TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 02/12 pp259–265
DOI: 10.26599/TST.2019.9010033
Volume 26, Number 3, June 2021

Cybersecurity Named Entity Recognition Using Bidirectional Long Short-Term Memory with Conditional Random Fields

Pingchuan Ma, Bo Jiang*, Zhigang Lu, Ning Li, and Zhengwei Jiang

Abstract: Network texts have become important carriers of cybersecurity information on the Internet. These texts include the latest security events such as vulnerability exploitations, attack discoveries, advanced persistent threats, and so on. Extracting cybersecurity entities from these unstructured texts is a critical and fundamental task in many cybersecurity applications. However, most Named Entity Recognition (NER) models are suitable only for general fields, and there has been little research focusing on cybersecurity entity extraction in the security domain. To this end, in this paper, we propose a novel cybersecurity entity identification model based on Bidirectional Long Short-Term Memory with Conditional Random Fields (Bi-LSTM with CRF) to extract security-related concepts and entities from unstructured text. This model, which we have named XBiLSTM-CRF, consists of a word-embedding layer, a bidirectional LSTM layer, and a CRF layer, and concatenates X input with bidirectional LSTM output. Via extensive experiments on an open-source dataset containing an office security bulletin, security blogs, and the Common Vulnerabilities and Exposures list, we demonstrate that XBiLSTM-CRF achieves better cybersecurity entity extraction than state-of-the-art models.

Key words: security blogs; Long Short-Term Memory (LSTM); Named Entity Recognition (NER)

1 Introduction

Much attention is focused on cybersecurity incidents, and there are more and more reports of security incidents on the Internet. For example, some companies post the security vulnerabilities of their software on their official websites, and some security researchers choose to use social networking sites to post security information[1]. Alternatively, they may post articles on their personal blogs or blogs in the security community to highlight security threats. This means there is a large amount of security information on the network, but it is inefficient for researchers to manually identify these information sources to extract useful information. Therefore, it is helpful for researchers to be able to automatically discover and extract security information on the network.

Some researchers have studied information extraction in the security domain. Mittal et al.[1] analyzed tweets related to cybersecurity and issued timely threat alerts to security analysts. Weerawardhana et al.[2] proposed a model that extracts information from online vulnerability databases, such as the Common Vulnerabilities and Exposures (CVE) list and the National Vulnerability Database. However, these resources focus on data from just one source. Twitter has a limit of 140 words, so it contains only a small amount of information. Online vulnerability databases usually contain structured data. More importantly, security researchers often choose to send security information in article form to a secure community or personal blog. Therefore, it is very important to find cybersecurity information from natural language sources. To address this issue, we decided to start with

• Pingchuan Ma, Zhigang Lu, and Zhengwei Jiang are with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: {mapingchuan, luzhigang, jiangzhengwei}@iie.ac.cn.
• Ning Li and Bo Jiang are with Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China. E-mail: {lining6, jiangbo}@iie.ac.cn.
* To whom correspondence should be addressed.
Manuscript received: 2019-07-19; accepted: 2019-07-26

Named Entity Recognition (NER) in the cybersecurity domain.

NER[3], a classic problem in the field of natural language processing, refers to the identification of entities with specific meaning in the text, including person names, place names, institution names, proper nouns, and so on. In essence, it is a sequence labeling task that produces an output sequence from a given input sequence. With the development of machine learning algorithms and the improvements in computing power, research on NER in the general domain has made great progress.

In the field of cybersecurity, there has been scant NER research. There are two main reasons for this. First, there is very little annotation data available, so it is difficult to classify entity categories in the security domain. Second, compared to other domains, unstructured text in the security domain contains more special vocabulary, such as file names, hash values, and even code snippets which can cause great confusion to researchers. A successful NER system can greatly facilitate the work of security analysts. In this paper, we propose a model for identifying entities, named XBiLSTM-CRF, which is based on Bidirectional Long Short-Term Memory with Conditional Random Fields (Bi-LSTM with CRF)[4]. To evaluate our model, we used open-source data, but when manually checking these annotation data, we found some mislabeled and missing annotations. Therefore, part of our work involved re-labeling these data. Overall, we identified six entity categories, including *software*, *network*, *attack*, *filename*, *hardware*, and *modifier*, which we describe in detail below. In summary, the contributions of this work are as follows:

- We leverage a neural network model, XBiLSTM-CRF, to perform NER in the cybersecurity domain. This model is based on Bi-LSTM with CRF. Compared to the classic model, it concatenates word embedding input with Bi-LSTM output. The performance of our model is greatly improved compared with those of other models, without increasing the complexity.
- To evaluate our model, we used an open-source NER dataset in the security domain. This domain contains six categories that are meaningful for security analysis, and the dataset size is sufficient for NER. Due to the presence of incorrect and missing annotations in the dataset, we manually relabeled the dataset.
- To verify the effectiveness of our model, we evaluated it on an open-source dataset. Compared to other models, the experimental results show that our model achieved the best overall precision, recall, and F1 score.

The rest of this paper is organized as follows. We present related works in Section 2, then introduce our model in Section 3. We present our experiments and evaluations in Section 4. In Section 5, we draw our conclusion and suggest future work.

2 Related Work

To date, much work has been done in the domain of information extraction and NER has been applied in various domains. In this section, we explain our motivations and explore some possible approaches to apply NER in other domains.

Liao et al.[5] proposed an innovation solution for fully automated Indicators Of Compromise (IOC)[6] extraction, called iACE. An IOC is an artifact observed on a network or in an operating system that indicates with high confidence a computer intrusion. However, IOCs are not very intuitive and offer no help in better understanding these intrusions. The security entities and their relations in the text are more helpful in grasping the nature of cybersecurity threats. This is our primary motivation for doing this work.

Lal[7] provided a dataset that contains several categories of cybersecurity entities, and used the Stanford NER to solve this problem. Recently, however, Bi-LSTM with CRF[4] has been identified as the best model to perform NER in the general domain. It demonstrates the best performance in the CoNLL-2003 dataset. We made some improvements to LSTM with CRF that make it suitable for the security domain and applied it to NER tasks.

More and more people are applying NER to specific areas to extract entities from unstructured text. Luo et al.[8] used a CRF model with a variety of traditional and novel features to find gene and protein mentions in biomedical abstracts. Ritter et al.[9] proposed a model to find persons, companies, movies, and other entities mentioned in tweets. Minkov et al.[10] used NER to identify personal names in emails. In the cybersecurity domain, however, researchers have always focused on vulnerability mining and analysis, which means the security intelligence contained in security reports has been ignored. Cyber threat intelligence is an important concept that includes emerging security threats. These threats can greatly facilitate companies and organizations in addressing security threats. More

et al.[11] used knowledge from the Internet to perform intrusion detection, but found the efficiency of using unstructured text to be unsatisfactory. Therefore, NER research in cybersecurity is very necessary.

3 Proposed Model

In this section, we describe XBiLSTM-CRF in detail.

XBiLSTM-CRF is a bidirectional LSTM model with CRF that concatenates X input with bidirectional LSTM output. As shown in Fig. 1, our model has three components, namely a word-embedding layer, a bidirectional LSTM layer, and a CRF layer. In this study, our input is unstructured text, which is natural language. The word-embedding layer converts each word into a vector. The bidirectional LSTM layer generates two hidden states after forward and backward propagations. The Bi-LSTM output is concatenated with a word-embedding vector as input to the CRF layer. The CRF layer then produces a sequence tag with the highest probability as its output. Next, we describe in detail how these three parts work.

3.1 Word embedding layer

As we know, to make a word calculable, we must convert it into a vector. There are several ways to do so, including one-hot encoding, using a skip-gram model, or a continuous bag of words model, and so on[12]. However these are general methods, and the generated word-embedding vector does not focus on the specific task. Due to the particularity of the security domain, some words may have different meanings than usual. So we added the generation of word vectors to the model training, and trained the model and the word vector at the same time.

As shown in Fig. 2, on the left is the one-hot encoding of the vocabulary, which is expressed as W_O. On the right is the word-embedding vector, which is expressed as W_E. The parameter is θ.

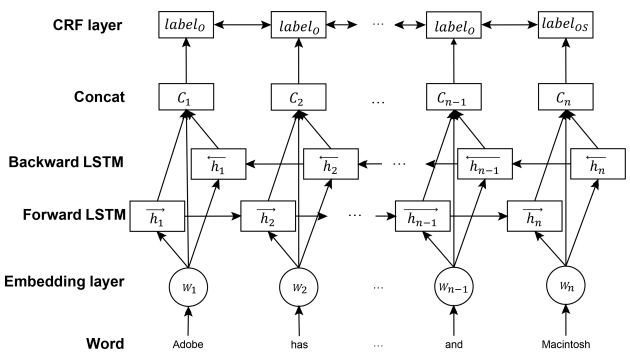

Fig. 1 Architecture of the proposed XBiLSTM-CRF. The bottom is the word-embedding layer, the middle is the bidirectional LSTM layer, and the top is the CRF layer.

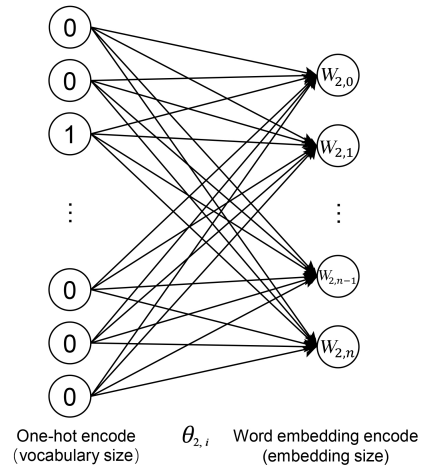

Fig. 2 Word-embedding layer. From one-hot encoding to word-embedding encoding.

$$W_E = W_O \times \theta \tag{1}$$

As a result, we can obtain the word vector for the security-domain task.

3.2 Bidirectional LSTM layer

LSTM can solve timing-related classification problems. Considering the dependency of the text, a word may be related to its previous and next words. In this case, bidirectional LSTM is the best choice. Our model can be described by the following formulas. With the forward LSTM layer, the input and output vectors can be written as follows:

$$O_t^f = H_t^f = f_{\text{forward}}(W_t, H_{t-1}^f) \tag{2}$$

where W_t is the input of time t, H_{t-1} is the hidden state of time $t-1$, O_t is the output of time t, and H_t is the hidden state of time t. For the backward LSTM layer,

$$O_t^b = H_t^b = f_{\text{backward}}(W_t, H_{t+1}^b) \tag{3}$$

The total output at time t is as follows:

$$C_t = \text{concat}(O_t^f, O_t^b, W_t) \tag{4}$$

Because there are too many proprietary vocabularies in the security domain, we concatenate the input vector at time t with the total output. By doing so, we can predict each vocabulary label more accurately.

Considering the length of sentences, we use a dynamic Recurrent Neural Network (RNN) for decoding, so its performance is not dependent on the sentence length. No matter how long the input sentence is, we can generate a suitable prediction.

3.3 CRF layer

Generally speaking, when we have the output of

a bidirectional LSTM, we can simply predict the label using a fully connected layer classification. However, this prediction method does not consider the relationships between labels. So instead, we use CRF after the LSTM. CRF has a transfer matrix that represents the probability that one label will be transferred to another in a sequence. The transfer matrix is also trained. By using CRF layer, the results are made more accurate and meaningful.

4 Experiment

In this section, we present the experiments we conducted to evaluate our model. Using the Stanford NER[13] as our baseline, we conducted an experiment to determine the setting of the hyperparameters. As usual, we used precision (P), recall (R), and F1-score (F1) to evaluate our model.

4.1 Dataset

We used an open-source security-domain unstructured text dataset in our experiment[7]. This dataset consists of manually extracted data from the cybersecurity domain, including articles from the CVE, Adobe Security Bulletins, Microsoft Security Bulletins, and various blog posts. In total, the data comprised more than 45 000 tokens and nearly 5000 tagged entities. This data had following classes and subclasses:

- SOFTWARE (number: 1811, e.g., Microsoft.Net Framework 3.5)
  - OPERATING SYSTEM (number: 996, e.g., Linux Ubuntu 10.4)
- NETWORK (number: 100, e.g., Whatis, IP Address, and HTTP)
- ATTACK (number: 83, When you can not decide between means and consequence)
  - MEANS (number: 479, Immediate system reaction to input (e.g., buffer overflow))
  - CONSEQUENCES (number: 481, Final result of an attack (e.g., denial of service))
- FILE_NAME (number: 71, e.g., index.php)
- HARDWARE (number: 21, e.g., IBM Mainframe B152)
- MODIFIER (number: 781, e.g., Acrobat Reader X and earlier versions. Here, "and earlier versions" is the MODIFIER.)

Our task was to identify these types of entities contained in the unstructured text. In the process of manually checking the data, we found some label errors and missing labels, as follows: Adobe has released security updates for Adobe Flash Player 11.5.502.146 and earlier versions for Windows and Macintosh. Here, "and earlier versions" is entities of the *modifier*, but the label is "O". To ensure that our algorithm can generate better results, we relabeled the data.

4.2 Baseline methods

In this subsection, we introduce three well-known models we used for comparison with our proposed model.

- **Stanford NER**: Stanford NER is a Java implementation of NER that uses a CRF classifier[7]. This is a good NER for a variety of features, particularly for the nine features provided by Stanford NER such as UseNGrams, UsePrev, and UseWordPairs. In this paper, based on our relabeled dataset, we can retrain the Stanford NER model to perform cybersecurity NER.
- **LSTM with Dense Layer**: LSTM[14] is a simple sequential model, with the output of each moment related to the input of all previous moments. But the earlier the input was, the less weight it has. Classification can be achieved by adding a fully connected layer after the output of each moment.
- **LSTM with CRF**: This is the classic NER model. By adding a CRF layer after the LSTM, the model performance is greatly improved. The CRF layer does not extract features, and only chooses the sequence with the highest probability. It is widely used in the identification of named entities in various domains. By comparing our model with this model, we can better explain the effect of our improvement in using concatenated input with bidirectional output.

4.3 Experimental setting

In our proposed model, the embedding layer has 128 dimensions, so we set the units of hidden layers to 128, and the number of hidden layers to 2. To prevent overfitting, we added some dropout layers with a dropout rate of 0.2. We set the learning rate to 0.01, and we compare the effects of different learning rate values later. The two other neural network models were given the same settings. For the Stanford NER model, we used the officially recommended feature templates.

In our experiments, as ordinary operations, we split our dataset into training data and testing data (about 8:2). Because we use the dynamic RNN, sentences of different lengths are placed in different buckets for training. Starting at 20, every time the sentence length exceeds 10, it is placed in the next bucket.

Therefore, we set our batch size to 8. We compare the performances using different batch sizes in a later section.

All the experimental results were obtained by running the considered systems on a computer equipped with an Inter Core i7-8086K CPU, with 128 GB of memory and a Nvidia 1080 Ti, and running a Ubuntu 16.10 OS.

4.4 Evaluation metrics

We evaluated our experimental results using standard information-retrieval metrics, including P, R, and F1. We utilized 5-fold cross validation to identify the best parameters and evaluated the model performances with respect to test accuracy at the instance level.

4.5 Results and analysis

In Fig. 3, we show the cybersecurity NER results of our proposed model and those of the other methods. From the figure, we can see that with respect to P, R, and F1, our model achieved the best results. By concatenating the Bi-LSTM output with the input word-embedding vector, the precision is significantly improved. This shows that our model can recognize special vocabulary in the cyber security field. Compared with the dense layer, the CRF layer improves the recall, because it considers the dependencies between different labels. Table 1 shows the P, R, and F1 values for each category, from which we can see that our model performs best in most cases.

From Table 1, we can see that the P and R values of the *network* and *hardware* categories are very different. This is because the number of entities in these two categories is small. Therefore, we argue that our proposed model is suitable for cybersecurity NER for large-scale collections.

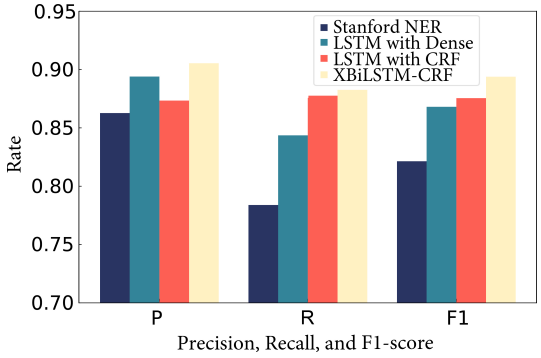

Fig. 3 Performance comparison of our proposed model and the baseline methods on cybersecurity named entity recognition.

Table 1 Performance comparison of our method with baseline models with different entity categories in the constructed dataset.

Category	Method	Precision	Recall	F1-score
SOFTWARE	Stanford NER	0.8968	0.8128	0.8527
	LSTM with Dense	0.9387	0.8703	0.9032
	LSTM with CRF	**0.9488**	0.8999	0.9237
	XBiLSTM-CRF	0.9455	**0.9046**	**0.9245**
MODIFIER	Stanford NER	0.8902	0.8148	0.8508
	LSTM with Dense	0.9120	0.8287	0.8683
	LSTM with CRF	0.9149	0.8816	0.8979
	XBiLSTM-CRF	**0.9163**	**0.8868**	**0.9013**
OPERATING SYSTEM	Stanford NER	**0.8942**	0.9337	**0.9135**
	LSTM with Dense	0.8765	0.9342	0.9044
	LSTM with CRF	0.8831	**0.9441**	0.9125
	XBiLSTM-CRF	0.8871	0.9309	0.9084
CONSE QUENCES	Stanford NER	0.8148	0.8381	0.8263
	LSTM with Dense	0.8723	0.9264	0.8985
	LSTM with CRF	0.8267	**0.9709**	0.8930
	XBiLSTM-CRF	**0.9094**	0.9341	**0.9215**
ATTACK	Stanford NER	0.7273	0.4444	0.5517
	LSTM with Dense	**0.8438**	0.675	**0.7500**
	LSTM with CRF	0.4615	0.45	0.4556
	XBiLSTM-CRF	0.7632	**0.725**	0.7436
MEANS	Stanford NER	0.7174	0.6055	0.6567
	LSTM with Dense	0.762	0.7231	0.7420
	LSTM with CRF	0.6772	0.75	0.7117
	XBiLSTM-CRF	**0.7723**	**0.793**	**0.7825**
FILE_NAME	Stanford NER	**0.7143**	0.5556	**0.625**
	LSTM with Dense	0.4783	0.3929	0.4314
	LSTM with CRF	0.4255	**0.7143**	0.5333
	XBiLSTM-CRF	0.6522	0.5357	0.5882
NETWORK	Stanford NER	**0.6**	0.1111	0.1875
	LSTM with Dense	0.5217	0.2353	0.3243
	LSTM with CRF	0.375	0.1765	0.2400
	XBiLSTM-CRF	**0.6**	**0.3529**	**0.4444**
HARDWARE	Stanford NER	**1.0000**	0.1667	**0.2857**
	LSTM with Dense	0.3333	0.125	0.1818
	LSTM with CRF	0.2857	**0.25**	0.2666
	XBiLSTM-CRF	**1.0000**	0.125	0.2222
Total	Stanford NER	0.8627	0.7839	0.8214
	LSTM with Dense	0.894	0.8436	0.8680
	LSTM with CRF	0.8734	0.8776	0.8754
	XBiLSTM-CRF	**0.9054**	**0.8826**	**0.8938**

4.6 Impact of batch size

We now consider the impact of batch size on our proposed model. As shown in Fig. 4, we find that when the batch size is set to 8, the model achieved the best result, and also has higher precision. Generally speaking, a large batch size can make the model learn faster and perform better overall, and a small batch size makes the learning slower. In this case, although the model can easily perform particularly well on a certain

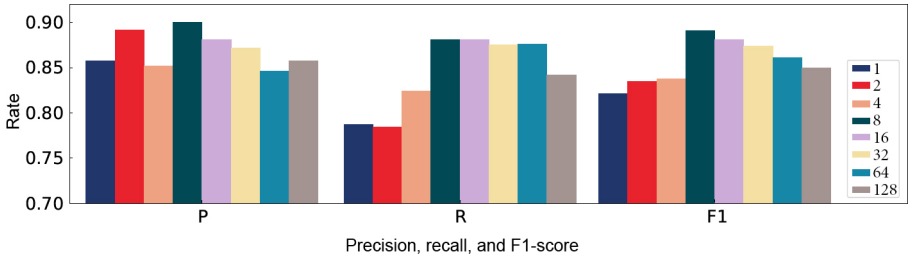

Fig. 4 Comparison of precision, recall, and F1 scores of our proposed model with different batch sizes.

type of sample, the overall performance is poor. Given the above factors and the results of the experiment, we consider the most suitable batch size to be 8.

4.7 Impact of learning rate

The learning rate is also a very important parameter in neural network algorithms. A high learning rate enables the model to quickly approach the optimal solution, but it may oscillate around the optimal solution and never become stabilized at the optimal solution. A low learning rate will slow the loss function of the model, and enable it to stabilize at the optimal solution. Taking into account the above two factors, we decided to add a learning rate decay mechanism, in which a high learning rate is used at the beginning of training and a low learning rate is used after a certain number of iterations. In this way, the optimal solution can be achieved quickly and oscillation around the optimal solution can be avoided.

As shown in Fig. 5, without using the learning rate decay mechanism, the loss function of the model is stable within a small range. But when the learning rate decay mechanism is applied, the model's loss function drops very quickly and eventually settles in a small range.

5 Conclusion and Future Work

In this paper, to perform NER in the cybersecurity domain, we proposed the machine learning model XBiLSTM-CRF, which is based on LSTM with CRF. We used an open-source dataset and relabeled the dataset. The experimental results reveal that XBiLSTM-CRF can identify entities in the security domain with relatively high precision and recall. We use a dynamic model, which enables our model to be independent of sentence length.

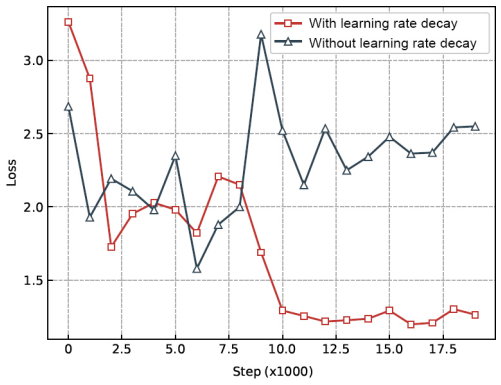

Fig. 5 Comparison of learning rate decay mechanism.

The results of our work will have a very positive effect on the extraction of knowledge in the security domain and the construction of knowledge graphs. Future work can be considered as follows:

- Relational extraction of unstructured text in the security domain. For example, the discovery of attackers and victims described in the text and which attack methods are used. Alternatively, the determination of what kind of attack has occurred against an operating system. This research can greatly reduce the workload of security analysts.
- Mining the entities and relationships described in a security report, constructing a graphic representation, and combining existing knowledge to build a broad knowledge graph. This research can facilitate the discovery of possible connections between different organizations or people. With the help of a security knowledge graph, cybersecurity situations can be better understood.

Acknowledgment

This research was supported by the National Natural Science Foundation of China (Nos. 61702508, 61802404, and U1836209), the National Key Research and Development Program of China (Nos. 2018YFB0803602 and 2016QY06X1204), and the National Social Science Foundation of China (No. 19BSH022). This research was also supported by the Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences, and Beijing Key Laboratory of Network Security and Protection Technology.

References

[1] S. Mittal, P. K. Das, V. Mulwad, A. Joshi, and T. Finin,

CyberTwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities, in *Proc. 2016 IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining*, San Francisco, CA, USA, 2016, pp. 860–867.

[2] S. Weerawardhana, S. Mukherjee, I. Ray, and A. Howe, Automated extraction of vulnerability information for home computer security, in *Int. Symp. on Foundations and Practice of Security*, F. Cuppens, J. Garcia-Alfaro, N. Zincir Heywood, and P. Fong, eds. Springer, 2014, pp. 356–366.

[3] E. F. T. K. Sang and F. De Meulder, Introduction to the CONLL-2003 shared task: Language-independent named entity recognition, in *Proc. 7^{th} Conf. on Natural Language Learning at HLT-NAACL 2003 – Volume 4*, Edmonton, Canada, 2003, pp. 142–147.

[4] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, Neural architectures for named entity recognition, arXiv preprint: 1603.01360, 2016.

[5] X. J. Liao, K. Yuan, X. F. Wang, Z. Li, L. Y. Xing, and R. Beyah, Acing the IOC game: Toward automatic discovery and analysis of open-source cyber threat intelligence, in *Proc. 2016 ACM SIGSAC Conf. on Computer and Communications Security*, Vienna, Austria, 2016, pp. 755–766.

[6] L. Obrst, P. Chase, and R. Markeloff, Developing an ontology of the cyber security domain, in *STIDS*, 2012, pp. 49–56.

[7] R. Lal, Information extraction of security related entities and concepts from unstructured text, Master dissertation, University of Maryland Baltimore County, Baltimore, MD, USA, 2013.

[8] L. Luo, Z. H. Yang, P. Yang, Y. Zhang, L. Wang, H. F. Lin, and J. Wang, An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition, *Bioinformatics*, vol. 34, no. 8, pp. 1381–1388, 2018.

[9] A. Ritter, S. Clark, Mausam, and O. Etzioni, Named entity recognition in tweets: An experimental study, in *Proc. 2011 Conf. on Empirical Methods in Natural Language Processing*, Edinburgh, UK, 2011, pp. 1524–1534.

[10] E. Minkov, R. C. Wang, and W. W. Cohen, Extracting personal names from email: Applying named entity recognition to informal text, in *Proc. Conf. on Human Language Technology and Empirical Methods in Natural Language Processing*, Vancouver, Canada, 2005, pp. 443–450.

[11] S. More, M. Matthews, A. Joshi, and T. Finin, A knowledge-based approach to intrusion detection modeling, in *2012 IEEE Symp. on Security and Privacy Workshops*, San Francisco, CA, USA, 2012, pp. 75– 81.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, Efficient estimation of word representations in vector space, arXiv preprint: 1301.3781, 2013.

[13] J. R. Finkel, T. Grenager, and C. Manning, Incorporating non-local information into information extraction systems by Gibbs sampling, in *Proc. 43^{rd} Annual Meeting on Association for Computational Linguistics*, Ann Arbor, MI, USA, 2005, pp. 363–370.

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

Pingchuan Ma received the BS degree from Dalian University of Technology in 2017. Now he is pursuing the MS degree at Institute of Information Engineering, Chinese Academy of Sciences. His research interests include cybersecurity situational awareness and natural language processing.

Bo Jiang received the PhD degree from the University of Chinese Academy of Sciences in 2016. He is currently an assistant researcher at the Institute of Information Engineering, Chinese Academy of Sciences. His research interests include network security situational awareness, knowledge mapping, and data mining.

Ning Li received the PhD degree from Chinese Academy of Sciences in 2014. She is currently an assistant researcher at the Institute of Information Engineering, Chinese Academy of Sciences. Her research field is cybersecurity situational awareness.

Zhigang Lu received the PhD degree from Chinese Academy of Sciences in 2010. He is currently a senior engineer at the Institute of Information Engineering of the Chinese Academy of Sciences, and an associate professor at the School of Cyberspace Security, Chinese Academy of Sciences. His research interests include network security situational awareness, network attack detection, and mobile terminal security.

Zhengwei Jiang received the PhD degree from University of Chinese Academy of Sciences in 2014. He is currently a senior engineer at the Institute of Information Engineering, and an associate professor at the School of Cyberspace Security, Chinese Academy of Sciences. His research areas are threat intelligence, situational awareness, and cyber threat discovery.