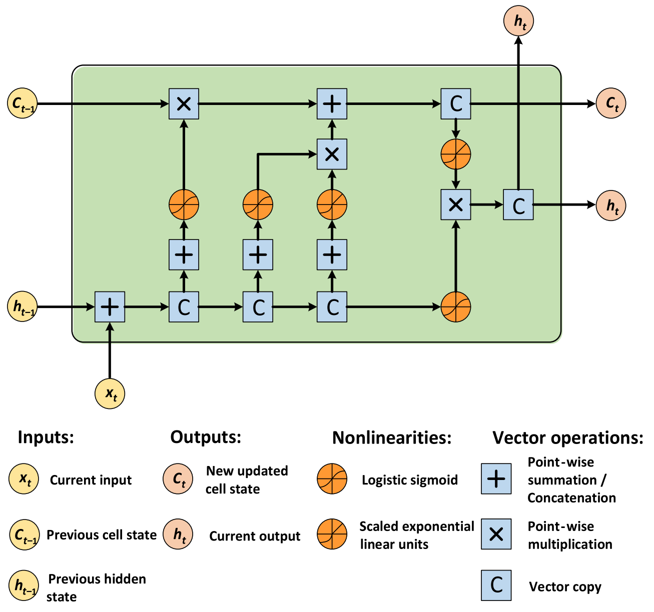

FIGURE 2. The unit structure of the proposed SN-LSTM network.

be observed that the SN-LSTM unit is composed of three gates and a memory cell. The hidden state h_t of the SN-LSTM at time step t is updated as the following equations:

$$h_t = f(x_t, h_{t-1}) = \begin{cases} l_t = SELU(W_l x_t + U_l h_{t-1} + b_l)) \\ i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ C_t = f_t \odot C_{t-1} + i_t \odot l_t \\ o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ h_t = SELU(C_t) \odot o_t \end{cases} \tag{17}$$

where W_l, W_i, W_f, W_o and b_l, b_i, b_f, b_o are input weights and bias weights, respectively. U_l, U_i, U_f, U_o are recurrent weights. SELU and σ are two element-wise nonlinear activation functions. i_t, f_t, C_t, o_t, h_t represents the input gate, the forget gate, the cell state, the output gate and the hidden state at time step t, respectively. l_t denotes the intermediate state that is used to update the cell state.

For all users in the OSNs, they are divided into two groups, which are Sybil and human. In real-world OSNs, there are many correlations between users belonging to the same group and users belonging to different groups that are hidden in the input data. In order to fully extract and utilize the correlations to improve the performance of the classification model, we utilize two separate hidden layers of the bidirectional SN-LSTM (bi-SN-LSTM) to summarize the sequence of feature maps from both directions (i.e., forward and backward). We leverage bi-SN-LSTM instead of the single direction SN-LSTM because of only the bidirectional network can extract the feature information of correlations between a user and all other users. In other words, if we only apply the forward (backward) SN-LSTM, then we can only summarize

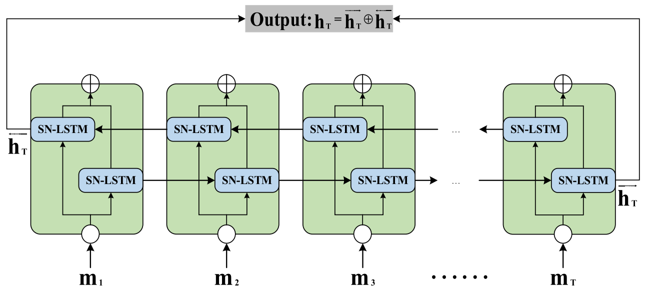
Output: $\mathbf{h}_T = \overrightarrow{\mathbf{h}_T} \oplus \overleftarrow{\mathbf{h}_T}$
SN-LSTM
$\overleftarrow{\mathbf{h}_T}$
$\overrightarrow{\mathbf{h}_T}$
$\mathbf{m}_1$
$\mathbf{m}_2$
$\mathbf{m}_3$
$\mathbf{m}_T$

FIGURE 3. The architecture of the proposed bidirectional SN-LSTM network.

the feature information between a user and other users in front of (behind) it in the input sequence. The lack of some higher feature information will lead to the insufficient feature extraction and adverse impact on the classification results, which will be further demonstrated in Section V. In conclusion, we feed the sequence of feature maps $[\mathrm{m}_1, \mathrm{m}_2, \mathrm{m}_3, \ldots, \mathrm{m}_T]$ (i.e., the output of the self-normalizing CNN structure) into the bi-SN-LSTM to extract higher features as shown in Fig. 3. The bi-SN-LSTM contains a forward SN-LSTM network and a backward SN-LSTM network, and we respectively describe the forward and backward process at time step t as follows,

$$\overrightarrow{h_t} = f(\overrightarrow{x_t}, \overrightarrow{h_{t-1}}) = \begin{cases} \overrightarrow{l_t} = SELU(\overrightarrow{W_l}\overrightarrow{x_t} + \overrightarrow{U_l}\overrightarrow{h_{t-1}} + \overrightarrow{b_l}) \\ \overrightarrow{i_t} = \sigma(\overrightarrow{W_i}\overrightarrow{x_t} + \overrightarrow{U_i}\overrightarrow{h_{t-1}} + \overrightarrow{b_i}) \\ \overrightarrow{f}_t = \sigma(\overrightarrow{W_f}\overrightarrow{x_t} + \overrightarrow{U_f}\overrightarrow{h_{t-1}} + \overrightarrow{b_f}) \\ \overrightarrow{C_t} = \overrightarrow{f_t} \odot \overrightarrow{C_{t-1}} + \overrightarrow{i_t} \odot \overrightarrow{l_t} \\ \overrightarrow{o_t} = \sigma(\overrightarrow{W_o}\overrightarrow{x_t} + \overrightarrow{U_o}\overrightarrow{h_{t-1}} + \overrightarrow{b_o}) \\ \overrightarrow{h_t} = SELU(\overrightarrow{C_t}) \odot \overrightarrow{o_t} \end{cases} \tag{18}$$

$$\overleftarrow{h_t} = f(\overleftarrow{x_t}, \overleftarrow{h_{t-1}}) = \begin{cases} \overleftarrow{l_t} = SELU(\overleftarrow{W_l}\overleftarrow{x_t} + \overleftarrow{U_l}\overleftarrow{h_{t-1}} + \overleftarrow{b_l}) \\ \overleftarrow{i_t} = \sigma(\overleftarrow{W_i}\overleftarrow{x_t} + \overleftarrow{U_i}\overleftarrow{h_{t-1}} + \overleftarrow{b_i}) \\ \overleftarrow{f}_t = \sigma(\overleftarrow{W_f}\overleftarrow{x_t} + \overleftarrow{U_f}\overleftarrow{h_{t-1}} + \overleftarrow{b_f}) \\ \overleftarrow{C_t} = \overleftarrow{f_t} \odot \overleftarrow{C_{t-1}} + \overleftarrow{i_t} \odot \overleftarrow{l_t} \\ \overleftarrow{o_t} = \sigma(\overleftarrow{W_o}\overleftarrow{x_t} + \overleftarrow{U_o}\overleftarrow{h_{t-1}} + \overleftarrow{b_o}) \\ \overleftarrow{h_t} = SELU(\overleftarrow{C_t}) \odot \overleftarrow{o_t} \end{cases} \tag{19}$$

where $\rightarrow$ and $\leftarrow$ denote the forward and backward layer of the proposed bi-SN-LSTM network, respectively.

The outputs of the CNN structure are further fed into the bi-SN-LSTM networks to obtain the forward hidden state $\overrightarrow{h_t}$ and the backward hidden state $\overleftarrow{h_t}$, respectively. Then, the concatenation of $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ is adopted to obtain the complete hidden state h_t of the proposed bi-SN-LSTM network at time step t, shown as below,

$$h_t = \overrightarrow{h_t} \oplus \overleftarrow{h_t} \tag{20}$$

where $t = 1, 2, \ldots, T$, and h_t contains higher feature information from both directions. To output the extraction result of higher features, the hidden states from both directions are collected into a matrix as below,

$$H^o = [h_1, h_2, h_3, \ldots, h_T] \tag{21}$$

where each row of the H^o represents the higher feature extraction result of the corresponding user.

Then, the H^o will be fed into the dense layer to seek a higher-level representation of the feature extraction results of each sample. At a dense layer, each neuron is connected to all the neurons from the previous layer, and also receives input from those neurons. The input vector puts through the dense layer to summarize features extracted from the previous layers of our model. This process can be expressed as:

$$o_i = SELU(W_i \cdot H_i^o + b_i) \tag{22}$$

where H_i^o denotes the $i-th$ row of the H^o (i.e., the feature vector that contains both lower and higher feature extraction results of the $i-th$ user), o_i represents the output vector of the dense layer, the trainable weight matrix and bias vector is denoted as W_i and b_i, respectively, the function $SELU(\cdot)$ means utilize SELU as the activation function of the dense layer. Then, another layer of alpha dropout is utilized to reduce the risk of over-fitting and also working with SELU to preserve the self-normalizing property of the proposed model.

Finally, the output features of the dense layer are fed into the softmax classifier as expressed in Eq. (23) to obtain the distribution of two categories, and then output the final classification result. The output probabilities of the softmax classifier will add up to 1 and each probability are always in the range of [0,1]. We take the category with the maximum probability as the predicted label $\hat{T}_i$.

$$P(y = j|x_i) = \frac{\exp(W_j x_i + b_j)}{\sum_{p=1}^{K} \exp(W_p x_i + b_p)} \tag{23}$$

where K denotes the number of categories, W and b are the trainable parameters of the classification layer, $P(y = j|x_i)$ represents the probability of the input x_i being predicted as category j. Our model is optimized by minimizing the cross-entropy loss between the real label and the predicted label.

V. EVALUATION

In this section, we present the datasets, experimental setup, comparison models, evaluation metrics, experimental results, ablation analysis, and further analysis of the proposed model.

A. GENERAL INTRODUCTION TO THE DATASET

The datasets we use in this paper to test and verify the performance of our proposed model is generated by My Information Bubble (MIB) [37]. The statistical information of the dataset is shown in Table 1. In the MIB dataset, there are a total of five sub-datasets, which are TFP, E13, FSF, INT, and TWT. After performing multiple comparison experiments, work [37] proved that the balanced distribution of Sybils and humans enables the trained classifier to achieve the best classification results. The entire set of the Sybils is randomly under-sampled to match the size of the verified human users. In conclusion, the MIB dataset consists of 1950 Sybils and 1950 human users with a total of 3900 Twitter accounts. Each user in the dataset has multi-dimensional feature information, which will be fed into the proposed model to extract lower features (i.e., the feature information of the input data) and higher features (i.e., the feature information of the correlation between different users hidden in the feature map sequence) by using self-normalizing CNN and the proposed bi-SN-LSTM network, respectively.

TABLE 1. The statistical information of MIB dataset.

Dataset description	MIB dataset				
	TFP	E13	FSF	INT	TWT
Accounts	469	1481	1169	1337	845
Tweets	563,693	2,068,037	22,910	58,925	114,192

B. EXPERIMENTAL SETUP

The z-score normalization method is applied to rescale the raw input data before the training process. The filter size of the first two convolutional layers and the last two convolutional layers are set to 32 and 64, respectively. The size of the max pooling windows is set to (2,2). Moreover, we set 16 as the number of hidden state dimension of the bi-SN-LSTM network. Then, the 256-dimensional output space of the dense layer is utilized to seek a higher-level representation of the extracted features. Furthermore, we take alpha dropout to prevent our model from overfitting and preserve the self-normalizing property of the proposed model during the training process. Furthermore, the gradient-based optimizer Adam [38] with a learning rate of 0.001 is used to minimize the categorical cross-entropy loss between predicted and true distributions. Note that the choice of the hyperparameters (i.e., batch size, training epochs, dropout rate, etc.) are obtained by performing a grid search strategy. We use the Tesla K80 graphics card and 12G memory as the hardware environment. Our model is implemented with Tensorflow.

C. COMPARISON MODELS

The experimental results of the proposed model are compared with other state-of-the-art models, including 1) RandomForest (RF) [28], this model uses several decision trees and assign input data point into each tree. After getting each tree's classification result, it takes the majority vote from all trees as the final output. The number of trees is taken as 100. 2) K-Nearest Neighbors (KNN) [28], this method is a non-parametric classification algorithm and has achieved good results in many classification problems based on supervised learning methods. 2 is taken as the number of neighbors. 3) Adaptive Boosting (Ada) [28], this method is the first step-

TABLE 2. Confusion matrix.

	Predicted Positive (Label=0)	Predicted Negative (Label=1)
Actual Positive (Label=0)	True Positive (TP)	False Negative (FN)
Actual Negative (Label=1)	False Positive (FP)	True Negative (TN)

ping stone of the boosting algorithm and it combines many weak classifiers into a strong classifier to get accurate results. 100 is taken as the number of estimators. 4) XgBoost (Xgb), this method is an ensemble ML algorithm that uses a gradient boosting framework. We consider this method into the comparison experiments to further compare the performance of our proposed model with the most state-of-the-art boosting algorithm. The key parameter n_estimators (i.e., the number of estimators) is set to the same as Ada (i.e., 100).

D. EVALUATION METRICS

Many performance metrics have been introduced to investigate a binary classification problem. These metrics are calculated by those indicators illustrated in the confusion matrix as shown in Table 2. We considered four commonly utilized metrics in this work, which are precision (refer to as purity), recall (refer to as accuracy), f1-score (the harmonic mean between precision and recall) and AUC (area under ROC curve). The first three metrics are respectively expressed in Eq. (24) to (26). Moreover, the value of AUC is obtained by computing the area under the ROC (i.e., receiver operating characteristic) curve, which is plotted by the true-positive rate and false-positive rate (i.e., TPR and FPR, which can be calculated from the confusion matrix as shown in Table 2).

$$Precision = = \frac{TP}{TP + FP} \tag{24}$$

$$Recall = \frac{TP}{TP + FN} \tag{25}$$

$$F1 - score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{26}$$

E. EXPERIMENTAL RESULTS

As demonstrated in the experimental setup (i.e., Section V-B), the proposed model is fed with the normalized input data. To quantify the influence of applying feature scaling method during data preprocessing, and to make a comprehensive comparison of each model, we perform two sets of experiments, one of which utilizes raw input data and the other uses normalized data. As a result, the classification performance of the proposed model compared with other methods is shown in Table 3.

The proposed model with normalized input data significantly outperforms the RF, the KNN, the Ada and the Xgb model. The main reason is that our model extracts lower features (i.e., the feature information that directly extracted from the normalized input data) and long-term dependencies (i.e., the hidden features in the feature map sequence) by self-normalizing CNN and bi-SN-LSTM, respectively. Although the Xgb model with raw input data yields similar precision as that of the proposed model, but our model achieves better results on f1-score and AUC which not only shows that the proposed model achieves more balanced trade-off between the accuracy and purity of classification results, but also demonstrates the better overall performance of our proposed model. The highly hierarchical architecture of the proposed model has more sufficient feature extraction capabilities than other comparison methods. Moreover, the above comparison results demonstrate the necessity of leveraging the feature scaling method to rescale the input data before feeding them into the proposed model.

Compared with the proposed model with raw input data, the proposed model with normalized data achieves better results. The main reason is that after applying the z-score normalization method, the rescaled data has the distribution with a standard deviation of 1 and a mean of 0, which provides the model with self-normalizing property from the beginning of the training process.

F. ABLATION ANALYSIS

As demonstrated in Section IV, the self-normalizing CNN and bi-SN-LSTM are utilized to automatically extract lower and higher features of the normalized input data, respectively. In this section, we conduct ablation studies to quantify the influence of those two feature extraction components in the proposed model. Note that the stacked two dense layers are applied in the following models to seek a higher-level

TABLE 3. Experimental results on Precision, Recall, F1-score and AUC (the best results are in bold).

Metrics	Raw Input Data					Normalized Input Data				
	RF	KNN	Ada	Xgb	Proposed Model	RF	KNN	Ada	Xgb	Proposed Model
Precision	0.9831	0.9216	0.9872	0.9915	0.9516	0.9233	0.8956	0.6051	0.9160	**0.9999**
Recall	0.9708	0.9792	0.9667	0.9750	0.9833	0.9651	0.9292	0.8875	0.9542	**0.9863**
F1-score	0.9769	0.9495	0.9768	0.9832	0.9672	0.9397	0.9121	0.7196	0.9347	**0.9931**
AUC	0.9712	0.9187	0.9727	0.9804	0.9491	0.9899	0.8724	0.4508	0.9026	**0.9932**

1. The comparison results are obtained by using exactly the same parameters as demonstrated in their original paper.

VOLUME XX, 2017

TABLE 4. Ablation studies on different components of the proposed model.

Methods	Precision	Recall	F1-score	AUC
Lower-only	0.9949	0.8125	0.8945	0.9027
Higher-only	0.9899	0.8167	0.8951	0.9012
Uni-higher	0.9951	0.8458	0.9144	0.9193
Self/Uni	0.9953	0.8833	0.9361	0.9381
Proposed model	**0.9999**	**0.9863**	**0.9931**	**0.9932**

representation of the extracted features and finally output the classification results.

"Lower-only": This model extract features by using the self-normalizing CNN structure only. In other words, the bi-SN-LSTM in Fig. 1 is not applied in this model, and the higher-level representation (i.e., the output of the dense layer) is only determined by the lower feature extraction result.

"Higher-only": A model that only using bi-SN-LSTM to automatically extract higher features hidden in the input data. We obtain the classification result without leveraging the lower features.

"Uni-higher": A model that is similar to the "Higher-only" model but utilizing the unidirectional SN-LSTM instead of the bi-SN-LSTM to extract higher features, which will lead to a part of higher feature information to be missing.

"Self/Uni": The architecture of this model is the same as the proposed model but using the unidirectional SN-LSTM to replace the bi-SN-LSTM network.

The comparison results between the proposed model and other ablation models are shown in Table 4. We utilize four metrics (i.e., precision, recall, f1-score, and AUC) to evaluate the classification results and conduct further analysis. The proposed model significantly outperforms "Lower-only" which demonstrates the necessity of leveraging higher features to accurately classify Sybils. Moreover, the performance of "Higher-only" drops compared with the proposed model, which illustrates the positive impact of lower features on classification results. Comparing the "Uni-higher" and the "Self-Uni" with the proposed model, the performance drops not only further demonstrates the effectiveness of two feature extraction components in the proposed model, but also proves that the bi-SN-LSTM network owns the better feature extraction ability than unidirectional SN-LSTM network. The model "Self-Uni" leverages a single SN-LSTM layer to summarize the higher features from the feature map sequence, which will cause the partial loss of the higher features. To address this problem, the proposed model utilizes bi-SN-LSTM to extract the higher features between a user and all users that precede and follow it in the same sequence (i.e., summarize feature map sequence from both directions). In conclusion, the proposed model significantly outperforms all four of the other ablation models by utilizing self-normalizing CNN and bi-SN-LSTM to fully extract lower and higher features of the input data, respectively. The effectiveness of the bi-SN-LSTM will be further demonstrated in Section V-G.

TABLE 5. Comparison results of the proposed bi-SN-LSTM method and other commonly utilized RNN-based methods.

Methods	Precision	Recall	F1-score	AUC	Time
RNN	0.9326	0.7501	0.8314	0.8289	**829**
bi-RNN	0.9951	0.8501	0.9169	0.9215	871
GRU	0.9902	0.8458	0.9124	0.9158	883
bi-GRU	0.9903	0.8542	0.9172	0.9199	941
LSTM	0.9952	0.8625	0.8924	0.8992	841
bi-LSTM	0.9950	0.8922	0.9045	0.9110	1026
bi-SN-LSTM	**0.9999**	**0.9863**	**0.9931**	**0.9932**	1060

G. ANALYSIS OF THE PROPOSED MODEL

1) ANALYSIS OF THE PROPOSED bi-SN-LSTM MODEL

To validate the effectiveness and evaluate the performance of the proposed bi-SN-LSTM network, we compare the performance of bi-SN-LSTM to other commonly utilized RNN-based methods, including vanilla RNN, bidirectional RNN (bi-RNN), gated recurrent unit (GRU), bidirectional GRU (bi-GRU), LSTM and bidirectional LSTM (bi-LSTM). We utilize those RNN-based methods to replace the bi-SN-LSTM in Fig. 1 to extract higher features. Note that those methods have similar network hyperparameters and parameters as the proposed bi-SN-LSTM model.

The comparison result is shown in Table 5, the proposed model achieves the best classification performance compared to other models. Comparing with bi-LSTM, the proposed model (i.e., the bi-SN-LSTM network) significantly improves the Sybil classification performance with the exponentially grown hidden state and the sparseness property provided by the recurrent step of each SN-LSTM unit with SELU as its activation function. A similar conclusion can be concluded by comparing the performance of the proposed model with bi-RNN and bi-GRU in Table 5.

According to the "LSTM" row of Table 5, the performance drops compared with the bi-LSTM which shows the better performance of the model when utilizing a bidirectional RNN-based network for higher feature extraction. The same phenomenon can also be found when comparing the RNN method and bi-RNN method, GRU method, and bi-GRU method. The above two sets of comparison experiment and the experimental results in Section V-F (i.e., the comparison results between "Self/Uni" and the proposed model) jointly further proves the effectiveness of leveraging bidirectional SN-LSTM (i.e., bi-SN-LSTM) as the higher feature extractor of the proposed model. Moreover, the comparison result between the proposed model and the bi-LSTM network shows that SELU is a more appropriate activation function for the recurrent step of the LSTM network when dealing with the Sybil detection problem.

According to the "Time" column of Table 5, the training efficiency of the vanilla RNN method outperforms other methods because of the simplest mathematical model behind it. Note that the time of different comparison models is obtained after training 100 epochs. However, the classification performance of the vanilla RNN method is the worst among all the methods, which means it cannot be applied to the real-world Sybil detection problems.

Compared with their bidirectional variants, the vanilla RNN, GRU, and LSTM have a more efficient training process, but the classification performance of the model is decreased. This comparison result shows the trade-off problem between the training efficiency and the classification performance when utilizing RNN-based methods for Sybil classification in OSNs. As shown in the "bi-SN-LSTM" row of Table 5, the training time of our proposed model increase because of the exponentially grown hidden state of the proposed bi-SN-LSTM network. However, our model achieves the best classification results while keeping the training efficiency close to those RNN-based methods with a sophisticated mathematical background (i.e., bi-GRU and bi-LSTM). In conclusion, the proposed model not only outperforms all other RNN-based methods but also achieves a more balanced trade-off between the training efficiency and the classification performance. Moreover, when dealing with the complex real-world Sybil detection problem, we can utilize the graphics processing unit to increase the computation efficiency of the proposed model based on the bi-SN-LSTM network.

2) EFFECTS OF THE NUMBER OF CONVOLUTIONAL LAYERS AND THE FEATURE SCALING METHOD

This section conducts a comparison experiment to quantify the influence of the feature scaling (i.e., data normalization) method and the number of convolutional layers in the proposed model. To better utilize the lower features of the input data, we considered different amounts of convolutional layers into our comparison experiments. Moreover, applying the appropriate feature scaling (i.e., data normalization) method will have a significant influence on the classification results. Therefore, three commonly utilized feature scaling methods are considered into our comparison experiments as follows,

- **Min-Max Scaling:** transform the data such that features are within the specified range of [0.1]:

$$x'_{(m,n)} = \frac{x_{(m,n)} - x_n^{min}}{x_n^{max} - x_n^{min}} \tag{27}$$

- **Standardization** (i.e., z-score normalization method): transform the data such that the resulting distribution has a standard deviation of 1 and a mean of 0, expresses as below:

$$x'_{(m,n)} = \frac{x_{(m,n)} - \mu_n}{\sigma_n} \tag{28}$$

- **Mean Normalization:** transform the data such that they can be described as a normal distribution (i.e., bell curve) and all the values are within the range of [0,1]:

$$x'_{(m,n)} = \frac{x_{(m,n)} - \mu_n}{x_n^{max} - x_n^{min}} \tag{29}$$

where m is the $m-th$ user of the input data, n denotes the $n-th$ input features, x_n^{max} and x_n^{min} are the maximum and minimum value of the $n-th$ feature, respectively. $x_{(m,n)}$ denotes raw input data, $x'_{(m,n)}$ represents the scaled input data, μ_n and σ_n denote the value of the mean and standard deviation of the $n-th$ feature, respectively.

The comparison results in Fig. 4 show that the model obtains the best result on precision, F1-score and AUC, when the number of convolutional layers is set as four with z-score normalization method, is applied. Although the best result on recall can be obtained by utilizing five convolutional layers and mean normalization, it has worse results on the other three metrics (i.e., precision, f1-score, and AUC). It can be concluded from Fig. 4 that the z-score normalization method significantly outperforms the other two methods (i.e., mean normalization and min-max scaling), which shows that the standardization method is the most appropriate feature scaling method for scaling the raw input data before feeding them into the proposed model. Moreover, the best results on f1-score and AUC is obtained when four convolutional layers are applied in the proposed model. The higher value of f1-score indicates the model reaches the more balanced trade-off between precision and recall, which provides better robustness of the classification model. Furthermore, the better results on the AUC metric means the better overall performance of the proposed model. Moreover, the precision value significantly increases when applying z-score normalization and four convolutional layers into our model, which means that few human users are mistakenly classified as Sybils in the classification results.

For each convolutional layer, it contains a certain number of filters to create feature maps for the model to extract and compress lower features of the input data. Comparing the performance of 4 convolutional layers with other variants that use more convolutional layers, it can be obtained that the classification results do not improve with the increasing number of convolutional layers, which shows that the over-compressed feature information will decrease the performance of the classification model. The above results demonstrate the effectiveness of utilizing z-score normalization as the feature scaling method and four convolutional layers as the lower feature extractor in the proposed model.

3) INFLUENCES OF THE DIMENSION OF HIDDEN STATES

Hidden states of the proposed bi-SN-LSTM is to extract and store the higher feature information from the feature map sequence. The number of hidden states directly affects the classification performance of the proposed model. We conduct experiments to quantify the influence of the dimension of hidden states with varied numbers (i.e., 2, 4, 8, 16, 32, 64, 128, 256). Note that other hyperparameters remain unchanged. The experimental results are shown in Fig. 5 and Fig. 6. Obviously, the proposed model with 16-dimensional hidden states significantly outperforms other variants. Moreover, when the dimension of hidden states increases from 2 to 8, the precision rise while the recall, f1-score, and AUC decrease. The

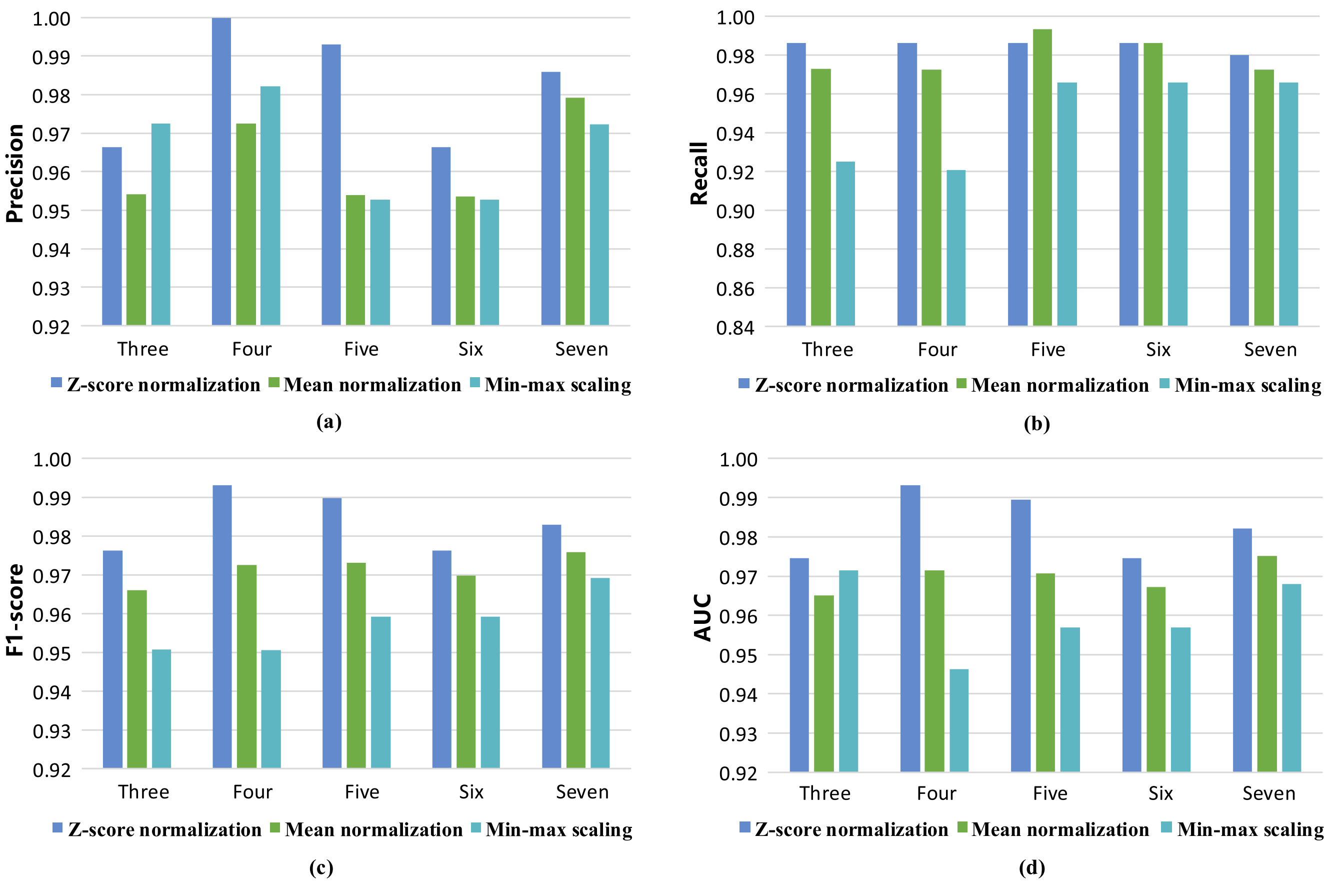

FIGURE 4. Performance of the proposed model with different number of convolutional layers and feature scaling methods. (a) Precision. (b) Recall. (c) F1-score. (d) AUC. The value on the x-axis represents the number of convolutional layers.

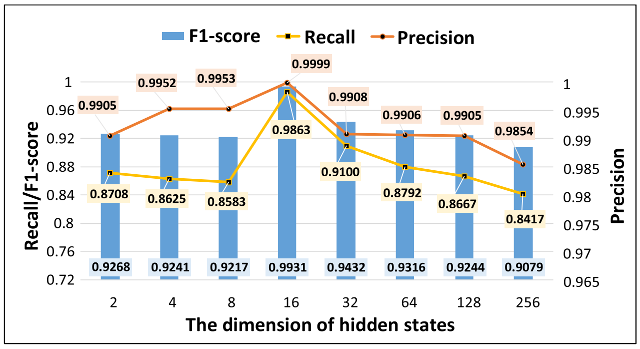

FIGURE 5. Precision, Recall and F1-score of the proposed model with different dimension of hidden states.

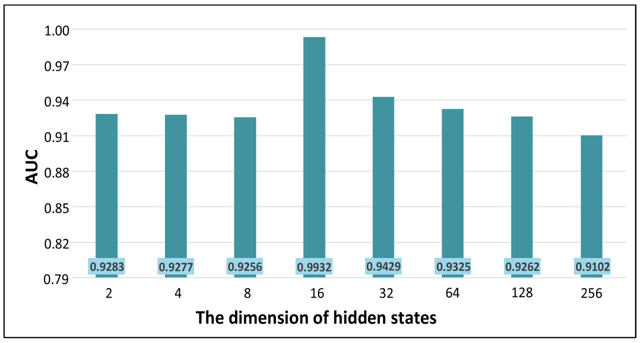

FIGURE 6. AUC of the proposed model with different dimension of hidden states.

performance drops when the dimension of hidden states is larger than 16 (e.g., 64).

The main reason is that the model suffers from over-fitting and becomes too complicated for the dataset. When the dimension of hidden states is too large, our model will take a high risk of over-fitting even if alpha dropout is applied in the proposed model. Meanwhile, the feature extraction capability may be insufficient when the number of hidden states is too small. In conclusion, we choose 16 as the dimension of hidden states in this paper to make our model achieve better classification results.

VI. CONCLUSION

In this work, we propose a content-based end-to-end model to achieve a more accurate Sybil detection results in online social networks. The hierarchical architecture enables the proposed model to better utilize the feature information. Moreover, our model enjoys the advantage of saving human effort to design features. The performance of the proposed model is greatly improved by utilizing self-normalizing CNN and bi-SN-LSTM to extract lower and higher features of the input data, respectively. We utilize the combination of SELU and alpha dropout for preserving the self-normalizing property and relieving the overfitting problem along the training process. The bi-SN-LSTM is developed to summarize the feature map sequences (i.e., the output of CNN structure) from both directions to extract hidden feature information (e.g., the correlation between users) for classification, which is more effective than other commonly utilized RNN-based methods. Through the case study of MIB dataset, the experimental results demonstrate that our model outperforms the state-of-the-art methods and achieves an excellent classification performance. Additionally, we perform ablation analysis, and analysis of the proposed model to further prove the effectiveness of the classification model in this work. In our future work, the graph neural network-based method will be considered to classify Sybils with structure-based features.

ACKNOWLEDGMENT

The authors would like to convey our grateful appreciation to the corresponding author of this paper, Jin Yang, who has offered advice with huge values in all stages when writing this essay to us. The authors would like to thank My Information Bubble (MIB) for sharing their datasets for evaluation.

REFERENCES

[1] J. Clement. *Twitter—Statistics & Facts*. Accessed: Jan. 10, 2019. [Online]. Available: https://www.statista.com/topics/737/twitter/

[2] B. Wang, J. Jia, L. Zhang, and N. Z. Gong, "Structure-based sybil detection in social networks via local rule-based propagation," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 3, pp. 523–537, Jul. 2019.

[3] K. Thomas, C. Grier, D. Song, and V. Paxson, "Suspended accounts in retrospect: An analysis of Twitter spam," in *Proc. ACM SIGCOMM Conf. Internet Meas. Conf. (IMC)*, 2011, pp. 243–258.

[4] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song, "Design and evaluation of a real-time URL spam filtering service," in *Proc. IEEE Symp. Secur. Privacy*, May 2011, pp. 447–462.

[5] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda, "All your contacts are belong to us: Automated identity theft attacks on social networks," in *Proc. 18th Int. Conf. World Wide Web (WWW)*, 2009, pp. 551–560.

[6] G. Yan, G. Chen, S. Eidenbenz, and N. Li, "Malware propagation in Online social networks: Nature, dynamics, and defense implications," in *Proc. 6th ACM Symp. Inf., Comput. Commun. Secur. (ASIACCS)*, 2011, pp. 196–206.

[7] H. Yu, M. Kaminsky, P. B. Gibbons, and A. D. Flaxman, "SybilGuard: Defending against sybil attacks via social networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 576–589, Jun. 2008.

[8] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao, "SybilLimit: A near-optimal social network defense against sybil attacks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 885–898, Jun. 2010.

[9] G. Danezis and P. Mittal, "Sybilinfer: Detecting sybil nodes using social networks," in *Proc. NDSS*, 2009, pp. 1–15.

[10] N. Tran, J. Li, L. Subramanian, and S. S. M. Chow, "Optimal sybil-resilient node admission control," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 3218–3226.

[11] Q. Cao *et al.*, "Aiding the detection of fake accounts in large scale social online services," presented at the 9th USENIX Symp. Netw. Syst. Design Implement. (NSDI), 2012, pp. 197–210.

[12] Q. Cao and X. Yang, "SybilFence: Improving social-graph-based sybil defenses with user negative feedback," 2013, *arXiv:1304.3819*. [Online]. Available: http://arxiv.org/abs/1304.3819

[13] W. Wei, F. Xu, C. C. Tan, and Q. Li, "SybilDefender: Defend against sybil attacks in large social networks," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 1951–1959.

[14] Z. Yang, J. Xue, X. Yang, X. Wang, and Y. Dai, "VoteTrust: Leveraging friend invitation graph to defend against social network sybils," in *Proc. IEEE INFOCOM*, Apr. 2016, pp. 2400–2408.

[15] Z. Yang, J. Xue, X. Yang, X. Wang, and Y. Dai, "VoteTrust: Leveraging friend invitation graph to defend against social network sybils," *IEEE Trans. Depend. Sec. Comput.*, vol. 13, no. 4, pp. 488–501, Jul. 2016.

[16] Y. Boshmaf, D. Logothetis, G. Siganos, J. Leria, J. Lorenzo, M. Ripeanu, and K. Beznosov, "Integro: Leveraging victim prediction for robust fake account detection in OSNs," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2015, pp. 8–11.

[17] S. Misra, A. S. M. Tayeen, and W. Xu, "SybilExposer: An effective scheme to detect sybil communities in Online social networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

[18] L. Shi, S. Yu, W. Lou, and Y. T. Hou, "SybilShield: An agent-aided social network-based sybil defense among multiple communities," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 1034–1042.

[19] H. Bansal and M. Misra, "Sybil detection in Online social networks (OSNs)," in *Proc. IEEE 6th Int. Conf. Adv. Comput. (IACC)*, Feb. 2016, pp. 569–576.

[20] B. Wang, L. Zhang, and N. Z. Gong, "SybilSCAR: Sybil detection in online social networks via local rule based propagation," in *Proc. IEEE Conf. Comput. Commun.*, May 2017, pp. 1–9.

[21] N. Z. Gong, M. Frank, and P. Mittal, "SybilBelief: A semi-supervised learning approach for structure-based sybil detection," *IEEE Trans. Inf. Forensics Secur.*, vol. 9, no. 6, pp. 976–987, Jun. 2014.

[22] X. Zhang, H. Xie, and J. C. S. Lui, "Sybil detection in social-activity networks: Modeling, algorithms and evaluations," in *Proc. IEEE 26th Int. Conf. Netw. Protocols (ICNP)*, Sep. 2018, pp. 44–54.

[23] G. Wang, "You are how you click: Clickstream analysis for sybil detection," in *Proc. 22nd USENIX Secur. Symp.*, 2013, pp. 241–256.

[24] M. Alsaleh, A. Alarifi, A. M. Al-Salman, M. Alfayez, and A. Almuhaysin, "TSD: Detecting sybil accounts in Twitter," in *Proc. 13th Int. Conf. Mach. Learn. Appl.*, Dec. 2014, pp. 463–469.

[25] K. Kang, "Compound approach for sybil users detection in social networks," *Comput. Sci.*, vol. 43, no. 1, pp. 172–177, 2016.

[26] Y. Xia, L. Pan, L. Shi, and F. Zou, "Attribute credibility based sybil goup detection in Online social networks," in *Proc. IEEE 1st Int. Conf. Data Sci. Cyberspace (DSC)*, Jun. 2016, pp. 358–363.

[27] Z. Xu, B. Chen, X. Meng, and L. Liu, "Towards efficient detection of sybil attacks in location-based social networks," in *Proc. IEEE Symp. Ser. Comput. Intell.(SSCI)*, Nov. 2017, pp. 1–7.

[28] D. Mulamba, I. Ray, and I. Ray, "On sybil classification in Online social networks using only structural features," in *Proc. 16th Annu. Conf. Privacy, Secur. Trust (PST)*, Aug. 2018, pp. 1–10.

[29] M. Al-Qurishi, M. Alrubaian, S. M. M. Rahman, A. Alamri, and M. M. Hassan, "A prediction system of sybil attack in social network using deep-regression model," *Future Gener. Comput. Syst.*, vol. 87, pp. 743–753, Oct. 2018.

[30] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, p. 60, Jul. 2019.

[31] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3D object detection methods for autonomous driving applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3782–3795, Oct. 2019.

[32] R. Wang, Z. Li, J. Cao, T. Chen, and L. Wang, "Convolutional recurrent neural networks for text classification," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–6.

[33] G. Klambauer, "Self-normalizing neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 971–980.

[34] L. Peel, "Data driven prognostics using a Kalman filter ensemble of neural network models," in *Proc. Int. Conf. Prognostics Health Manage.*, Oct. 2008, pp. 1–6.

[35] N. Srivastava, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[36] F. Godin, J. Degrave, J. Dambre, and W. De Neve, "Dual rectified linear units (DReLUs): A replacement for tanh activation functions in quasi-recurrent neural networks," *Pattern Recognit. Lett.*, vol. 116, pp. 8–14, Dec. 2018.

[37] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "Fame for sale: Efficient detection of fake Twitter followers," *Decis. Support Syst.*, vol. 80, pp. 56–71, Dec. 2015.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

TIANYU GAO is currently pursuing the bachelor's degree in software engineering with Sichuan University, Sichuan, China.

His main research interests include network security, data mining, machine learning, and deep learning.

Mr. Gao presided over the Chinese University Student Innovation Training Program at the national level. He won the Outstanding Student Award of IBM-ICBC summer camp, in 2018. He received the highest score for the summer workshop hosted by the School of Computing (SOC), National University of Singapore (NUS), in 2019.

JIN YANG received the M.S. and Ph.D. degrees in computer science from Sichuan University, Sichuan, China, in 2004 and 2007, respectively.

He is currently an Associate Researcher with the College of Cybersecurity, Sichuan University, China. His main research interests include network security, knowledge discovery, and expert systems.

WENJUN PENG is currently pursuing the bachelor's degree in software engineering with Sichuan University, Sichuan, China.

His current research interests include network security, deep learning, data mining, and three-dimensional reconstruction.

LUYU JIANG is currently pursuing the bachelor's degree in software engineering with Sichuan University, Sichuan, China.

His current research interests include network security, deep learning, and data mining.

Mr. Jiang has participated in several programs regarding deep learning.

YIHAO SUN is currently pursuing the bachelor's degree in software engineering with Sichuan University, Chengdu, China.

His current research interests include machine learning, deep learning, and network security.

FANGCHUAN LI is currently pursuing the bachelor's degree in software engineering with Sichuan University, Sichuan, China.

His current research interests include network security, data mining, and three-dimensional reconstruction.