Collecting Indicators of Compromise from Unstructured Text of Cybersecurity Articles using Neural-Based Sequence Labelling

Zi Long   Lianzhi Tan   Shengping Zhou
Platform and Content Group
Tencent
Shenzhen, Guangdong, China

Chaoyang He
AI Lab
Tencent
Shenzhen, Guangdong, China

Xin Liu
Platform and Content Group
Tencent
Shenzhen, Guangdong, China

Abstract—**Indicators of Compromise (IOCs) are artifacts observed on a network or in an operating system that can be utilized to indicate a computer intrusion and detect cyber-attacks in an early stage. Thus, they exert an important role in the field of cybersecurity. However, state-of-the-art IOCs detection systems rely heavily on hand-crafted features with expert knowledge of cybersecurity, and require large-scale manually annotated corpora to train an IOC classifier. In this paper, we propose using an end-to-end neural-based sequence labelling model to identify IOCs automatically from cybersecurity articles without expert knowledge of cybersecurity. By using a multi-head self-attention module and contextual features, we find that the proposed model is capable of gathering contextual information from texts of cybersecurity articles and performs better in the task of IOC identification. Experiments show that the proposed model outperforms other sequence labelling models, achieving the average F1-score of 89.0% on English cybersecurity article test set, and approximately the average F1-score of 81.8% on Chinese test set.**

I. INTRODUCTION

Indicators of Compromise (IOCs) are forensic artifacts that are used as signs when a system has been compromised by an attacker or infected with a particular piece of malware. To be specific, IOCs are composed of some combinations of virus signatures, IPs, URLs or domain names of botnets, MD5 hashes of attack files, etc. They are frequently described in cybersecurity articles, many of which are written in unstructured text, describing attack tactics, technique and procedures. For example, a snippet from a cybersecurity article is shown in Fig. 1. From the text , token "INST.exe" is the name of an executable file of a malicious software, and the file "ntdll.exe" downloaded by "INST.exe" is a malicious file as well. Obviously, these kinds of IOCs can be then utilized for early detection of future attack attempts by using intrusion detection systems and antivirus software, and thus, they exert an important role in the field of cybersecurity. However, with the rapid evolvement of cyber threats, the IOC data are produced at a high volume and velocity every day, which makes it increasingly hard for human to gather and manage them.

A number of systems are proposed to help discover and gather malicious information and IOCs from various types

Fig. 1. Example of IOCs contained in cybersecurity articles

of data sources [1]–[6]. However, most of those systems consist of several components that identify IOCs by using human-crafted features that heavily rely on specific language knowledge such as dependency structure, and they often have to be pre-defined by experts in the field of the cybersecurity. Furthermore, they need a large amount of annotated data used as the training data to train an IOC classifier. Those training data are frequently difficult to be crowed-sourced, because non-experts can hardly distinguish IOCs from those non-malicious IPs or URLs. Thus, it is a time-consuming and laborious task to construct such systems for different languages.

In this work, we consider the task of collecting IOCs from cybersecurity articles as a task of sequence labelling of natural language processing (NLP). By applying a sequence labelling model, each token in an unstructured input text is assigned with a label, and tokens assigned with IOC labels are then collected as IOCs. Recently, sequence labelling models have been utilized in many NLP tasks. Huang et al. [7] proposed using a sequence labelling model based on the bidirectional long short-term memory (LSTM) [8] for the task of named entity recognition (NER). Chiu et al. [9] and Lample et al. [10] proposed integrating LSTM encoders with character embedding and the neural sequence labelling model to achieve a remarkable performance on the task of NER as well as part-of-speech (POS) tagging. Besides, Dernoncourt et al. [11] and Jiang et al. [12] proposed applying the neural sequence labelling model to the task of de-identification of medical records.

Among the previous studies of the neural sequence labelling

task, Zhou el al. [13] firstly propose using an end-to-end neural sequence labelling model to fully automate the process of IOCs identification. Their model is on the basis of an artificial neural networks (ANN) with bidirectional LSTM and CRF. However, their newly introduced spelling features bring a more extraction of false positives, i.e., tokens that are similar to IOCs but not malicious. In this paper, we further introduce a multi-head self-attention module and contextual features to the ANN model so that the proposed model can perform better in gathering the contextual information from the unstructured text for the task of IOCs identification. Based on the results of our experiments, our proposed approach achieves an average precision of 93.1% and the recall of 85.2% on English cybersecurity article test set, and an average precision of 82.9% and recall of 80.7% on Chinese test set. We further evaluate the proposed model by training the model using both the English dataset and Chinese dataset, which even achieves better performance.

II. MODEL

Fig. 2 shows the 3 components (layers) of the proposed neural network architecture.

A. Token Embedding Layer

The token embedding layer takes a token as input and outputs its vector representation. As shown in Fig. 2, given an input sequence of tokens $x_1, \ldots, x_n$, the output vector e_i $(i = 1, \ldots, n)$ of each token x_i results from the concatenation of two different types of embeddings: token embedding $V_t(x_i)$ and the character-based token embeddings $\overrightarrow{b_i}$, $\overleftarrow{b_i}$ that come from the output of a character-level bi-LSTM encoder.

B. Sequence Representation Layer

The Sequence Representation Layer takes the sequence of embeddings e_i $(i = 1, \ldots, n)$ as input, and outputs a sequence $p_i (i = 1, \ldots, n)$, where the tth element of p_i represents the probability that the ith token has the label t.

Different from the previous work of sequence labelling in news articles or patient notes [10], [11], sentences from a cybersecurity report often contain a large number of tokens as well as lists of IOCs with little context, making it much more difficult for LSTM to encode the input sentence correctly. Therefore, instead of the token LSTM layer in [13], we propose sequence representation layer that consists of 3 modules, i.e., attention-based Bi-LSTM module, multi-head self-attention module and token feature module.

a) Attention-based Bi-LSTM: Considering that tokens cannot contribute equally to the representation of the input sequence, we introduce attention mechanism to Bi-LSTM to extract such tokens that are crucial to the meaning of the sentence. Then, we aggregate the representation of those informative words to form the vector of the input sequence.

The attention mechanism is similar to the one proposed by Yang et al. [14], which is defined as follows:

$$
\begin{aligned}
\boldsymbol{u}_i &= \tanh(W_w \boldsymbol{h}_i + \boldsymbol{b}_w) \\
\alpha_i &= \frac{\exp(\boldsymbol{u}_i^\top \boldsymbol{u}_w)}{\sum_i \exp(\boldsymbol{u}_i^\top \boldsymbol{u}_w)} \\
\boldsymbol{s} &= \sum_i \alpha_i \boldsymbol{h}_i
\end{aligned}
$$

That is to say, we first compute the $\boldsymbol{u}_i$ as a hidden representation of the hidden states of Bi-LSTM $\boldsymbol{h}_i$ for i^{th} input token, where $\boldsymbol{h}_i$ is obtained by concatenating the ith hidden states of forward and backward LSTM, i.e., $\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i]$. Then, we measure the importance of the ith token with a trainable vector $\boldsymbol{u}_w$ and get a normalized importance weight α_i through a softmax function. After that, the sentence vector $\boldsymbol{s}$ is computed as a weighted sum of $\boldsymbol{h}_i$ $(i = 1, \ldots, n)$. Here, weight matrix W_w, bias $\boldsymbol{b}_w$ and vector $\boldsymbol{u}_w$ are randomly initialized and jointly learned during the training process. Note that each input sentence merely has one sentence vector $\boldsymbol{s}$ as its weighted representation, and $\boldsymbol{s}$ is then used as a part of the ith output of attention-based Bi-LSTM module, where $\boldsymbol{h}_i^L = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i; \boldsymbol{s}]$ $(i = 1, \ldots, n)$.

b) Multi-head self-attention: Motivated by the successful application of self-attention in many NLP tasks [15], [16], we add a multi-head self-attention module to enhance the embedding of each word with the information of other words in a text adaptively. By means of this, the local text regions where convolution performs carry the global information of text. Following the encoder part of Vaswani et al. [15], multi-head self-attention module is composed of a stack of several identical layers, each of which consists of a multi-head self-attention mechanism and two convolutions with kernel size 1. Given the sequence of embeddings e_i as input, and the output is defined as follows:

$$
\begin{aligned}
\text{Attention}(Q, K, V) &= \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \\
\text{MultiHead}(Q, K, V) &= [\text{head}_1; \ldots; \text{head}_h] \\
where\ \text{head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)
\end{aligned}
$$

where, W_i^Q, W_i^K, W_i^V are parameter matrices for the projections of queries Q, keys K and values V in the ith head, respectively. Here, Q, K and V are set as the input sequence e_i $(i = 1, \ldots, n)$. The MultiHead(Q, K, V) is then given to the two convolutions and the output of multi-head self-attention $\boldsymbol{h}_i^M$ $(i = 1, \ldots, n)$ is obtained.

c) Token features: Furthermore, we introduce some features to defined IOCs to improve the performance of the proposed model on a very small amount of training data. Here, we define two types of features, i.e., spelling features and contextual features, and map each token x_i $(x = 1, \ldots, n)$ to a feature vector $feature(x_i) = [feature^s(x_i); feature^c(x_i)]$, where $feature^s(x_i)$ is the spelling feature vector and $feature^c(x_i)$ is the contextual feature vector. Note that the values of features are then jointly learned during the process
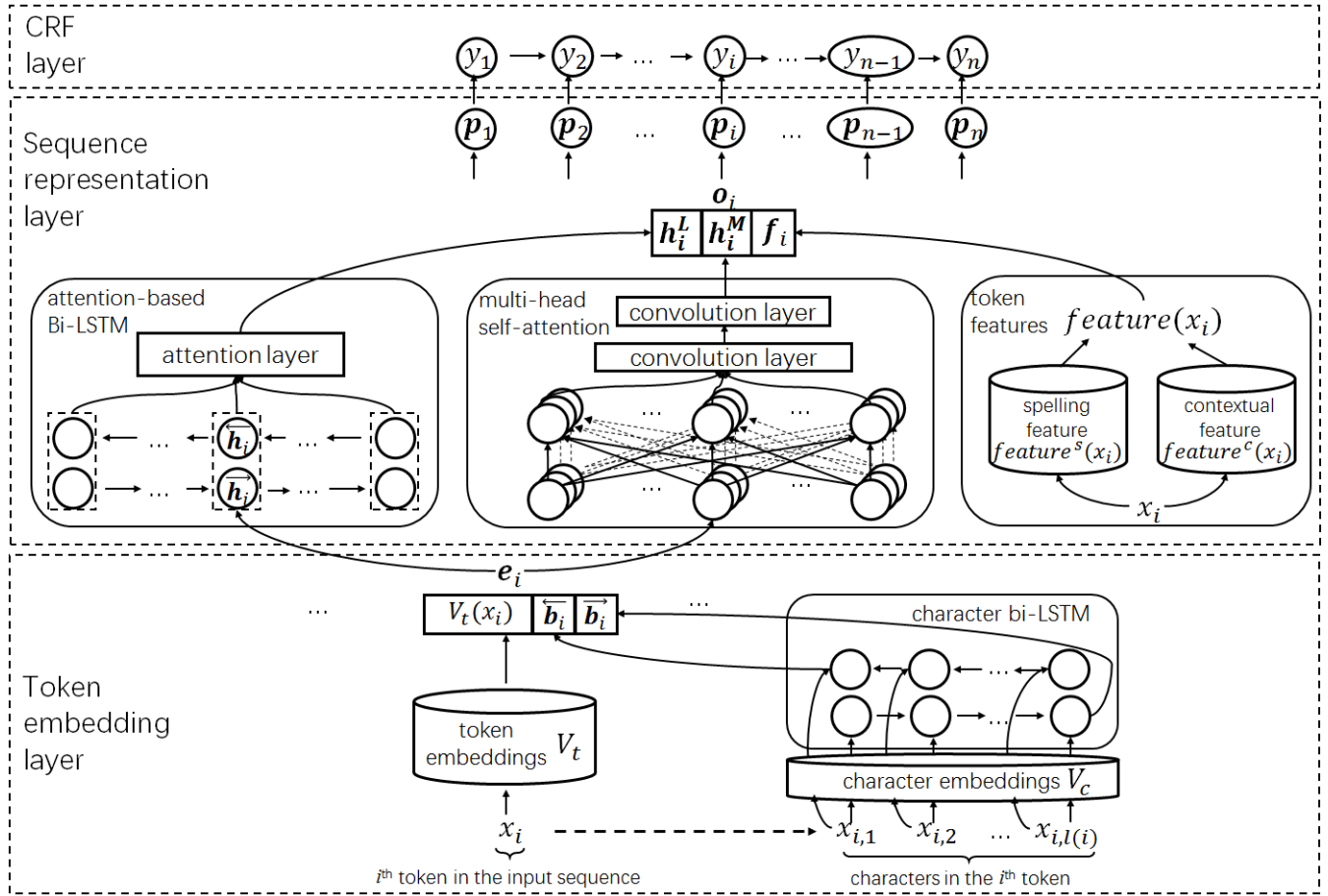
paper N-19774.pdf

Fig. 2. ANN model of sequence labeling for IOCs automatic identification

of training. In Section III, we will explain the features in more detail.

As shown in Fig. 2, the vector $\boldsymbol{o}_i$ ($i = 1, \ldots, n$) is a concatenation of the $\boldsymbol{h}_i^L$, $\boldsymbol{h}_i^M$ and $f(i)$. Each vector $\boldsymbol{o}_i$ is then given to a feed-forward neural network with one hidden layer, which outputs the corresponding probability vector $\boldsymbol{p}_i$.

C. CRF Layer

We also introduce a CRF layer to output the most likely sequence of predicted labels. The score of a label sequence $\boldsymbol{y} = y_1, \ldots, y_n$ is defined as the sum of the probabilities of unigram labels and the bigram label transition probabilities:

$$score(\boldsymbol{y}) = \sum_{i=1}^{n} p_i[y_i] + \sum_{i=2}^{n} T[y_{i-1}, y_i]$$

where T is a matrix that contains the transition probabilities of two subsequent labels. Vector $\boldsymbol{p}_i$ is the output of the token LSTM layer, and $\boldsymbol{p}_i[y_i]$ is the probability of label y_i in $\boldsymbol{p}_i$. $T[g, h]$ is the probability that a token with label g is followed by a token with the label h. Subsequently, these scores are turned into probabilities of the label sequence by taking a softmax function over all possible label sequences.

III. Features

We extract a vector of features for each tokens of input sequences. In this section, we present each feature category in detail.

A. Spelling Features

Since the IOCs tend to follow fixed patterns, we predefined several regular expressions and spelling rules to identify IOCs. For example, to identify a URL, we defined a regular expression `http(s)?:\\[0-9a-zA-Z_\.\-\\]+` and set the value of the URL feature to 1 when the input token matches the regular expression.[1] However, such expressions and spelling rules could introduce false positives, i.e., tokens that have the same spelling patterns as IOCs but are not malicious. In this work, we further introduce the contextual features as described next.

B. Contextual Features

IOCs in cybersecurity articles are often described in a predictable way: being connected to a set of contextual keywords [2], [17]. For example, a human user can infer that the word "ntdll.exe" is the name of a malicious file on the basis of the

---

[1]Details of the spelling features can be found in [13].

paper N-19774.pdf

words "download" and "compromised" from the text shown in Fig. 1. By analyzing the whole corpus, it is interesting that malicious file names tends to co-occur with words such as "download", "malware", "malicious", etc. In this work, we consider words that can indicate the characteristics of the neighbor words as contextual keywords and develop an approach to generate features from the automatically extracted contextual keywords.

Taking the above into account, we introduce the contextual feature vector $feature^c(x)$ for a given input token x, where the ith element of $feature^c(x)$ is defined as follows:

$$feature_i^c(x) = \frac{freq_w(k_i; x, size)}{freq(x)}$$
$$(k_i \in K, i = 1, \ldots, |K|)$$

$freq(x)$ is the frequency of token x in the whole corpus, while $freq_w(k_i; x, size)$ is the frequency of contextual keyword k_i from the windowed portions of the texts centering on the token x in the whole corpus and $size$ is the size of window. The set of contextual keywords K are automatically extracted from the annotated texts, where each contextual keyword $k_i(k_i \in K)$ satisfies the following conditions:

(1) $\sum_{t \in T} freq_w(k_i; t, size) \geq lb$, where T is the set of manually annotated IOCs and lb is a the lower bound of the frequency.

(2) k_i is not a punctuation or stopword[2].

Note that we extract contextual keywords only from manually annotated data (e.g., training set), while we compute the contextual feature vector in all of the unlabeled data. According to this definition, it is obvious that the dimension of the contextual feature vector is as the same as the number of extracted contextual keywords. The size of window $size$ and the lower bound of frequency lb are then tuned by the validation set.

C. Usage of Features

The feature vector for an input token is the concatenation of the token spelling feature vector and the contextaul feature vector. Here, to elucidate the best usage of the feature vector, we evaluate the feature vector by concatenating it at different locations in the proposed model, i.e., the input of the token LSTM layer ($e_i = [V_t(x_i); \overrightarrow{b_i}; \overleftarrow{b_i}; feature(x_i)]$), the hidden state of the token LSTM ($h_i = [\overrightarrow{h}_i; \overleftarrow{h}_i; feature(x_i)]$), and the output of token LSTM ($o_i = [h_i; s; feature(x_i)]$). Among them, to concatenate the feature vector with the LSTM hidden state vector and the sentence vector of attention in the token LSTM layer, as shown in Section II-B, achieved the best performance. We speculate that the features played an important role in the task of IOCs identification and feature vectors near the output layer were able to improve the performance more significantly than those at other locations.

[2]https://github.com/hankcs/HanLP/blob/master/data/dictionary/stopwords.txt

TABLE I
STATISTICS OF DATASETS (NUMBERS OF TRAINING / VALIDATION / TEST SET)

	English Dataset	Chinese Dataset
attacker	5,304 / 1,067 / 1,609	742 / 230 / 126
attack method	2,737 / 610 / 882	126 / 41 / 27
attack target	3,055 / 1,055 / 695	161 / 17 / 15
domain	6,443 / 1,054 / 1,701	1,682 / 438 / 468
e-mail address	1,284 / 154 / 222	67 / 16 / 49
file hash	10,367 / 2,055 / 2,459	2,484 / 1,231 / 1,055
file information	4,353 / 1,024 / 1,131	931 / 469 / 531
IPv4	3,012 / 729 / 819	969 / 252 / 264
malware	7,317 / 1,585 / 1,974	2,958 / 1,524 / 1,035
URL	1,849 / 105 / 156	1,618 / 333 / 487
vulnerability	1,557 / 309 / 359	469 / 166 / 143
tokens	1,169,896 / 253,336 / 350,406	775,549 / 282,600 / 249,946
paragraphs	6,702 / 1,453 / 2,110	6,423 / 2,314 / 2,059
articles	250 / 70 / 70	363 / 122 / 122

IV. EVALUATION

A. Datasets

For English dataset, we crawl 687 cybersecurity articles from a collection of advanced persistent threats (APT) reports which are published from 2008 to 2018[3]. All of these cybersecurity articles are used to train the English word embedding. Afterwards, we randomly select 370 articles, and manually annotate the IOCs contained in the articles. Among the selected articles, we randomly select 70 articles as the validation set and 70 articles as the test set; the remaining articles are used for training.

For Chinese dataset, we crawl 5,427 cybersecurity articles online from 35 cybersecurity blogs which are published from 2001 to 2018. All of these cybersecurity articles are used to train the Chinese word embedding. Afterwards, we randomly select 607 articles, and manually annotate the IOCs contained in the articles. Among the selected articles, we randomly select 122 articles as the validation set and 122 articles as the test set; the remaining articles are used for training.

TABLE I shows statistics of the datasets. The output labels are annotated with the BIO (which stands for "Begin", "Inside" and "Outside") scheme.

B. Training Details

For pre-trained token embedding, we apply word2vec [18] to all crawled 687 English APT reports and 5,427 Chinese cybersecurity articles described in Section IV-A respectively. The word2vec models are trained with a window size of 8, a minimum vocabulary count of 1, and 15 iterations. The negative sampling number of word2vec is set to 8 and the model type is skip-gram. The dimension of the output token embedding is set to 100.

The ANN model is trained with the stochastic gradient descent to update all parameters, i.e., token embedding, character embedding, parameters of Bi-LSTM, weights of sentence

[3]https://github.com/CyberMonitor/APT_CyberCriminal_Campagin_Collections

paper N-19774.pdf

Models	English Dataset			Chinese Dataset		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Baseline	47.1	58.8	52.3	60.1	42.7	49.9
Huang et al. [7]	64.8	33.6	51.6	60.2	35.0	44.3
Lample et al. [10]	83.0	75.2	78.9	78.3	71.8	74.9
Rei et al. [19]	81.6	74.5	77.9	76.1	71.0	73.5
Zhou et al. [13]	90.4	**87.2**	88.8	80.8	80.1	80.4
Our model	**93.1**	85.2	**89.0**	**82.9**	**80.7**	**81.8**

attention, weights of multi-head self-attention, token features, and transition probabilities of CRF layers at each gradient step. For regularization, the dropout is applied to the output of each sub layer of the ANN model. Further training details are given below: (a) For attention-based Bi-LSTM module, dimensions of character embedding, hidden states of character-based token embedding LSTM, hidden states of Bi-LSTM, and sentence attention are set to 25, 25, 100 and 100, respectively. For multi-head self-attention module, we employ a stack of 6 multi-head self attention layer, each of which has 4 head and dimension of each head is set to 64. (b) All of the ANNs parameters are initialized with a uniform distribution ranging from -1 to 1. (c) We train our model with a fixed learning rate of 0.005. The minimum number of epochs for training is set as 30. After the first 30 epochs had been trained, we compute the average F1-score of the validation set by the use of the currently produced model after every epoch had been trained, and stop the training process when the average F1-score of validation set fails to increase during the last ten epochs. We train our model for, if we do not early stop the training process, 100 epochs as the maximum number. (d) We rescale the normalized gradient to ensure that its norm does not exceed 5. (e) The dropout probability is set to 0.5.

C. Results

As shown in TABLE II, we report the micro average of precision, recall and F1-score for all 11 types of labels for a baseline as well as the proposed model. As the baseline, we simply judge the input token as IOCs on the basis of the spelling features described in [13]. As presented in TABLE II, the score obtained by the proposed model is clearly higher than the baseline. Here, as described in Section III-B, the sizes of window and lower bounds of frequency for selecting contextual keywords are tuned as 4 and 7 throughout the evaluation of English dataset, and tuned as 3 and 4 throughout the evaluation of Chinese dataset. The number of extracted contextual keywords from the English dataset is 1,328, and from the Chinese dataset is 331.

Furthermore, we quantitatively compare our study with other typical works of sequence labelling, i.e., the work of Huang et al. [7], the work of Lample et al. [10] and the work of Rei et al. [19]. Huang et al. [7] proposed a bidirectional LSTM model with a CRF layer, including hand-

crafted features specialized for the task of sequence labelling. Lample et al. [10] described a model where the character-level representation was concatenated with word embedding and Rei et al. [19] improved the model by introducing an attention mechanism to the character-level representations. We train these models by employing the same training set and training parameters as the proposed model. As shown in TABLE II, the proposed model obtains the highest precision, recall and F1-score than other models in the task of IOCs extraction. Compared with the second-best model of Lample et al. [10], the performance gain of the proposed model on the English dataset is approximately 10.1% of precision and 10.0% of recall. The performance gain of the proposed model on the Chinese dataset is approximately 4.2% of precision and 9.0% of recall.

We also quantitatively compare our study with the work of Zhou et al. [13], which proposed a bidirectional LSTM model with a CRF layer, including hand-crafted spelling features for the task of IOC identification. As shown in TABLE II, the proposed model obtains a slightly higher F1-score on the English dataset and significantly higher F1-score on the Chinese dataset.

TABLE III compares several examples of correct IOC extraction produced by the proposed model with one by the work of Lample et al. [10]. In the first example, the model of Lample et al. [10] fails to identify the malicious URL "http://www7.chrome-up.date/0m5EE", because the token only appears in the test set and consists of several parts that are uncommon for URLs, such as "www7" and "date", and thus both the token embedding and the character embedding lack proper information to represent the token as a malicious URL. The proposed model correctly identifies the URL, where the token is defined as a URL by spelling features and is then identified as a malicious URL by the use of the context information. In the second example, the model of Lample et al. [10] fails to identify token "cr.sh" of the input Chinese text as a malicious file name, while the token is assigned with a correct label by the proposed model. It is mainly because that the token "cr.sh" is defined as a token of file information by spelling features and tends to co-occur with words, "下载"(download) and "挖矿程序"(mining software). These two words often appear nearby malicious file information and are then extracted as contextual keywords in Section III-B. The token "cr.sh" is then correctly identified as a token of malicious file information by the use of the contextual features.

D. Analysis of Contextual Features

The proposed model provides an intuitive way to inspect the contextual information of each given token. As described in Section III-B, we initialize the contextual features of each given token using the automatically extracted contextual keywords and jointly learn them during the process of training with the whole ANN model. To prove the effectiveness of the contextual features, we visualize the learned weights matrix of each contextual keyword of contextual feature and

TABLE III
EXAMPLES OF CORRECT IDENTIFICATION BY THE PROPOSED MODEL

| Input & Reference	The **shellcode** *attack method* excuted by this command is the same as in the delivery documents as well, specifically taken from Metasploit to obtain additional **shellcode** *attack method* to execute using an HTTP request to the following URL **http://www7.chrome-up.date/0m5EE** *url* .
Our model	The **shellcode** *attack method* excuted by this command is the same as in the delivery documents as well, specifically taken from Metasploit to obtain additional **shellcode** *attack method* to execute using an HTTP request to the following URL **http://www7.chrome-up.date/0m5EE** *url* .
Lample et al. (2016)	The **shellcode** *attack method* excuted by this command is the same as in the delivery documents as well, specifically taken from Metasploit to obtain additional **shellcode** *attack method* to execute using an HTTP request to the following URL http://www7.chrome-up.date/0m5EE . *(failed to extract the malicious url)*
Input & Reference	**cr.sh** *(file information)* 脚本中内置了两个挖矿程序的MD5，如果当前主机中的挖矿程序MD5不在这两个 MD5中，则**cr.sh** *(file information)* 会从指定的服务器上下载对应的挖矿程序。 (The script **cr.sh** contains the MD5 hashes of two mining softwares. If the MD5 hash of mining software in the current host is different from those of the script, **cr.sh** will download the mining softwares from the specified server.)
Our model	**cr.sh** *(file information)* 脚本中内置了两个挖矿程序的MD5，如果当前主机中的挖矿程序MD5不在这两个 MD5中，则**cr.sh** *(file information)* 会从指定的服务器上下载对应的挖矿程序。
Lample et al. (2016)	cr.sh *(failed to extract the malicious file information)* 脚本中内置了两个挖矿程序的MD5，如果当前主机中的挖矿程序MD5不在这两个 MD5中，则cr.sh会从指定的服务器上下载对应的挖矿程序。 *(failed to extract the malicious file information)*
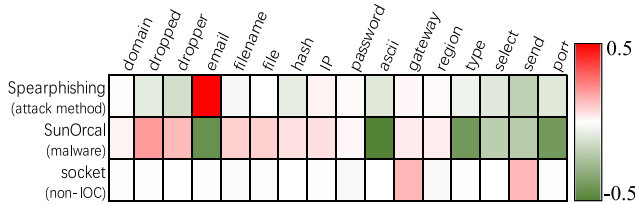
Fig. 3. Heatmap of part of contextual features martix in the English dataset

TABLE IV
TOP 10 AND BOTTOM 10 LARGEST WEIGHTED CONTEXTUAL KEYWORDS
OF CONTEXTUAL FEATURE IN THE ENGLISH DATASET

	top−10	bottom−10
IOC token	hash, domain, server, filename, ip, dropped, dropper, register, request, email	function, content, campaign, payload, key, sample, referred, specialized, sources, effort
non-IOC token	ascii, password, researcher, eset, select, communicate, java, gateway, type, region	indicators, dropped, port, copies, lead, detection, dropper, send, PDB, register

show several examples in Fig. 3. Each row of the matrix in each plot indicates the weights of contextual keywords for the given tokens. From this we see which contextual keyword were considered more important to represent the contextual information of the given token. We can see from the matrix in Fig. 3 that, for the token "spearphshing", which is an email-spoofing attack method, the contextual keyword

"email" has the largest weight. For the malware "SunOrcal", which drops several malicious executable files, contextual keywords "droppper" and "dropper" have larger weights than other contextual keywords such as "ascii", "port" and "type". For non-IOC token "socket", contextual keywords "gateway" and "port" yield larger weights than other keywords because "socket" tends to co-occur with "gateway" and "port".

We further calculate the average weight of each contextual keyword and show the top 10 and bottom 10 largest weighted contextual keywords in TABLE IV. From this we see that contextual keywords such as, "hash" and "filename", which tends to co-occur with malicious filenames, have the largest weights for IOCs, while the contextual keywords such as "ascii", "password" have the largest weights for non-IOCs. Here, it is interesting to find that contextual keyword "dropped" and "droppper", which tend to co-occur with malicious file information and malwares, yield large weights for IOCs but small weights for non-IOCs. The proposed ANN model benefits from the differences of contextual information between IOCs and non-IOCs that is represented by the contextual features, and thus, achieves better performance than the previous works.

E. Training the Proposed Model with Bilingual Data

Even though security articles are written in different languages, most of the IOCs are written in English, and are described in a similar pattern. Therefore, using multilingual corpora could be a solution for addressing the lack of annotated data, and the performance of the proposed model is expected

paper N-19774.pdf

TABLE V
COMPARISON OF EVALUATION RESULTS WHEN TRAINING THE PROPOSED
MODEL WITH DIFFERENT TRAINING SETS (MICRO AVERAGE PRECISION /
RECALL / F1-SCORE FOR 11 LABELS)

	English test set	Chinese test set
Only English training set	**93.1** / 85.2 / 89.0	- / - / -
Only Chinese training set	- / - / -	82.9 / 80.7 / 81.8
English + Chinese training set	91.6 / **87.3** / **89.6**	**85.6** / **83.3** / **84.4**

to be improved by extending the training set. To examine the hypothesis, we ran a number of additional experiments using both the English dataset and Chinese dataset, both of which are described in Section IV-A and are not parallel data or comparable data.

As pre-trained word embeddings for the bilingual training dataset, we applied a cross-lingual word embedding obtained by the work of Duong el al [20], where the English-Chinese cross-lingual dictionary is obtained by simply translating all the English words from English dataset to Chinese and Chinese words from Chinese dataset to English using Google translation[4]. As contextual feature vector, we concatenate the contextual feature vector obtained from English dataset with the contextual feature vector obtained from Chinese dataset. Then we merge the English training set and the Chinese training set into one set and train the proposed model with the merged bilingual training set. TABLE V shows that the proposed model trained with the English training set and Chinese training set achieves a small improvement of F1-score on English test set when compared with the model trained with only English training set, and a great improvement of F1-score on Chinese test set when compared with the model trained with only Chinese training set.

We compare scores of each label when the proposed model is trained with different training sets in TABLE VI. When using the English test set, the F1-scores of labels "attack method", "attack target" and "malware" by the model trained with the English training set and Chinese training set are lower than those scores by the model trained with only the English training set. It is mainly because that tokens of these labels can be written in different languages, which harms the model trained with the bilingual training data set. In contrast, benefiting from the extension of training set, for types of labels that are often written in English, e.g., "domain ", "file imformation", "IPv4" and "vlunerability", the proposed model trained with the English training set and the Chinese training set achieves higher scores than the model trained with only the English training set. When using the Chinese test set, the proposed model trained with the English training set and the Chinese training set obtained a obviously higher F1-scores than the model trained with only the Chinese training set for almost all the types of labels. It is interesting to find that types of labels "e-mail address", "attack method", "attacker", which lack of instances in Chinese training set, show the biggest improvement by using the model trained with the bilingual

[4]https://translate.google.com

training set.

V. RELATED WORK

a) NLP in cybersecurity: Few references in cyber security utilize natural language processing. Neuhaus and Zimmermann [21] analyze the trend of vulnerability by applying latent Dirichlet allocation to vulnerability description. Liao et al. [2] put forward a system to automatically extract IOC items from blog posts. Husari et al. [3] proposed a system that automatically extracted threat actions from unstructured threat intelligence reports by utilizing a pre-defined ontology. A concurrent work by Zhu et al. [6] automatically extracted IOC data from security technical articles and further categorized them into different stages of malicious campaigns. All of those systems consist of several components that rely heavily on manually defined rules, while our proposed model is an end-to-end model using word embedding and token features as input, which is more general and applicable to a broader area.

b) Neural sequence labelling models: There are amount of ANN-based works in the area of sequence labelling. Collobert et al. [22] described one of the first task-independent neural sequence labelling model on the basis of convolutional neural networks. Hammerton [23] first proposed a sequence labelling model with LSTM. Huang et al. . [7] proposed using a sequence labelling model based on the bidirectional LSTM for the task of name entity recognition (NER). Lample et al. [10] proposed integrating LSTM encoders with character embedding and the neural sequence labelling model. Rei et al. [19] improved the model by introducing an attention mechanism to the character-level representations. Dernoncourt et al. [11] proposed applying the neural sequence labelling model to the task of de-identification of medical records. Liu et al. [24] proposed a sequence labeling framework, which effectively leverages the language model to extract character-level knowledge. One appealing property of those works is that they can achieve excellent performance with a unified architecture and without task-specific feature engineering. It remains unclear whether such works can be used for tasks without a large amount of training data. Several works such as Yang et al. [25] and Lee et al. [26] proposed applying transfer learning to NER using a limited number of training corpora. Nevertheless, a large dataset that has same labels as the small training dataset is required for transfer learning, which is hard to obtain in the field of cybersecurity. In this paper, we introduce several spelling features without expert knowledge of cybersecurity to the neural model, and achieve excellent performance even using a small dataset for training.

VI. CONCLUSIONS

To conclude, in this paper, we newly introduce a multi-head self-attention module and contextual features to the neural based sequence labelling model, which significantly improved the performance in the task of IOC identification. Based on the evaluation results of our experiments, our proposed model is proved effective on both the English test set and the Chinese test set. We further evaluated the proposed model by training

paper N-19774.pdf

TABLE VI

EVALUATION RESULTS FOR EACH LABEL WHEN TRAINING THE PROPOSED MODEL WITH DIFFERENT TRAINING SETS (PRECISION / RECALL / F1-SCORE)

	English test set		Chinese test set	
	Only English training set	English + Chinese training set	only Chinese training set	English + Chinese training set
attacker	96.7 / 74.5 / 84.2	93.1 / 83.4 / **88.0**	81.6 / 38.9 / 52.7	86.8 / 38.5 / **53.4**
attack method	90.6 / 93.4 / **92.0**	90.7 / 91.9 / 91.3	16.3 / 29.6 / 21.1	70.0 / 33.7 / **45.5**
attack target	90.3 / 86.4 / **88.3**	90.7 / 80.4 / 85.2	22.2 / 40.0 / 28.6	87.1 / 66.7 / **74.5**
domain	93.2 / 93.5 / 93.4	91.6 / 97.1 / **94.3**	79.9 / 88.5 / 84.0	79.2 / 91.9 / **85.1**
e-mail address	98.4 / 84.4 / 90.8	97.5 / 91.0 / **94.1**	20.4 / 20.4 / 20.4	81.3 / 53.1 / **64.2**
file hash	90.3 / 99.9 / **94.8**	89.7 / 99.9 / 94.5	99.2 / 99.4 / **99.3**	98.9 / 99.8 / **99.3**
file information	90.0 / 69.5 / 78.4	89.6 / 75.2 / **81.8**	58.0 / 87.2 / 67.7	60.9 / 88.0 / **74.1**
IPv4	92.5 / 93.1 / 92.8	92.2 / 93.9 / **93.0**	92.7 / 88.8 / **90.7**	92.3 / 89.3 / **90.7**
malware	97.3 / 68.8 / **80.6**	96.4 / 67.9 / 79.6	93.9 / 59.7 / 69.7	92.6 / 64.0 / **72.0**
URL	99.7 / 92.3 / **95.9**	98.9 / 92.5 / 95.6	82.5 / 83.1 / 82.8	84.9 / 86.2 / **85.5**
vulnerability	97.9 / 89.7 / 93.6	93.8 / 96.1 / **94.9**	98.6 / 96.5 / **97.5**	96.1 / 98.8 / **97.5**
micro average	93.1 / 85.2 / 89.0	91.6 / 87.3 / **89.6**	82.9 / 80.7 / 81.8	85.6 / 83.3 / **84.4**

the proposed model using both the English training set and the Chinese training set and compared it with models that are trained with only one training set, where the model trained with the merged bilngual training set performs better.

One of our future works is to integrate the contextual embeddings from the bidirectional language model into our proposed model. The pretrained neural language models are proved effective in the sequence labelling models [27]–[29]. It is expected to improve the performance of the proposed model by integrating both the contextual features and contextual embeddings into the neural sequence labelling model.

REFERENCES

[1] Z. Zhu and T. Dumitras, "FeatureSmith: automatically engineering features for malware detection by mining the security literature," in *Proc. CCS 16*, 2016, pp. 767–778.

[2] X. Liao, K. Yuan, X. Wang, Z. Li, L. Xing, and R. Beyah, "Acing the ioc game: toward automatic discovery and analysis of open-source cyber threat intelligence," in *Proc. CCS 16*, 2016, pp. 755–766.

[3] G. Husari, E. Al-Shaer, M. Ahmed, B. Chu, and X. Niu, "TTPDrill: automatic and accurate extraction of threat actions from unstructured text of cti sources," in *Proc. ACSAC 2017*, 2017, pp. 103–112.

[4] C. Huang, S. Hao, L. Invernizzi, J. Liu, Y. Fang, C. Kruegel, and G. Vigna, "Gossip: automatically identifying malicious domains from mailing list discussions," in *Proc. ASIA CCS 17*, 2017, pp. 494–505.

[5] B. J. Kwon, V. Srinivas, A. Deshpande, and T. Dumitras, "Catching worms, trojan horse and PUPs: unsupervised detection of silent delivery campagins," in *Proc. NDSS 17*, 2017.

[6] Z. Zhu and T. Dumitras, "ChainSmith: automatically learning the semantics of malicious campaigns by mining threat intelligence reports," in *Proc. EuroS&P 2018*, 2018.

[7] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv:1508.01991*, 2015.

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

[9] J. P. Chiu and E. Nichols, "Named entity recognition with bidirectional LSTM-CNNs," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 357–370, 2016.

[10] G. Lample, M. Ballesteros, S. Subramanian, K. Kwakami, and C. Dyer, "Neural architectures for name entity recognition," in *Proc. NAACL 2016*, 2016, pp. 260–270.

[11] F. Dernoncourt, J. Y. Lee, O. Uzuner, and P. Szolovits, "De-identification of patient notes with recurrent neural networks," *Journal of the American Medical Informatics Association*, vol. 24, no. 3, pp. 596–606, 2017.

[12] Z. Jiang, C. Zhao, B. He, Y. Guan, and J. Jiang, "De-identification of medical records using conditional random fields and long short-term memory networks," *Journal of Biomedical Informatics*, vol. 75, pp. S43–S53, 2017.

[13] S. Zhou, Z. Long, L. Tan, and H. Guo, "Automatic identification of indicators of compromise using neural-based sequence labelling," in *Proc. PACLIC 2018*, 2018.

[14] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proc. NAACL-HLT 2016*, 2016, pp. 1480–1489.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. . Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. 31st NeurIPS*, 2017, pp. 5998–6008.

[16] J. Xie, Z. Yang, G. Neubig, N. A. Smith, and J. Carbonell, "Neural cross-lingual named entity recognition with minimal resources," in *Proc. EMNLP 2018*, 2018, pp. 369–379.

[17] M. Darling, G. Heileman, G. Gressel, A. Ashok, and P. Poornachandran, "Lexical approach for classifying malicious urls," in *IEEE International Conference on High Performance Computing & Simulation (HPCS)*, 2015, pp. 195–202.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[19] M. Rei, G. K. O. Crichton, and S. Pyysalo, "Attending to characters in neural sequence labeling models," in *Proc. COLING 2016*, 2016.

[20] L. Duong, H. Kanayama, T. Ma, S. Bird, and T. Cohn, "Learning crosslingual word embeddings without bilingual corpora," in *Proc. EMNLP 2016*, 2016, pp. 1285–1295.

[21] S. Neuhaus and T. Zimmermann, "Security trend analysis with CVE topic model," in *IEEE 21st International Symposium on Software Reliability Engineering*, 2010, pp. 111–120.

[22] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scatch," *The Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2001.

[23] J. Hammerton, "Named entity recognition with long short-term memory," in *Proc. CONLL 2003*, 2003, pp. 172–175.

[24] L. Liu, J. Shang, X. Ren, F. F. Xu, H. Gui, J. Peng, and J. Han, "Empower sequence labeling with task-aware neural language model," in *Proc. 32nd AAAI*, 2018, pp. 5253–5260.

[25] Z. Yang, R. Salakhutdinov, and W. W. Cohen, "Transfer learning for sequence tagging with hierarchical recurrent networks," in *Proc. ICLR 2017*, 2017.

[26] J. Y. Lee, F. Dernoncourt, and P. Szolovits, "Transfer learning for named-entity recognition with neural networks," in *Proc. LERC 2018*, 2018, pp. 4471–4473.

[27] M. E. Peters, W. Ammar, C. Bhagavatula, and R. Power, "Semi-supervised sequence tagging with bidirectional language models," in *Proc. 55th ACL*, 2017, pp. 1756–1765.

[28] Y. Xiong and M. Nuo, "Attention-based blstm-crf architecture for mongolian named entity recognition," in *Proc. PACLIC 2018*, 2018.

[29] M. E. Peters, M. Neumanm, M. Iyyer, and M. Gardner, "Deep contextualized word representations," in *Proc. NAACL-HLT 2018*, 2018, pp. 2227–2237.