Hindawi
Security and Communication Networks
Volume 2019, Article ID 6417407, 9 pages
https://doi.org/10.1155/2019/6417407

Research Article

Multifeature Named Entity Recognition in Information Security Based on Adversarial Learning

Han Zhang,[1,2] **Yuanbo Guo**,[1] **and Tao Li**[1]

[1]*Information Engineering University, Zhengzhou 450001, China*
[2]*Zhengzhou University, Zhengzhou 450000, China*

Correspondence should be addressed to Han Zhang; zhang_han@zzu.edu.cn

Received 5 November 2018; Accepted 6 February 2019; Published 24 February 2019

Guest Editor: Pelin Angin

In order to obtain high quality and large-scale labelled data for information security research, we propose a new approach that combines a generative adversarial network with the BiLSTM-Attention-CRF model to obtain labelled data from crowd annotations. We use the generative adversarial network to find common features in crowd annotations and then consider them in conjunction with the domain dictionary feature and sentence dependency feature as additional features to be introduced into the BiLSTM-Attention-CRF model, which is then used to carry out named entity recognition in crowdsourcing. Finally, we create a dataset to evaluate our models using information security data. The experimental results show that our model has better performance than the other baseline models.

1. Introduction

Named entity recognition (NER) aims to extract various types of entities from text. This is a fundamental step in text mining and has received much attention recently, especially in medicine [1–6] and biochemistry [7–10]. In contrast, the development of NER tasks in information security has been relatively slow. In previous works, several methods have been proposed for extracting vulnerabilities and extracting information from unstructured texts [11–14]. In the past two years, research into NER has basically entered a stagnant state in the domain of information security. The lack of large-scale labelled data in this field is one of the main reasons for this situation.

Snow et al. proposed a way to quickly and cost-effectively obtain large-scale labelled data using Amazon Mechanical Turk [15] and demonstrated that nonexpert annotations were relatively useful for training models [16]. We can use crowdsourcing as an effective way of obtaining large-scale labelled data at low cost within a short time. However, in a professional field, crowd annotations may be of lower quality than those of experts, and we therefore need to integrate high-quality consensus labelling from crowdsourcing annotations. Although we can obtain high-quality annotations using the majority vote method [17], this requires a great deal of manpower, and for some sentences or entities, whose meanings are rather ambiguous, it may be difficult to reach an agreement among the annotators. A generative adversarial network (GAN) has the ability to generate data. Goodfellow et al. proved theoretically that when the GAN model converges, the generated data have the same distribution as the real data [18]. Yang et al. demonstrated the usability of the GAN model for NER using Chinese crowd-sourced annotations [19]. However, the focus of this work is on using the GAN model to find the features of the trust annotators, and its application is mainly in the general domain.

In practice, there is a significant difference between entity types used in general applications and those used in information security. Some entity categories in the information security domain are not simply nouns or nominal phrases, as traditionally defined in NER. For example, consider the text: [a Trojan] "known as 'Bicololo' was first discovered in October 2012 and specially designed to steal login credentials". In this sentence, the phrasal verb "steal login credentials" should be extracted as an entity of the consequence class of the unified cybersecurity ontology (UCO) [20]. Therefore, when

we identify named entities in the information security field, we need to use certain supplementary features in addition to the traditional (word and character) features.

In this paper, we propose a new model entitled BiLSTM-Attention-CRF-Crowd to improve the quality of the crowdsourcing annotations in information security. Our goal is to combine a GAN model with the BiLSTM-Attention-CRF model. The GAN model is used to find the common features of annotations in order to integrate the best unique consensus annotations and then pass them to the BiLSTM-Attention-CRF submodel as one type of additional feature. Here, we add an attention layer between the BiLSTM and CRF layers, primarily to process long sentences appearing in the text. A neural network model using a conventional encoder-decoder structure is needed to represent the information in the input sequence as a fixed-length vector; it is difficult to retain all the necessary information when the input sequence is long, especially when the length of the input sequence is longer than the length of the sequence in the training data set, and we therefore added the attention layer to address this limitation. The submodel performs NER again on the crowdsourced annotations for the information security data to improve the quality of these annotations. The main contributions of our work can be summarised as follows:

(I) In order to solve the problem of a lack of high-quality annotated data in the field of information security, a new model called BiLSTM-Attention-CRF-crowd is proposed to improve the quality of the crowdsourcing dataset by combining a GAN model with the BiLSTM-Attention-CRF model.

(II) Due to the diversity and specificity of the entity categories in information security, only basic features such as word and character features are used as input for the BiLSTM-Attention-CRF model, which cannot meet the requirements for NER tasks in this field. Based on this, domain dictionary features and sentence dependency features are introduced. These are used as additional features along with the common features learned by the GAN model. The experimental results show that these additional features have practical value for improving the performance of the model.

2. Related Works

2.1. GAN. Compared with its applications in computer vision, the GAN model is less widely used in the field of language processing because the values used in images and video are continuous, and the generator and discriminator can be directly trained using the gradient descent method; in contrast, letters and words in text are all discrete, and the gradient descent method cannot be directly applied. Zhang et al. proposed the TextGAN model, which uses several techniques to deal with discrete variables [21]. For example, it uses a smooth approximation to approximate the discrete output of the LSTM and feature matching techniques in the generator training process. Since the number of parameters in the LSTM is significantly greater than that for the CNN, the LSTM is more difficult to train, and the TextGAN discriminator (CNN) only updates once after the generator (LSTM) has been updated multiple times. Yu et al. proposed SeqGAN, which draws on the concept of reinforcement learning to deal with the discrete output problem. It regards the error in the discriminator output as the reward value in reinforcement learning and regards the training process of the generator as a decision-making process in reinforcement learning. This model is applied to speech text and music generation [22]. Li et al. and Kusner et al. applied GAN to open dialogue text generation and context-free grammar (CFG), respectively [23, 24]. We mainly draw on the method of processing of discrete variables and its objective function for feature matching in [21].

2.2. Crowdsourcing. David et al. presented a confusion matrix for each annotator, using an expectation maximisation (EM) estimation of these matrices as parameters and the true token labels as hidden variables [25]. Dredze et al. proposed a conditional random field (CRF) model for learning from multiple annotations, but the features that the CRF can learn are limited [26]. In previous work [19, 27, 28], the aim was to model the differences between the annotators and to extract the more trustworthy annotators. Although this improves the performance of the model, this choice of annotation is too dependent on the credibility of an annotator.

2.3. NER in Information Security. In order to solve the problem of a lack of large-scale labelled data, Rodrigo Agerri et al. designed a projection algorithm to transport NER annotations across languages [29]. However, in information security, there are no scaled annotations sets in any language. Giorgi et al. combined gold-standard corpora with silver-standard corpora by using transfer learning to expand the scale of high-quality labels [30]. However, this is applied in the field of biology.

Few documents from the last three years can be found on the subject of NER in information security. A small number of works have focused on extracting vulnerabilities and attack information from unstructured texts in the past few years [11–14]. Bridges et al. proposed a maximum entropy model trained with an averaged perceptron to extract entities from text, but these authors extracted only the entities and did not classify the types of these entities [11]. Weerawardhana et al. extracted vulnerability information from an online vulnerability database [12]. Lal proposed a CRF to extract vulnerabilities from the text [13]. Mulwad et al. used wikitology to extract vulnerabilities and attacks from web text, although wikitology is an ontology in the general domain [14].

3. Feature Selection

A high-quality feature set is key to the success of NER tasks in information security. In this paper, we use word and character features as our basic feature set and others (such as domain dictionary and sentence dependency features) as two kinds of additional features.

3.1. Word Features. Word features (word embedding) involve a distributed representation of a word in vector space. This can capture semantic and grammatical information about a word from an unlabeled corpus [31]. Words with similar context or semantics are closer in the word vector space, and word vectors can therefore be used for natural language processing tasks such as entity classification, entity alignment, and relation extraction. Word2vec [32] is the most commonly used tool for training word vectors. In this paper, in order to obtain high-quality word vectors, we select 94,534 vulnerability record descriptions from entries in the Common Vulnerabilities and Exposures (CVE) corpus [33] listed since 1997 for word vector training.

3.2. Character Features. Character features contain structural information about the name of the entity and can represent the specific composition of the entity's name, especially in the information security domain. For example, viruses such as Backdoor.Win32.Gpigeon.pd and Backdoor.Win32.Gpigeon2010.pc, which are PE viruses affecting Windows, have the same prefix; hence, when we encounter one these words, we know that it is the name of a PE virus for windows, based on its prefix. Unlike traditional manually designed character features, we can obtain the character feature vectors for words through training. First, the character vector of each character in the word is obtained by querying the character table, and then the character vector corresponding to the word is used as input for the BiLSTM.

3.3. Additional Features

3.3.1. Domain Dictionary Feature. In information security, in order to better identify entities, it is not sufficient to use only word and character features; we also need to add domain knowledge, including features such as a domain dictionary. At present, there is no established dictionary for reference in information security, so we can use the Internet to construct a preliminary domain dictionary. In this paper, we use Wikipedia as a corpus and the UCO in the information security domain to construct a domain dictionary. Wikipedia contains three types of page: an entry page, no_created entry page, and a list page. Entry pages are mainly used to describe concepts. In Wikipedia, the URLs of these three types of pages follow certain rules. For example, the URL for the entry page is usually in the form http://en.wikipedia.org/wiki/∗, and the URL for the list page is usually in the form n.wikipedia.org/wiki/Category. In computer-related fields, we therefore only need to use en.wikipedia.org/wiki/Category: computing as a crawler input for concept capture. After the concept is captured, k-means clustering is carried out. There are eight important concept classes in UCO, and k can therefore be set to eight for k-means clustering.

The resulting clustering categories are labelled by hand and then reclassified using the Mahalanobis distance. The specific algorithms are as follows:

(I) We have a set of classes $\{c_{1,}c_{2,\ldots}c_k\}$ where k is the number of classes, and $\{x_a^1, x_a^2, \ldots x_a^n\}$ expresses the original concept vectors in class C_a in UCO. Then, the mean vector μ_a and covariance matrix $\sum_a \mathrm{f}\, C_a$ are as follows:

$$\mu_a = \frac{1}{n}\sum_n x_a^i$$
$$\sum_a = \frac{\sum_{i=1}^n \left(x_a^i - u_a\right)}{n-1} \tag{1}$$

(II) If w_i represents the vector of the subclass of C_a after clustering, then the semantic similarity is calculated by the Mahalanobis distance:

$$D_m\left(w_i, c_a\right) = \sqrt{\left(w_i - u_a\right)^T \sum_a^{-1} \left(w_i - u_a\right)} \tag{2}$$

The clustering result is determined again. We set the threshold y; if $D_m(w_i, c_a) \geq y$, then the concept is considered not to belong to the current concept category. The initial value of the threshold y can be obtained from the original concept under the training category.

(III) If a concept has not been classified into a category after filtering, it can be judged based on its upper and lower concept in Wikipedia. If its upper and lower concept are not in the same category at this time, then the distance between the upper (lower) concept and the class centre of the category is calculated separately. We select a category with a smaller distance as its category. In contrast, we put the concept into the class to whose upper and lower concept belong.

(IV) Before classifying a new concept into a category, the algorithm returns to Step I to recalculate the mean vector and covariance matrix for the category.

3.3.2. Sentence Dependency Feature. As discussed above, the entity types in information security are no longer simply the types defined in traditional NER tasks.

We use the example given above to analyse the annotation of the semantic role and syntactic dependencies. We use Stanford CoreNLP [34] as a syntactic analysis tool. The result of this syntactic analysis is shown in Figure 1.

The core verbs in this sentence are "discovered" and "designed", and "Bicololo" is the passive subject of these two verbs. The subject of this sentence is a worm virus from the attacker class. If we want to extract the entity of the consequence class, we should focus on the modifier followed by "designed", for which the phrase "steal login credentials" is the open clausal complement. The open clausal complement is a verb or a verb phrase which is used to add a description of the core verb. At this point, we can create the following rules for extracting consequence class entities:

(I) The subject of the sentence should be the type of attacker entity.

(II) We consider the verbs associated with each attacker entity, such as "be designed to/for", "be used to",

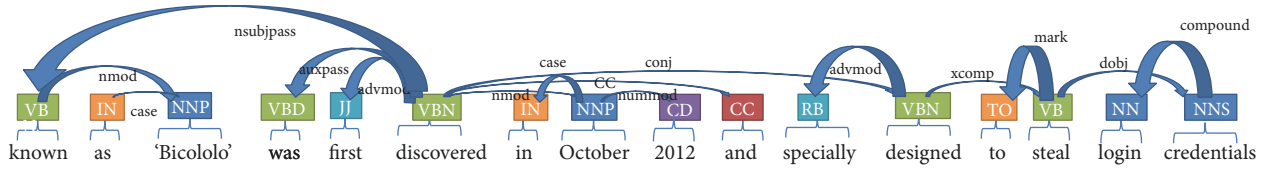

Figure 1: Sentence dependency analysis.

"result", "cause", and other predicate verbs and extract the minimum verb phrases or clauses associated with these. Here, the minimum verb phrase or clause is a phrase or clause that does not contain any nested or identical type.

(III) If the relationship between the minimum phrase or clause and the foregoing predicate verbs is that of a complement or modifier, the minimum phrase or clause can be considered as an entity of the consequence class.

4. Model Design for BiLSTM-Attention-CRF-Crowd

Our model focuses on two tasks. Task 1 is the generation of crowd annotations through adversarial learning in order to integrate the optimal single-consensus annotations. In Task 2, the common features generated in Task 1 are used in conjunction with the domain dictionary features and the sentence dependency features and are input to the BiLSTM-Attention-CRF submodel for renamed entity recognition of the crowdsourcing annotations in order to improve their quality. The architecture of our model is illustrated in Figure 2.

The model consists of two submodels. Figure 2(a) shows the GAN model, which consists of the generator BiLSTM-Attention and the discriminator CNN. The crowd annotations are used as input for the generator to form new feature representations after being processed by the BiLSTM-Attention layer. The new feature representations are passed to the CNN, which is trained using the expert annotations, and the CNN determines whether the distributions of the new features from the crowd annotations and the expert annotations are consistent. If so, the new features can be passed as one of the additional features to the model shown in Figure 2(b); otherwise, the new features are passed back to the BiLSTM layer. Figure 2(b) shows the BiLSTM-Attention-CRF submodel for NER in the crowdsourced domain dataset. Several features, such as the dictionary, sentence dependency, and common features of the crowdsourced annotations are input to the model to improve the quality of the crowdsourced annotations.

4.1. Adversarial Learning for Common Features. The structure of the GAN contains two models: a generative network and discriminative model. The goal is to learn a generative distribution that matches the real data distribution. More

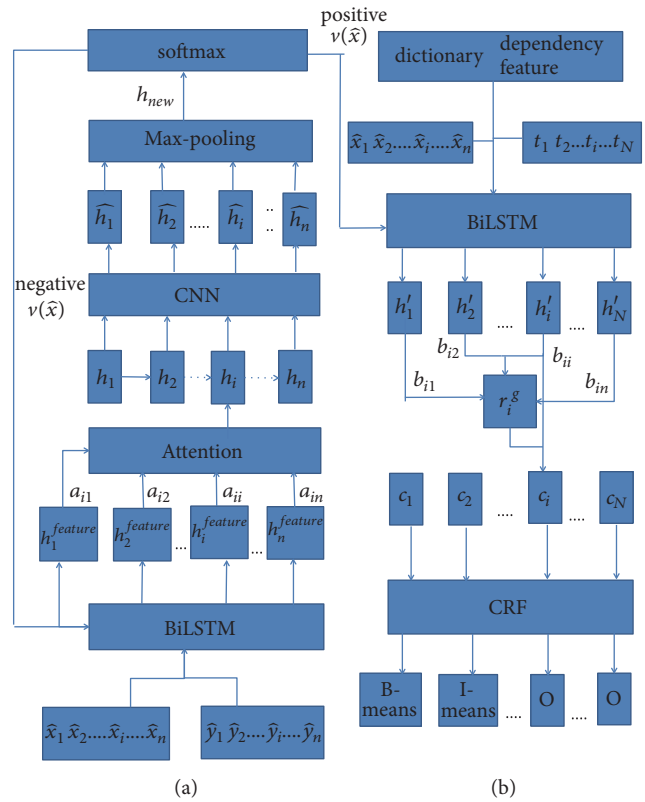

Figure 2: Diagram of the proposed model.

specifically, the generative network generates samples from the generator distribution, and the discriminative model learns to determine whether a sample is from a generative distribution or a real data distribution.

Adversarial learning is used to find the common features of the crowd annotations. The discriminative model is trained using expert annotations and crowd annotations, which are used as the input for the generated model, and the generated feature distribution is passed to the discriminative model and used to determine the similarities and differences in the feature distributions of the crowd and expert annotations. The model is repeatedly trained until the discriminator can no longer distinguish a difference between them. At this point, the result of the model is the set of common features of the crowd annotations, which is also the set of optimal single-consensus annotations that we want to integrate. As shown in Figure 2(a), the model consists of

two parts: the generation model composed of BiLSTM-Attention and the discriminant model composed of the CNN. The max-pooling layer and the softmax layer in the CNN optimise the feature map generated by the CNN layer and normalise the selected new features to determine whether they are consistent with the expert annotations feature distribution.

4.1.1. BiLSTM-Attention Model. We consider a sentence $s = \{w_1, w_2, \ldots, w_n\}$, where n is the number of the words in the sentence. Given the crowd-annotated NE label sequence $y = \{y_1, y_2, \ldots, y_n\}$, where y_i represents the corresponding annotation of the named entity w_i, we use $\widehat{x_i}$ and $\widehat{y_i}$ as the word vectors and the annotation vectors of word w_i, which are trained using word2vec, and pass them as input to the generative model BiLSTM to get the features $h_i^{feature}$, where

$$h_i^{feature} = BiLSTM\left(\widehat{x}_i, \widehat{y}_i\right) \tag{3}$$

In order to obtain more important features, a new attention layer is used above the BiLSTM layer to capture a new representation of the feature h_i:

$$\begin{aligned} f\left(h_i, h_j\right) &= \left(h_i^{feature}\right)^T w_a h_j^{feature} \\ a_{ij} &= soft\max\left(f\left(h_{i-1}, h_j^{feature}\right)\right) \\ h_i &= \sum_j^n a_{ij} h_j^{feature} \end{aligned} \tag{4}$$

where w_a is the model parameter.

4.1.2. CNN. Following this, we add a CNN module based on the outputs of the BiLSTM-Attention model, to determine the similarities and differences between the feature distributions for the crowd and expert annotations. A convolutional operator with a window size of five is used, and then a max-pooling strategy is applied to the convolution sequence to obtain the final fixed-dimensional feature vector. The overall process can be described by the following equations:

$$\widehat{h_i} = \tanh\left(W^c\left[h_{i-2}, h_{i-1}, \ldots h_{i+2}\right]\right) \tag{5}$$

$$h_{new} = \max\left\{\widehat{h_1}, \widehat{h_2} \ldots \widehat{h_n}\right\} \tag{6}$$

where W^c is the CNN model parameters, and the activation function tanh is used primarily to normalise and prevent the loss of features. In the pooling method, the maximum feature is the most important, as it effectively filters out word combinations with less information and can ensure that the extracted features are independent of the length of the input sentence.

The feature h_{new} is then mapped to the output $\mathrm{D}(h_{new}) \in [0, 1]$ using a softmax function to determine whether the input feature is consistent with the feature distribution of the expert annotations.

4.1.3. Objective Function. We use the feature matching method in [21], set S as the expert annotations and use iterative optimisation schemes consisting of two steps:

$$\begin{aligned} &minimizing: L_D \\ &= -E_{s:S}\log\left(D\left(s\right)\right) - E_{\widehat{x}:p(\widehat{x})}\left[\log 1 - D\left(G\left(\widehat{x}\right)\right)\right] \\ &minimizing: L_G \\ &= \mathrm{tr}\left(\sum_s^{-1}\sum_{\widehat{x}} + \sum_{\widehat{x}}^{-1}\sum_s\right) \\ &\quad + \left(\mu_s - \mu_{\widehat{x}}\right)^T\left(\sum_s^{-1} + \sum_{\widehat{x}}^{-1}\right)\left(\mu_s - \mu_{\widehat{x}}\right) \end{aligned} \tag{7}$$

where $\sum_{\widehat{x}}$ and $\sum_s$ represent the covariance matrices for the expert and the crowd annotation features, respectively, μ_s, $\mu_{\widehat{x}}$ denote the mean features of the expert and the crowd annotation features, respectively, and their values are empirically estimated using mini-batch. L_G represents the Jensen-Shannon divergence between two multivariate Gaussian distributions $dtri(\mu_s, \sum_s)$ and $dtri\ (\mu_{\widehat{x}}, \sum_{\widehat{x}})$. The main purpose of this is to provide a stronger signal for modifying the generation model in order to make the feature distribution generated by the generated model more similar to that of the discriminant model [21].

In training the generation model, which contains discrete variables, the direct application of gradient estimation would fail. Thus, we draw on the method used in [21], and use a soft-argmax function when performing the inference as an approximation to the inputs of the generated model BiLSTM:

$$v\left(\widehat{x}\right) = W_e\, soft\max\left(Vh_{new} \circ L\right) \tag{8}$$

where $\circ$ represents the element-wise product, V is a weight matrix used to calculate the word distribution, and W_e is model parameter. When $L \longrightarrow \infty$, this expression approximates the default input vector calculation formula for BiLSTM.

4.2. BiLSTM-Attention-CRF SubModel. The BiLSTM-Attention-CRF submodel adds the attention mechanism to the classical BiLSTM-CRF model to allow it to pay attention to the correlation between the current entity and the other words in the sentence and to obtain the feature representation of words at the sentence level to improve the accuracy of the model labelling. The model structure is shown in Figure 2(b).

Using the word feature $\widehat{x_i}$, the character feature t_i, and additional features corresponding to the words w_i as the input to BiLSTM, we get the new representation h_i' of the word w_i (here, the method of calculating h_i' is the same as for the $h_i^{feature}$ above), which is used as input to the attention layer. The attention weight value b_{ij} in the attention matrix is derived by comparing the current word w_i with the other words $w_j\ (j = 1, 2, \ldots i-1, i+1, \ldots n)$ in the sentence.

$$b_{ij} = \frac{\exp\left(f\left(w_i, w_j\right)\right)}{\sum_{k=1}^{n} \exp\left(f\left(w_i, w_k\right)\right)} \tag{9}$$

The method of calculation of $f(w_i, w_j)$ is shown in (2).

A sentence-level vector r_i^g is then computed as a weighted sum of each BiLSTM output h_i':

$$r_i^g = \sum_{j=1}^{n} b_{ij} h_i' \tag{10}$$

Next, we combine the sentence-level vector r_i^g with the BiLSTM output of the target word as a vector $[r_i^g, h_i']$ to be passed to the tanh function to produce the output of the attention layer.

$$c_i = \tanh\left(W_g\left[r_i^g, h_i'\right]\right) \tag{11}$$

Finally, we use c_i as the input of the upper CRF layer. Here, the CRF has two roles: the first is to calculate the score o_i for each c_i in the corresponding annotation, and the second is to use the tagging transition matrix T (to define the score of two consecutive annotations) and the Viterbi algorithm to calculate the best annotation sequence. This process is expressed as follows:

$$o_i = Wc_i \tag{12}$$

$$score\left(D, y\right) = \sum_{i=1}^{N} \left(o_{i,y_i} + T_{y_{i-1}, y_i}\right) \tag{13}$$

$$y^{result} = \text{argmax}\ \left(score\left(D, y\right)\right) \tag{14}$$

where the function $score(\,)$ is used to calculate the score of the annotation sequence $y = y_1 y_2 \ldots y_n$ of the input sentence, y^{result} is the final output annotation sequence result (i.e., BIO annotation), and W represents the model parameter.

5. Experimental Results and Analysis

To evaluate our model, we divided the baseline models into two groups based on their different uses. First, our model and the first group of baseline models were applied to the crowdsourcing annotations to verify the ability of our model to integrate consensus annotations in comparison with other baseline models. Secondly, our model and the second group of baseline models were applied to identify specific entities in information security to verify the ability of our model to identify specific types of entities. Finally we verified the effect of additional features on the performance of the model.

5.1. Data Sources. The dataset used in this experiment was mainly drawn from the field of information security, and included related blog posts (such as we live security, threatpost), CVE descriptions, Microsoft security bulletins, and information security abstracts. From this corpus, 10,187 sentences were selected (consecutive paragraphs including 20 abstracts, 45 blog posts, 59 CVE descriptions and 50 Microsoft security bulletins) and each sentence was assigned to three annotators to generate crowd annotations. These three annotators were students at the authors' institution with no educational background in information security. Each annotator only needed to annotate four types of named entities in the sentence: the product, the consequence, the attacker, and the version. Two senior students taking information security courses were asked to annotate 1,000 sentences that were randomly selected to train the discriminant model in the GAN. From the crowd annotations, we randomly chose 7,000 sentences as a training set and used the remainder as a test set.

5.2. Baseline Models. The comparison models were divided into two groups for experiments.

Group 1: to learn the common features of crowd annotations, we used the following as comparison models:

(I) Majority vote (MV)[17]

(II) Dawid and Skene Model [25].

Group 2: to predict the named entity sequence from unlabeled text, we used the following as comparison models:

(I) BiLSTM-Attention-CRF: The model in [35] was trained directly using the crowd annotations. When we used this model, we removed the part of the model that used image features.

(II) BiLSTM-Attention-CRF-VT: This was trained on the data selected from the crowd annotations by majority vote.

(III) HMM-crowd [28].

(IV) CRF-MA: From the model in [26], we used the source code provided by the author.

5.3. Settings. There are several hyperparameters in our model. We set the dimensions of the futures vector to 300, the number of units in BiLSTM to 1000, and the minibatch size to 64. The max-epoch iteration was set to 100. The method described in [36] with a learning rate of 10^{-3} was used to update the model parameters, and the l_2 regularisation was set to 10^{-5}. We adopted the dropout technique to avoid overfitting. The dropout was 0.3 for BiLSTM and 0.5 for the attention layer.

Our experiment was implemented on two NVIDIA GTX 1080Ti GPU with 64 GB memory, and the model was trained for approximately one hour.

5.4. Evaluation of Experimental Results. The indicators used in the evaluation of the experiment were the accuracy rate (P), the recall rate (R), and the F1 value.

5.4.1. Performance Comparison of Integrated Crowd Annotation Model

(I) Performance Comparison between Our Model and the First Group of Baseline Models. We use the accuracy rate of the correct annotation obtained from the training corpus used by each model as our evaluation criterion.

Table 1: Performance comparison for the models used.

Method	Accuracy rate
MV	65.3%
Dawid and Skene	72.5%
BiLSTM-Attention-CRF-crowd	78.9%

Table 2: Comprehensive performance evaluation for each model.

Method	Precision	Recall	F1
BiLSTM-Attention-CRF	88.5%	79.9%	84.4%
CRF-MA	88.4%	75.6%	81.5%
Dawid and Skene-LSTM	79.3%	75.1%	77.1%
BiLSTM-Attention-CRF-VT	75.5%	75.6%	75.6%
BiLSTM-Attention-CRF-crowd	89.1%	90.0%	89.0%

Performance comparisons for various models on the test corpus are shown in Table 1.

The performance of MV was relatively poor. This is because it is difficult to achieve uniformity for an ambiguous entity due to the professionalism of the field and the uneven distribution of the annotation level. Our model achieves the best performance. In addition to obtaining the correct annotations in the training corpus, our model also can generate a positive sample that is consistent with the feature distribution of the expert annotations through repeat training. Through the secondary extraction of the BiLSTM-Attention-CRF submodel, the accuracy rate of correct annotations in the training corpus is improved.

(II) Comparison of the Overall Performance of NER in the Information Security Field for Each Model. Table 2 shows that in terms of precision, the BiLSTM-Attention-CRF model that was directly trained using unprocessed crowd annotations had the highest precision. This means that the crowd annotations are useful for training the NER model. The BiLSTM-Attention-CRF-VT model trained on data selected using the MV method from the crowd annotations showed the poorest performance. This model may therefore not be suitable for the information security field. The reason for this may be the complexity and professionalism of the information security text statements. Many entities cannot be selected by voting. In addition, contextual information is lost by voting selection, which means important feature information is lost. The BiLSTM-Attention-CRF model, which directly uses unprocessed crowdsourcing label data as training data, does not lose important context information; however, the level of noise is increased, so its overall performance is slightly lower than that of the BiLSTM- Attention-CRF-crowd model.

5.4.2. Performance of Specific Type Entity Recognition

(I) Integrated Performance Evaluation of Specific Types of Entity Annotation. In order to verify the integration performance of our model in terms of recognising different types of entities, the integration results of the four types of entities proposed above in the crowd annotations were compared. In Figure 3, the ordinates represent the P, R, and F1 values, respectively. The labels 1, 2, 3, and 4 on the abscissa correspond to the entity type (product, attacker, consequence, and version, respectively), and 5 represents the mean of the three indicators of the model. As can be seen from Figure 3, the model performs better on Types 1 and 4, mainly because the class indicated by Type 1 is the product. In general, although this type of entity belongs to the information security domain, public awareness of this field is relatively high, meaning that the precision of the annotation is relatively high. The category corresponding to Type 4 has a relatively fixed pattern; it always appears after the product name and is usually expressed by numbers. The performance of the model is best for entities with a fixed patterns, because its features are easier to learn. For Types 2 and 3, which are relatively specific types of entity in the information security field, the precision of the crowd annotations was low. In particular, the consequence class of Type 3 has higher requirements for the professional ability of the annotator and the entity part of the category is not fixed, so the performance is slightly weaker.

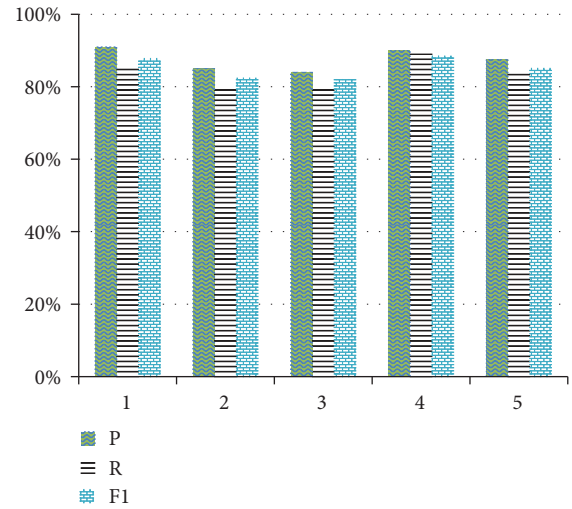

Figure 3: Performance comparison for different types of entity.

(II) Performance Evaluation of Specific Types of Entity with Other Models. In Figure 4, the ordinate represents the accuracy of each model, and the abscissa is the same as in Figure 3. These models take the basic, domain dictionary and sentence-dependency features as additional features for input. It can be seen from the figure that the performance of the BiLSTM-Attention-CRF-crowd model is better than that of the other models. However, the model also has a lower accuracy than the recognition of each type in Figure 3, which also proves that the common feature of adversarial learning has a significant effect on improving the accuracy of the model.

(III) Effect of Additional Features on the Performance of Each Model. The above four entity types are used as extraction targets, and we compare our model with itself which uses

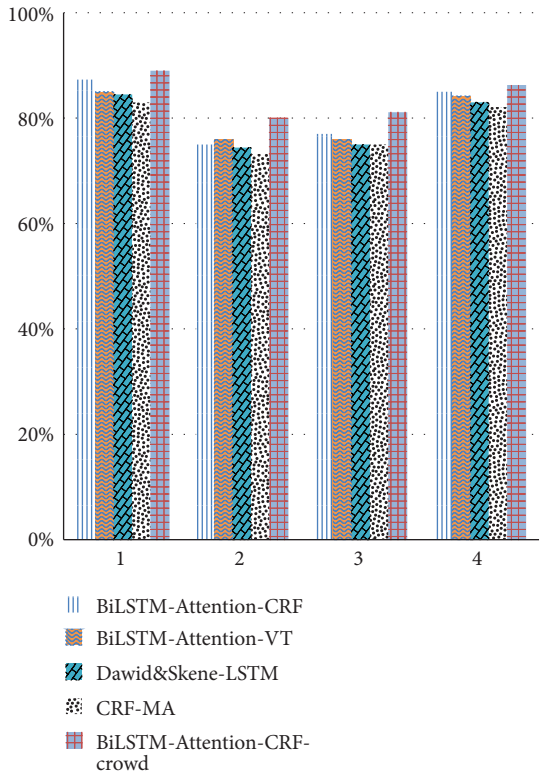

Figure 4: Performance comparison of models.

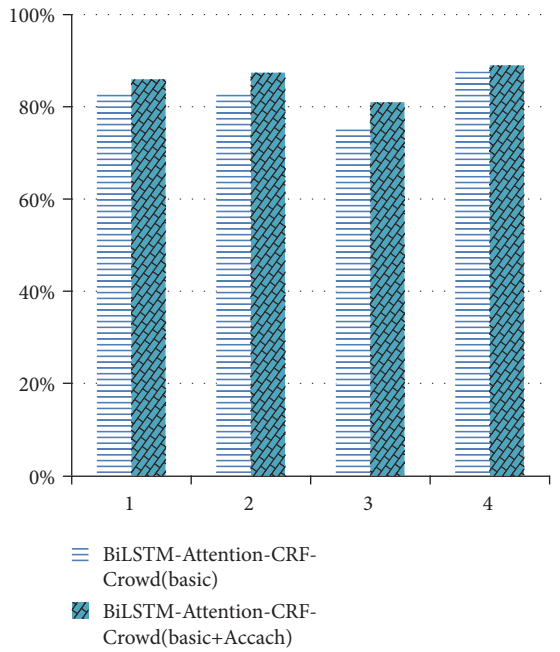

Figure 5: Effect of additional features on model performance.

word features and character features as inputs (i.e., BiLSTM-Attention-CRF-crowd (basic)) and another using basic features and additional features as inputs (BiLSTM-Attention-CRF-crowd (basic+attach)). The results for the extraction accuracy are shown in Figure 5.

It can be seen that the extraction accuracy of the BiLSTM-Attention-CRF-crowd (basic+attach) model is significantly higher than that of the BiLSTM-Attention-CRF-crowd (basic) model, which proves that the additional features have a measurable effect on the accuracy of entity extraction, and, especially on the performance of extracting the consequence class, the practical value of the sentence dependency feature for extracting the entity type of nonnoun or nonnoun phrases verified.

Combined with the above experiments, the performance of the BilSTM-Attention-CRF-Crowd model is excellent and is superior to the other models studied in this paper.

6. Conclusion

In this paper, we have proposed a new model BiLSTM-Attention-CRF-crowd to improve the quality of crowdsourcing annotations in information security field. The main work includes the following: (1) the common features of crowd annotations are found by the GAN model to generate the best unique consensus annotation; (2) these common features, domain dictionaries, and sentence dependencies are used as additional features to identify the entities of crowdsourcing annotations again, so as to improve the quality of crowdsourcing annotations. We evaluate our model on data sets in the field of information security, and the results show that its performance is better than the other baseline models mentioned in this paper. It is also verified that the proposed domain dictionary features and sentence dependency features have practical value for improving the performance of the model. However, the increase of input features will inevitably lead to an increase in the time complexity of the model. In future, we will consider further improvements to the model.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61501515) and the Project of Henan Provincial Key Scientific and Technology (Grant no. 172102210002).

References

[1] C. Jochim and L. A. Deleris, "Named entity recognition in the medical domain with constrained CRF models," in *Proceedings of the 15th Conference of the European Chapter of the Association*

for Computational Linguistics, pp. 839–849, Computational Linguistics, Spain, April 2017.

[2] S. Liu, B. Tang, Q. Chen, and X. Wang, "Drug name recognition: approaches and resources," *Information*, vol. 6, no. 4, pp. 790–810, 2015.

[3] J. Liang, X. Xian, X. He, M. Xu, S. Dai, J. Xin et al., "A novel approach towards medical entity recognition in chinese clinical text," *Journal of Healthcare Engineering*, vol. 2017, Article ID 4898963, 16 pages, 2017.

[4] J. Lei, B. Tang, X. Lu, K. Gao, M. Jiang, and H. Xu, "A comprehensive study of named entity recognition in chinese clinical text," *Journal of the American Medical Informatics Association*, vol. 21, no. 5, pp. 808–814, 2014.

[5] Z. Liu, M. Yang, and X. Wang, "Entity recognition from clinical texts via recurrent neural network," *BMC Medical Informatics & Decision Making*, vol. 17, no. 2, 2017.

[6] S. Ramamoorthy and S. Murugan, "An attentive sequence model for adverse drug event extraction from biomedical text," https://arxiv.org/abs/1801.00625.

[7] R. Leaman and C. H. Wei, "A high performance approach for chemical named entity recognition and normalization," *Journal of Cheminformatics*, vol. 7, no. 1, pp. S1–S3, 2015.

[8] L. Luo and Z. Yang, "An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition," *Bioinformatics*, vol. 34, no. 8, pp. 1381–1388, 2018.

[9] T. H. Dang, H.-Q. Le, T. M. Nguyen, and S. T. Vu, "D3NER: biomedical named entity recognition using CRF-biLSTM improved with fine-tuned embeddings of various linguistic information," *Bioinformatics*, vol. 34, no. 20, pp. 3539–3546, 2018.

[10] I. Korvigo, M. Holmatov, A. Zaikovskii, and M. Skoblov, "Putting hands to rest: efficient deep CNN-RNN architecture for chemical named entity recognition with no hand-crafted rules," *Journal of Cheminformatics*, vol. 10, no. 1, 2018.

[11] R. A. Bridges, C. L. Jones, M. D. Iannacone, K. M. Huffer, and J. R. Goodall, "Automatic labeling for entity extraction in cyber security," https://arxiv.org/abs/1308.4941.

[12] S. Weerawardhana, S. Mukherjee, I. Ray, and A. Howe, "Automated extraction of vulnerabilit-y information for home computer security foundations and practice of security," in *Proceedings of the springer International Publishing*, vol. 8930, pp. 356–366.

[13] R. Lal, "Information extraction of cybersecurity related terms and concepts from unstructured text," in *Proceedings of the 2013 IEEE Seventh International Conference on Semantic Computing (ICSC)*, University of Maryland, California, Calif, USA, September 2013.

[14] V. Mulwad, W. Li, and A. Joshi, "Extracting information about security vulnerabilities from Web text," in *Proceedings of the 2011 IEEE/WIC/ACM International Conference On Web Intelligence and Intelligent Agent Technology*, pp. 257–260, IEEE Computer Society, August 2011.

[15] Amazon Mechanical Turk., https://www.mturk.com/.

[16] R. Snow and B. O'Connor, "Cheap and fast – but is it good? Evaluating non-expert annotations for natural language tasks," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 254–263, ACL, USA, October 2008.

[17] R. S. Boyer and J. S. Moore, "MJRTY—a fast majority vote algorithm," in *Automated Reasoning Series*, vol. 1 of *Automated Reasoning Series*, pp. 105–117, Springer Netherlands, Dordrecht, 1991.

[18] I. J. Goodfellow, J. Pouget-Abadie, and M. Mirza, "Generative adversarial nets," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pp. 2672–2680, Quebec, Canada, 2014.

[19] Y. S. Yang, M. Zhang, and W. Chen, "Adversarial learning for chinese NER from crowd annotations," https://arxiv.org/abs/1801.05147.

[20] Z. Syed, A. Padia, and T. Finin, "UCO: a unified cybersecurity ontology," in *Proceedings of the twenty-sixth AAAI*, 2016.

[21] Y. Z. Zhang, Z. Gan, and L. Carin, "Generating text via adversarial training," in *Proceedings of the 2016 Conference of Neural Information Processing Systems Workshop on Adversarial Training*, 2016.

[22] L. Yu, W. Zhang, and J. Wang, "SeqGAN: Sequence generative adversarial nets with policy gradient," https://arxiv.org/abs/1609.05473.

[23] J. Li and W. Monroe, "Adversarial learning for neural dialogue generation," https://arxiv.org/abs/1701.06547.

[24] M. J. Kusner and J. M. Hernandeslobato, "GANS for sequences of discrete elements with the gumbel-softmax distribution," https://arxiv.org/abs/1611.04051.

[25] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer Error-Rates using the EM algorithm," *Journal of Applied Statistics*, vol. 28, no. 1, pp. 20–28, 1979.

[26] M. Dredze, P. Talukdar, and K. Crammer, "Sequence learning from data with multiple labels," *Rule Based*, pp. 39–48, 2010.

[27] H. Kajino, Y. Tsuboi, H. Kashima et al., "A convex formulation for learning from crowds," in *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pp. 73–79, 2012.

[28] A. T. Nguyen, B. C. Wallace, and J. J. Li, "Aggregating and predicting sequence labels from crowd annotations," *Meeting of the Association for Computational Linguistics*, pp. 299–309, 2017.

[29] R. Agerri, Y. Chung, I. Aldabe, N. Aranberri, G. Labaka, and G. Rigau, "Building named entity recognition taggers via parallel corpora," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018*, 2018.

[30] J. M. Giorgi and G. D. Bader, "Transfer learning for biomedical named entity recognition with neural networks," Bioinformatics, 2018.

[31] S. Lai, K. Liu, L. Xu, and J. Zhao, "How to generate a good word emdedding," *IEEE Intelligent Systems*, vol. 31, pp. 5–14, 2016.

[32] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representation of words and phrases compositionality," *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.

[33] "CVE homepag," https://cve.mitre.org/.

[34] "Corenlp homepage," http://corenlp.stanford.edu.

[35] Q. Zhang, J. Fu, X. Liu, and X. Huang, "Adaptive co-attention network for named entity recognition in tweets," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[36] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," https://arxiv.org/abs/1412.6980.