Hindawi
Security and Communication Networks
Volume 2022, Article ID 8477260, 12 pages
https://doi.org/10.1155/2022/8477260

Research Article

A Novel Threat Intelligence Information Extraction System Combining Multiple Models

Yongfei Li, Yuanbo Guo, Chen Fang, Yingze Liu, and Qingli Chen

PLA Information Engineering University, Zhengzhou 450001, China

Correspondence should be addressed to Yongfei Li; leekgfly@foxmail.com

Received 19 October 2022; Revised 23 November 2022; Accepted 1 December 2022; Published 9 December 2022

Academic Editor: Jie Cui

The increasing number of cyberattacks has made the cybersecurity situation more serious. Thus, it is urgent to use cyber threat intelligence to deal with the complex and changing cyber environment. However, cyber threat intelligence usually exists in an unstructured form, and a huge amount of data poses a great challenge to security analysts. To this end, this paper proposes a novel threat intelligence information extraction system combining multiple models, which contains four key steps: entity extraction, coreference resolution, relation extraction, and knowledge graph construction. In the entity extraction task, a multihead self-attention mechanism is adopted to extract the dependency relationships between words. In the coreference resolution task, contextual information and mention embedding are fused to improve the mention representation. Meanwhile, features of different dimensions are extracted using a convolutional neural network. In the relation extraction task, additional features such as part of speech, mention width, entity type, and distance of entity pairs are incorporated to improve the embedding representation. Finally, a knowledge graph is constructed to explicitly present entities and their relationships. Experimental results indicate that compared with the baseline model, the $F1$ score of our model is improved by at least 8.87, 9.82, and 10.56 on entity extraction, coreference resolution, and relation extraction, respectively. The knowledge graph in Neo4j demonstrates the effectiveness of our system.

1. Introduction

Modern IT infrastructures are under different degrees of cyberattacks. To address this problem, it is necessary to continuously monitor devices, collect and process information, and generate security alerts. To better understand the threat situation and coordinate the response to unknown threats, security experts have proposed to use Cyber Threat Intelligence (CTI) for cyber defense. In 2013, Gartner first pointed out that threat intelligence is knowledge about existing or upcoming threats against an asset, including scenarios, mechanisms, indicators, revelations, and actionable recommendations that can provide subjects with threat response strategies. Threat intelligence and intrusion detection systems [1–3] are different in two ways. On the one hand, cyber threat intelligence is created after the attack or defense. It can be used to provide informational support for the response to threats. While intrusion detection systems will alert when an attack is performed. On the other hand, threat intelligence can be obtained from security vendor bulletins, hacker forums, social media, and information security vulnerability databases. It helps to understand the behavior of cyberattacks and how they happen. However, due to the complex composition of the Internet, variable attacker behaviors, and an increasing number of security devices, threat intelligence has grown geometrically. Meanwhile, cyber threat intelligence is usually in the form of natural language, with related entities scattered throughout the article and intricate relationships between entities. This poses challenges to intelligence analysis, utilization, and sharing. The huge amount of alert data puts a lot of pressure on security analysts. As a result, many alarms are left unprocessed and become junk data. Therefore, analyzing and dealing with threat information has become a key problem to be solved.

Manual analysis of threat intelligence requires cybersecurity expertise and is time-consuming and laborious, so it

is difficult to cope with the increasing number of cyberattacks. Given its importance, much research efforts has devoted to extracting structured knowledge from unstructured threat intelligence, and the extraction process mainly involves four key techniques: entity extraction, coreference resolution, relation extraction, and knowledge graph construction.

Automated analysis of threat intelligence is faced with the following challenges: (1) Unlike the entities in the general domain, entities in the threat intelligence domain have strong domain characteristics. For instance, threat entities, including hacker organizations, attack techniques, malware, etc., are difficult to identify directly by an entity extraction model in the general domain; (2) in a threat intelligence document, an entity may appear multiple times, i.e., there are multiple mentions. Determining whether the mentions refer to the same entity requires fully utilizing contextual information and extracting semantic knowledge. (3) Threat intelligence documents have complex structures and relatively long sentences. The relationships between entities are usually inferred from multiple sentences. (4) There is a lack of publicly available threat intelligence datasets.

To overcome the above-mentioned challenges, this paper proposes a novel threat intelligence information extraction system that combines multiple models and involves four steps: entity extraction, coreference resolution, relation extraction, and knowledge graph construction. Zhou et al. [4] also proposed an extraction system for APT threat intelligence, but they could only extract related entities. Vulcan [5] extracted descriptive or static CTI data from unstructured text and determined their semantic relationships. However, their definitions of entities and relationships in threat intelligence are not comprehensive. The main contributions of this paper are summarized as follows:

(1) A hybrid model is proposed to implement information extraction in the threat intelligence domain. The model can output the input unstructured threat intelligence text in a structured manner and generate a knowledge graph. The knowledge graph is stored using the Neo4j graph database to explicitly present the entities in threat intelligence and the relationships between them, thus providing knowledge support and decision support for security analysts to understand attack events and make defense deployments.

(2) A novel entity extraction model called entity extraction with multihead attention and POS (EEMAP) is proposed. The model obtains vector representations that are important to entities through a multiheads self-attention mechanism. Then, it fuses the vector representations with the feature vectors generated by a recurrent neural network model. Next, the fused results are input to a linear layer to obtain sequence labels, and the entities in the text are extracted.

(3) A novel coreference resolution model called Coreference Resolution with CNN and POS (CRCP) is proposed. In this model, the mention representation is augmented by fusing contextual information and mention embedding. Also, a convolutional neural network is introduced to extract different dimensions of the mentioned features, which can effectively compensate for the low recall rate of traditional coreference resolution methods.

(4) To solve the problem of lacking publicly available datasets for threat intelligence, this paper collects and annotates 227 threat intelligence documents from security vendor bulletins, blogs, and forums. The experimental results in this paper validate the effectiveness of our model.

2. Related Work

This section mainly reviews the existing work that is closely related to our study. Specifically, advanced models related to entity extraction, coreference resolution, and relation extraction are presented.

2.1. Entity Extraction. Entity extraction is a fundamental task of natural language processing, which aims to extract and categorize people, places, organizations, and entities with specific meanings from unstructured text and organize them into semi-structured or structured information. After this, other techniques can be used to analyze and understand the text. In essence, entity extraction is a sequence labeling task.

The early research [6] on entity extraction mainly used rule-based or dictionary-based methods, and experts built many rule templates and adopted string matching as the main approach for entity extraction. This approach can achieve satisfactory results on fixed pattern datasets, but it has poor generalization ability and cannot adapt to changing data.

Statistical machine learning can obtain better results for the cybersecurity entity extraction task. This method does not require the manual definition of rule templates and has good portability. Statistical machine learning models include the hidden Markov model (HMM), maximum entropy model (MEM), and conditional random field (CRF) model. Joshi et al. [7] proposed to extract entities, concepts, and relationships from cybersecurity blogs and announcements using the CRF model to construct a custom cybersecurity ontology. Mulwad et al. [8] employed a support vector machine (SVM) as a classifier to identify attack means and results. Bridges et al. [9] evaluated the MEM on different security corpora to avoid the overfitting problem when training the model. The above methods are more robust than rule-based methods, but they require manual mining of text features, so it is difficult for these methods to achieve satisfactory results on datasets of small sizes.

With the rise of deep learning in various fields, neural networks are continuously improved and have achieved excellent performance in entity extraction tasks. For example, Chiu and Nichols [10] used a hybrid model of

BiLSTM and CNN to detect character-level and token-level features. Then, they developed a new dictionary encoding model and a matching algorithm. Dionísio et al. [11] first used CNN to determine whether the collected tweets contained security information related to the property of IT infrastructure, followed by extracting named entities with BiLSTM to obtain security alerts. Wu et al. [12] investigated new attack patterns using a dependency analysis model to extract tactics, technology, and entities in e-commerce threat intelligence. Gasmi et al. [13] employed a BiLSTM-CRF model to extract cybersecurity concepts and entities, and they compared the model with three LSTM-based models. Zhao et al. [14] proposed an IOC identification method based on the multigranular attention mechanism and constructed a heterogeneous information network to model the dependencies among IOCs to improve accuracy.

2.2. Coreference Resolution. The coreference resolution identifies whether the mention pairs belong to the same entity. In document-level relation extraction tasks, an entity may appear many times, i.e., there are multiple mentions. If two references refer to the same entity, they are said to be "coreferential." For example, in the sentence, "The malware also silently downloads and installs a known malicious app named ister59.apk (detected as Android.Reputation.3) from the following URL." ister59.apk and Android.Reputation.3 refer to the same malware. Coreference resolution is widely used in various tasks such as relation extraction [15], event extraction [16, 17], multiparty conversation [18], and abstract meaning generation [19].

2.3. Relation Extraction. Relation extraction aims to extract relation facts from the text. Early studies [20–23] focused on predicting the relationship between two entities in a single sentence. However, an increasing number of relationship facts need to be extracted through multiple sentences, that is, to perform document-level relation extraction.

Document-level relation extraction approaches are mainly divided into two categories: graph-based methods and transformer-based methods. Specifically, graph-based methods construct document graphs based on documents to intuitively model entity structures. Zeng et al. [24] constructed mention-level and entity-level document graphs, respectively, and they proposed a new path inference mechanism to infer relationships between entities. Sun et al. [25] presented a dual-channel hierarchical graph convolutional neural network called DHGCN to model token-level, mention-level, and entity-level complex interactions between different semantics in a document. Transformer-based methods employ pretrained models (Bert, Roberta, ERNIE, etc.) to obtain the representation of each word in a document. Yuan et al. [26] exploited an intersentence attention mechanism to dynamically obtain multiple key sentence features, and they designed a gating function to combine sentence-level features with document-level features. Zhang et al. [27] introduced a U-shape segmentation module to capture local and global information by predicting entity-level relationship matrices. To solve the multientity and multilabel problems in document-level relation extraction, Zhou et al. [28] proposed to enhance entity embedding by introducing local context pooling and reducing the optimization overhead by replacing global thresholding with adaptive thresholding.

3. Methodology

3.1. Model Framework. This paper proposes a threat intelligence information extraction system combining multiple models. The proposed system consists of named entity recognition, coreference resolution, document-level relation extraction, and knowledge graph construction. The structure of the system is shown in Figure 1. Firstly, the input is converted into POS-enhanced embeddings using a pretrained model BERT and a Python library NLTK. Then the embeddings are fed into the named entity recognition model, coreference resolution model and document-level relation extraction model, successively. After that, the outputs are arranged into triples and inserted into the knowledge graph.

3.2. Encoding Layer. Different from the traditional encoding layer using random word embedding, in this paper, BERT is introduced to provide rich semantic knowledge, and part-of-speech embedding is integrated to further improve the representation ability of mention embedding.

The pretrained model BERT is used as the encoder, and the special tags of "[CLS]" and "[SEP]" are added at the beginning of the document. For each mention in the document, the special tag " $*$ " is inserted at the beginning and end.

As shown in Figure 1, the given document is first input into the tokenizer to obtain tokenized document $D = [x_t]_{t=1}^{l}$, where x_t represents the word at position t. In this section, the BERT-base is used to encode the document and obtain the contextual representation H:

$$H = \mathrm{BERT}([x_1, \ldots, x_l]) = [h_1, \ldots, h_l], \tag{1}$$

where $H \in R^{l \times d_1}$, and d_1 is the dimension of hidden size.

The POS sequence of the document is obtained by NLTK. Based on this, the POS embedding matrix P is constructed as follows:

$$P = \mathrm{Pos}([x_1, \ldots, x_l]) = [p_1, \ldots, p_l], \tag{2}$$

where $P \in R^{l \times d_2}$, and d_2 is the dimension of POS embedding.

For each token, the POS-enhanced word representation is generated by fusing the contextual embedding and the POS embedding:

$$C = [h_1 \circ p_1, \ldots, h_l \circ p_l] = [c_1, \ldots, c_l], \tag{3}$$

where $C \in R^{l \times (d_1 + d_2)}$, and $\circ$ indicates the linking operation.

3.3. Entity Extraction. To obtain vector representations that are important for entities, our entity extraction model integrates a multihead self-attention mechanism. This mechanism can learn dependencies between any two words

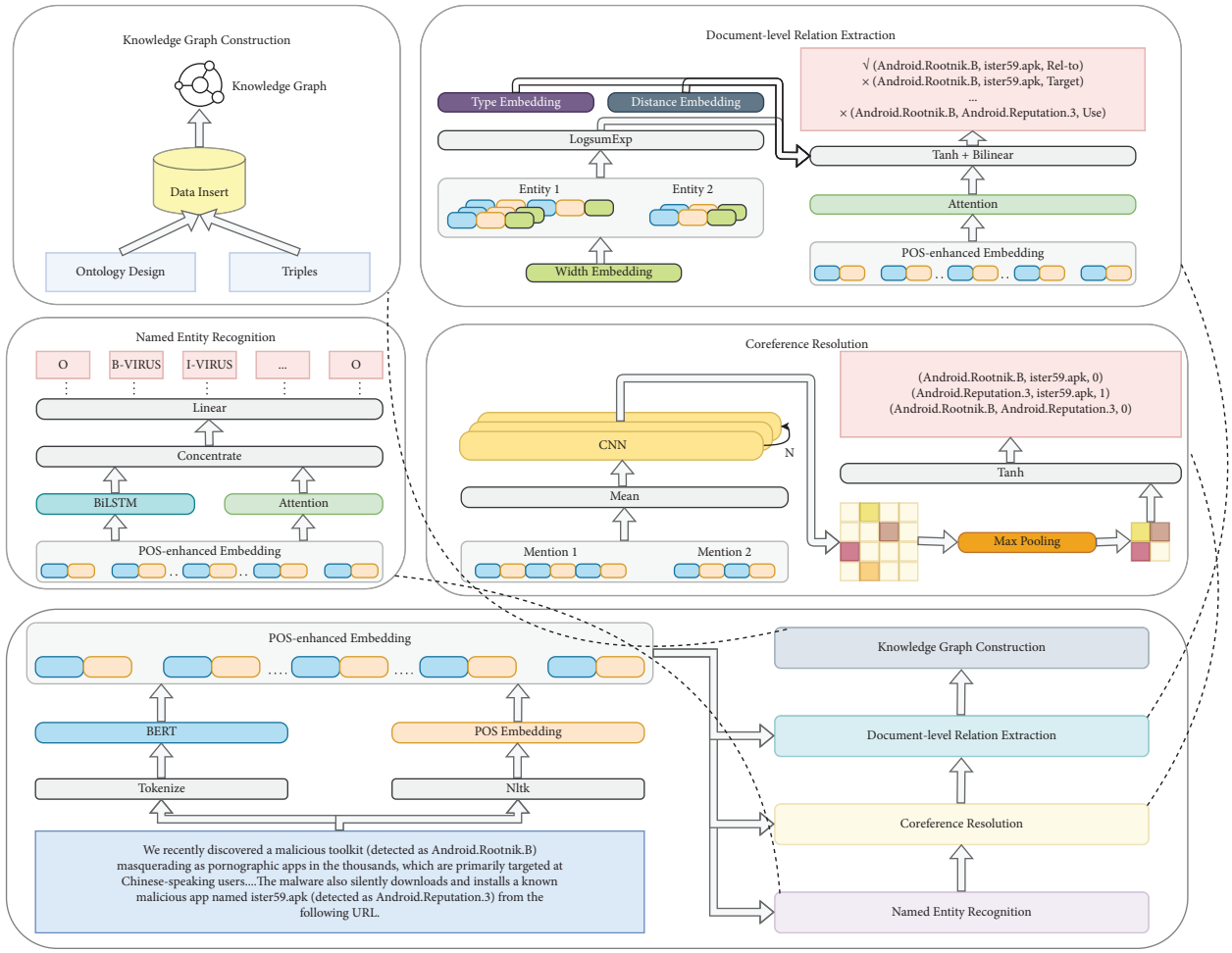


Figure 1: The proposed threat intelligence information extraction system.

and assign different weights to each token representation to obtain key information. Multiple attention heads can be used to learn features in different representation subspaces, thus significantly improving the model performance. Specifically, in this paper, a sequence of POS-enhanced token representations is used as input to the attention layer to obtain contextually significant embeddings of the current word:

$$\begin{aligned} \text{head}_i &= \text{softmax}\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right) \cdot V, \\ A &= \text{MultiHead}_i Q, K, V \\ &= \text{Concat}\left(\text{head}_1, \ldots, \text{head}_{H_1}\right) W_A, \end{aligned} \tag{4}$$

where Q, K, and V are the query, key, and value sequence, respectively; d_k is the dimension of the key sequence; and H is the number of attention heads.

This paper introduces BiLSTM to obtain historical and future information about the current word. The previous work [29] demonstrated the effectiveness of BiLSTM in capturing contextual semantic information. BiLSTM consists of a forward LSTM layer, a backward LSTM layer, and a connectivity layer. Each LSTM contains a set of cyclically connected subnetworks, called storage modules. Each time step is an LSTM storage module that is obtained based on the previous moment hidden vector, the previous moment storage cell vector, and the current input word embedding operation.

Feature vectors are obtained by inputting the POS-enhanced token representation sequence into BiLSTM.

$$B = \text{BiLSTM}[c_1, \ldots, c_1] = \left[\overrightarrow{b}_1 \circ \overleftarrow{b}_1, \ldots, \overrightarrow{b}_1 \circ \overleftarrow{b}_l\right]. \tag{5}$$

Then, the important contextual embeddings are fused with feature vectors, and the fusion results are input into the linear classifier to obtain sequence labels:

$$y_{ner} = \arg\max\left[W_1 (A \circ B) + b_1'\right]. \tag{6}$$

3.4. Coreference Resolution. Coreference resolution identifies whether two mentions refer to the same entity. This paper proposes a model that treats the coreference resolution task as a binary classification problem. The model obtains POS-enhanced token representations of mentions, followed by calculating the average vector of each token:

$$\begin{aligned}\text{mention}_1 &= \text{Mean}\left(c_{1_B}, \ldots, c_{1_I}\right),\\ \text{mention}_2 &= \text{Mean}\left(c_{2_B}, \ldots, c_{2_I}\right).\end{aligned} \tag{7}$$

CNN extracts depth features through a sliding window to alleviate the problem of long-distance dependence. Usually, a convolution layer contains one filter that performs convolution operations with word vectors through a convolutional kernel. The mentioned representations are fed into the convolution layer to obtain features in different dimensions, followed by the downscaling and compression operations using a pooling layer to remove redundant information and prevent overfitting. This paper introduces max-pooling, i.e., the maximum feature value is selected from the feature values obtained by each filter in the convolution layer, and the rest of the features are discarded.

$$\begin{aligned}\text{Mention} - \text{Pair}_i &= \text{Conv}_i\left(\text{mention}_1 \cdot \text{mention}_2\right),\\ M &= \text{Concat}\left(\ldots, \ldots, \text{Mention} - \text{Pair}_N\right)W_M,\\ \text{MP} &= \text{MaxPooling}\left(M\right).\end{aligned} \tag{8}$$

After obtaining the pooled feature vector of mention pairs, the tanh activation function is further used to calculate the label probability, i.e., whether two mentions refer to the same entity.

$$y_{\text{CR}} = \tanh\left(W_2 \cdot \text{MP} + b_2'\right). \tag{9}$$

Then, the sequence labels obtained from the entity extraction model introduced in Section is used to extract the corresponding mentions. Then, they are input into the coreference resolution model to predict whether the mentions point to the same entity.

3.5. Document-Level Relation Extraction. Document-level relation extraction determines whether there are corresponding relationships between entities. In our model, it is treated as a multilabel classification problem. Additional features are fused in the entity representation to fully utilize the document information.

Specifically, the POS-enhanced token representation of " $*$ " in front of the mention is used as the representation. Experiments have shown that the width of a mention is an important piece of information about an entity. Thus, a width embedding matrix is trained and fused with the mention representation to generate a width-enhanced representation:

$$\begin{aligned}W &= \text{Width}\left(l_i\right) = \left[w_i, \ldots, w_{n_1}\right],\\ \text{mention}_{m_j} &= c_{m_j} \circ w_{m_j},\end{aligned} \tag{10}$$

where $w_i \in R^{d_3}$, d_3 is the dimension of width embedding, and m_j is the j^{th} mention of the m^{th} entity.

For entity e_i with N_{e_i} mentions $\left\{m_j^i\right\}_{j=1}^{N_{e_i}}$, it is necessary to integrate the mention-level representation to obtain the entity-level representation. Traditional methods usually adopt the maximum pooling method that performs well when the mention pair can express the relation clearly. However, in practical scenarios, the relation between the mention pair of different entities is fuzzy. This paper adopts a smoothed version of maximum pooling, i.e., LogSumExp pooling, to obtain an entity-level representation:

$$h_{e_i} = \log \sum_{j=1}^{N_{e_i}} \exp\left(\text{mention}_{m_j}\right). \tag{11}$$

The multihead attention matrix $A \in R^{\text{HD}\times l\times l}$ of the encoder BERT is used. A_{ijk} denotes the attention score from token j to token k in the i^{th} head. The attention on " $*$ " before a mention is regarded as the attention score of that mention. Meanwhile, the entity-level attention score $A_m^E \in R^{\text{HD}\times l}$ is obtained by averaging all mention attentions of the same entity. Besides, the important contexts for a specific entity pair (e_s, e_o) are located by the attention matrix, and the local context embeddings are calculated as follows:

$$\begin{aligned}A^{(s,o)} &= A_s^E \cdot A_o^E,\\ q^{(s,o)} &= \sum_{i=1}^{\text{HD}} A_i^{(s,o)},\\ a^{(s,o)} &= \frac{q^{(s,o)}}{1^T q^{(s,o)}},\\ c^{(s,o)} &= \text{Ha}^{(s,o)}.\end{aligned} \tag{12}$$

Experiments have shown that the distance between entities and entity type also impacts the effect of relation extraction. Thus, in this paper, a distance embedding matrix and a type embedding matrix are introduced and merged into entity representation. Based on this, the representation encoding of a specific entity pair can be represented as follows:

$$\begin{aligned}D &= \text{Distance}\left(d\right) = \left[d_1, \ldots, d_{n_2}\right],\\ T &= \text{Type}\left(t_i\right) = \left[t_1, \ldots, t_{n_3}\right],\\ z_s^{(s,o)} &= \tanh\left[W_S h_{e_s} + W_{C_1} c^{(s,o)} + W_{D_1} D\left(d_{so}\right) + W_{T_1} T\left(e_s\right)\right],\\ z_o^{(s,o)} &= \tanh\left[W_O h_{e_o} + W_{C_2} c^{(s,o)} + W_{D_2} D\left(d_{os}\right) + W_{T_2} T\left(e_o\right)\right],\end{aligned} \tag{13}$$

where $d_i \in R^{d_4}$, $t_i \in R^{d_5}$, d_4, and d_5 are the dimensions of the distance embedding and type embedding, respectively; d_{so} indicates the distance between the first mentions of entities s and o; e_s and e_o are the type of s and o, respectively.

To reduce the computational overhead, in this paper, entity representations are divided into k semantic groups of the same size. Then, the entity representations are fused to obtain the following entity pair representation:

$$\begin{aligned}\left[z_s^1; \ldots; z_s^k\right] &= z_s,\\ \left[z_o^1; \ldots; z_o^k\right] &= z_o,\\ g^{(s,o)} &= \left(\sum_{i=1}^{k} z_s^i W_r^i z_o^i + b_r\right).\end{aligned} \tag{14}$$

Finally, the nonlinear activation function is employed to calculate the specific relationship probability.

$$y_{\mathrm{RE}} = \sigma\left(g^{(s,o)}\right) = \sigma\left(\sum_{i=1}^{k} z_s^i W_r^i z_o^i + b_r\right). \tag{15}$$

3.6. Knowledge Graph Construction. Ontology is fundamental for knowledge graph construction. In this paper, the threat intelligence ontology previously studied by our team is introduced, and it is shown in Figure 2.

The scattered and heterogeneous security data are organized by the abovementioned information extraction model. Then, a threat intelligence knowledge graph is constructed based on the ontology, displaying entities, and their relationships visually to provide data analysis results for threat modeling and attack reasoning in cybersecurity.

4. Experiment

4.1. Datasets. Due to the lack of publicly available information extraction datasets in the field of threat intelligence, this paper crawled 227 threat intelligence documents for preprocessing and manual annotation. Then, the documents were divided into a training set of 151 documents and a test set of 76 documents. The training set contains 1610 entities and 949 relationships. The distributions of entities and relationships are as shown in Figure 3.

4.2. Experiment Settings. In the experiment, the pretrained model BERT was used as the encoder to encode threat intelligence documents, and its contextual representation was obtained with a hidden layer dimension $d_1 = 768$. To achieve better performance, the training epoch was set as epoch = 120 (use the early stop strategy when the performance does not improve within 20 epochs), batch_size = 8. Our model use the AdamW optimizer with a linear warmup strategy to improve the model convergence speed. The learning rate was set to $5e-5$ for BERT and $1e-4$ for other layers. All models were trained on Nvidia Geforce RTX 3090 GPU based on Pytorch 1.7.1.

In the entity extraction task, 46 parts of speech and 13 entity tags were introduced. The dimensions of the POS embedding and feature vector of BiLSTM were set as $d_2 = 25$ and $d_h = 300$, respectively. Besides, the number of heads was set as $H_2 = 5$.

In the coreference resolution task, the convolution kernel size was set to kernel_size = 3, and the output channel was set to 200.

In the relation extraction task, the dimensions of width embedding, distance embedding, and type embedding were set to $d_3 = d_4 = d_5 = 25$. The number of semantic groups was $k = 64$, and the number of attention heads was $H_2 = 12$.

According to the previous work, precision, recall, and $F1$ score were used as evaluation metrics for model performance in the above-mentioned three tasks. Meanwhile, in the entity extraction task, exact-match accuracy (i.e., all tokens in the entity are predicted correctly) was introduced as an additional evaluation metric.

4.3. Result Analysis

4.3.1. Performance on Entity Extraction. Table 1 presents the comparison between our model EEMAP-BERT and the baselines on the entity extraction task, where EEMAP-WE indicates that the pretrained model BERT is replaced by random word embedding as the encoder. It can be seen from Table 1 that compared to baselines BiLSTM [11] and BiLSTM-CRF [13], our model obtained significantly better precision, recall, $F1$ score, and exact-match accuracy. Specifically, the $F1$ score was improved by 9.94 and 8.87, respectively. Then, ablation experiments were performed to analyze the effect of each module on the model performance.

First, the POS embedding was removed (labeled as NoPOS), and it was observed that the $F1$ score was decreased by 0.67, and the exact-match accuracy was decreased by 0.74. Particularly, according to the knowledge of threat intelligence, entities are mainly nouns and verbs, so these two types of embeddings have a significant impact on the model performance.

Then, the multihead self-attention layer was removed (labeled as attention). It was observed that the $F1$ score was decreased by 0.42, and the exact-match accuracy was decreased by 0.56. The experiment results indicate that the multihead self-attention mechanism can help to locate vital context and capture long-distance interdependent features.

Finally, both the POS embedding and multihead self-attention layer were removed (labeled as No POS + No Attention). It can be seen that the performance was significantly degraded, where the $F1$ score was decreased by 2.76, and the exact-match accuracy was decreased by 3.45.

Additionally, to explore the influence of the model encoder on performance, the performance of random word embedding and BERT were compared. It was indicated that the model performance was significantly reduced when random word embedding was used. The $F1$ score was reduced by 9.65, and the exact-match accuracy was reduced by 13.01. The experimental results showed that the pretrained model trained on a large corpus could learn universal language representations, which contributes to better generalization performance and faster convergence speed on the target task.

It can be seen from Figure 4 that when the number of attention heads is 5, both the $F1$ score and exact-match accuracy reached the highest point, so $H_2 = 5$ was set.

To further investigate the ability of the POS embedding and attention mechanism to fit different data types, Figure 5 shows the single-category fine-grained performance in detail.

4.3.2. Performance on Coreference Resolution. Table 2 presents the performance of our model on the coreference resolution task, where CRCP-WE indicates that the pretrained model BERT is replaced by random word embedding as the encoder. As shown in Table 2, compared to

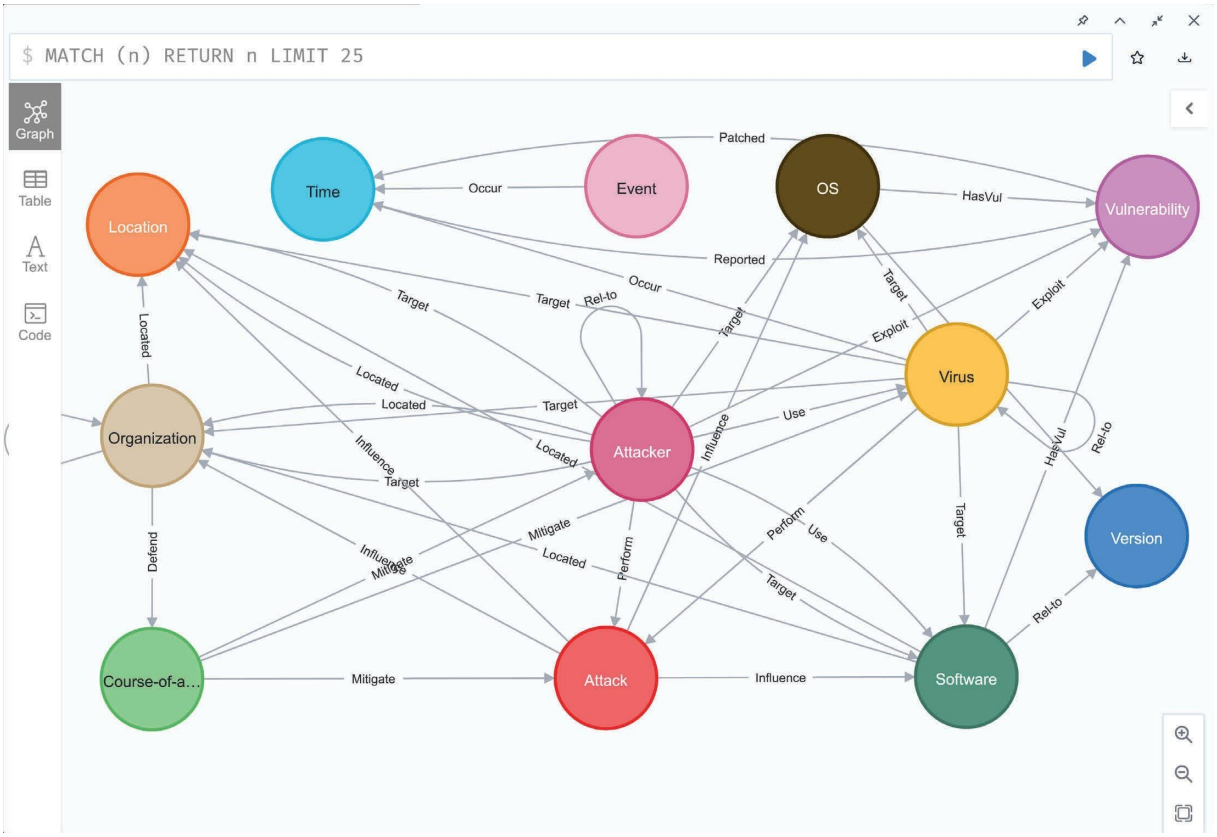


Figure 2: Threat intelligence ontology.

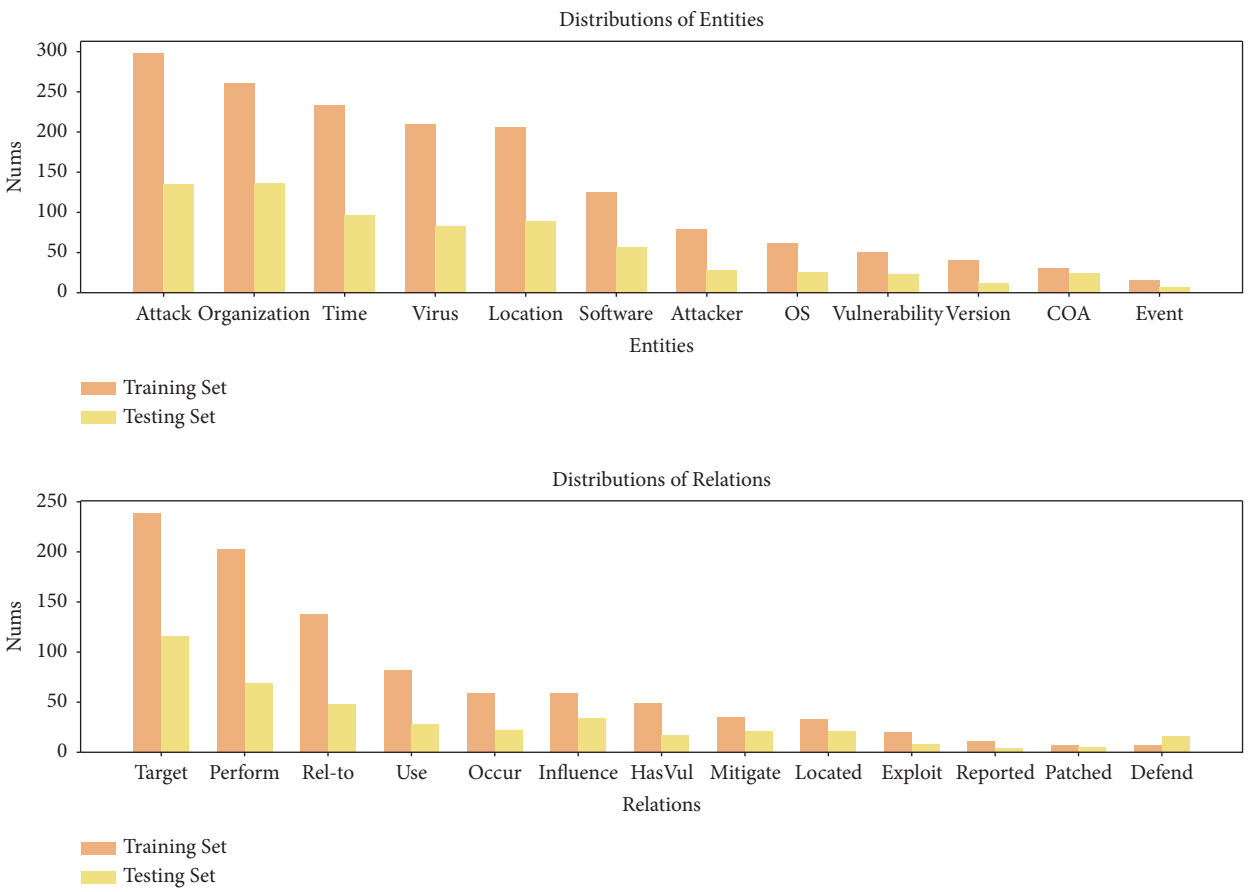


Figure 3: The distribution of entities and relationships.

Table 1: The performance of the entity extraction task.

Model	Precision	Recall	*F*1	Accuracy
EEMAP-WE	69.23	67.71	68.46	61.12
BiLSTM [11]	69.80	66.62	68.17	62.11
BiLSTM-CRF [13]	70.10	68.40	69.24	62.77
EEMAP-BERT (our model)	**79.02**	**77.22**	**78.11**	**74.13**
No POS	78.43	76.46	77.44	73.39
No attention	78.56	76.84	77.69	73.57
No POS + no attention	75.97	74.74	75.35	70.68

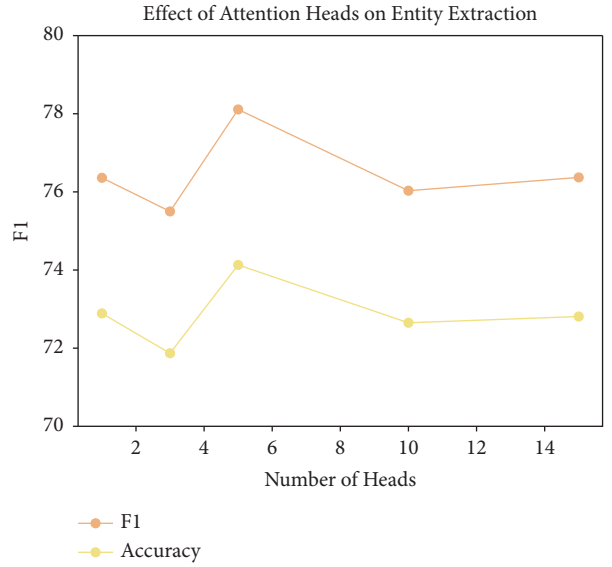
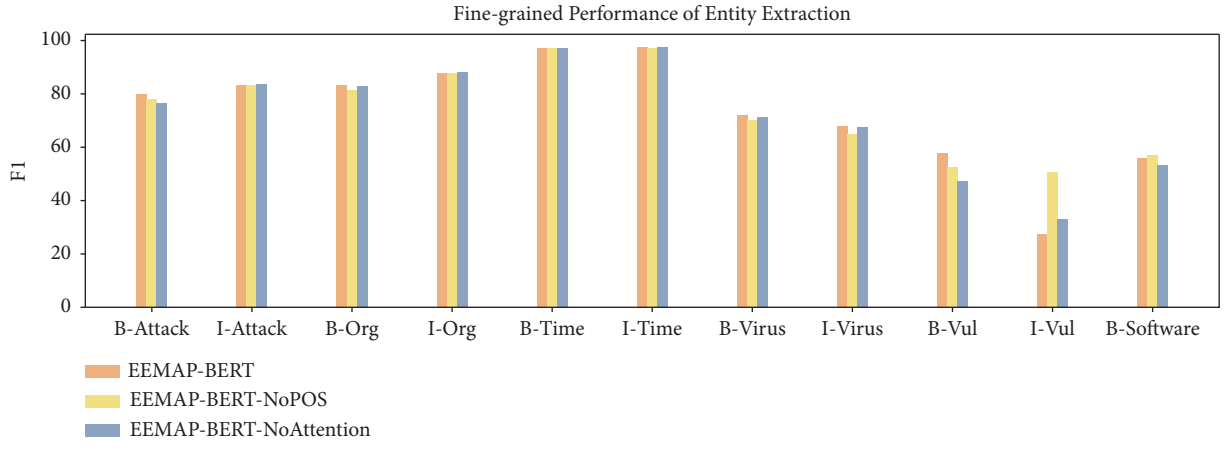
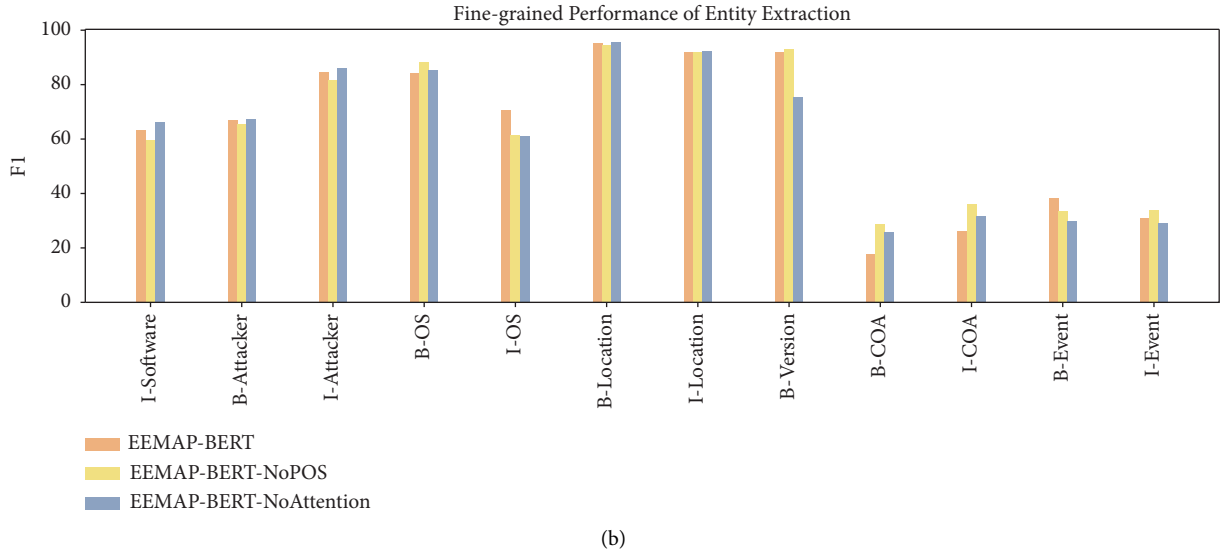


Figure 4: The effect of attention heads on entity extraction.



(a)

Figure 5: Continued.



(b)

Figure 5: Single-category fine-grained performance of entity extraction.

Table 2: The performance on the coreference resolution task.

Model	Precision	Recall	*F*1
E2E-CR [30]	62.87	55.46	58.93
CRCP-WE	**72.70**	62.74	67.35
CRCP-BERT (our model)	65.58	**72.25**	**68.75**
No POS	65.59	69.97	67.71
No CNN	63.35	68.81	65.97
No POS + No CNN	64.67	64.63	64.65

Table 3: The performance of the relation extraction task.

Model	Precision	Recall	*F*1
GAIN [24]	50.43	46.92	48.61
ATLOP [28]	51.09	48.55	49.79
DRE (our model)	**79.61**	48.59	**60.35**
No POS	67.48	44.58	53.69
No width	62.81	45.78	52.96
No type	68.31	44.71	54.05
No distance	62.24	**48.99**	54.83

E2E-CR, our model CRCP-BERT improved the *F*1 score by 9.82 [30]. Similar to the experiment on the entity extraction task, ablation experiments were performed on each module.

Firstly, POS embedding was removed (labeled as NoPOS) to investigate its effect on the performance of coreference resolution. It can be seen that the *F*1 score decreasing by 1.04 after the POS embedding was removed. If two entities are mentioned at different parts of the speech, they must not refer to the same entity. Therefore, POS embedding can be used as an auxiliary feature to obtain potential semantic information.

Then, the CNN module was removed (labeled as NoCNN), and it can be seen that the model performance was significantly degraded, with the *F*1 score decreased by 2.78. This is mainly because CNN can extract information from different dimensions and effectively identify the corresponding features.

Finally, both the POS embedding and CNN were removed (labeled as "No POS + No CNN"), the model performance was further degraded, and the *F*1 score was reduced by 4.15.

Additionally, it can be observed that when random word embedding was used as the model encoder, the model achieved the highest precision, but the recall and the *F*1 score decreased by 9.51 and 1.40, respectively. The analysis of the results indicated that after using random word embedding, the introduced part-of-speech embedding and CNN module might cause the model to overfit, resulting in a decrease in the overall performance.

4.3.3. Performance on Relation Extraction. To tackle the imbalance of the dataset, we adopt random over sampling to copy minority classes before training our model. Specifically, tokens are replaced by their synonyms to create new samples.

Table 3 presents the performance of our model (labeled as DRE) on the relation extraction task. Compared to GAIN [24] and ATLOP [28], our model improved the *F*1 score by 11.74 and 10.56, respectively. Then, ablation experiments were conducted to verify the effects of four features, namely parts-of-speech, mention width, entity type, and distance of entity pairs, on the model's performance. The experimental results indicated that the performance of the model decreased significantly after the four modules were removed, which validates the good effect of introducing features on the model performance.

The influence of each feature module on the *F*1 score was investigated further, as shown in Figure 6.

4.3.4. Knowledge Graph Construction. The threat intelligence text was input into the above-trained model, and the corresponding relation triplets were generated through the processes of entity extraction, coreference resolution, and relation extraction. Then, the neo4j-admin command was executed to insert triples into the Neo4j graph database to obtain the threat intelligence knowledge graph. Some partial results are illustrated in Figure 7.

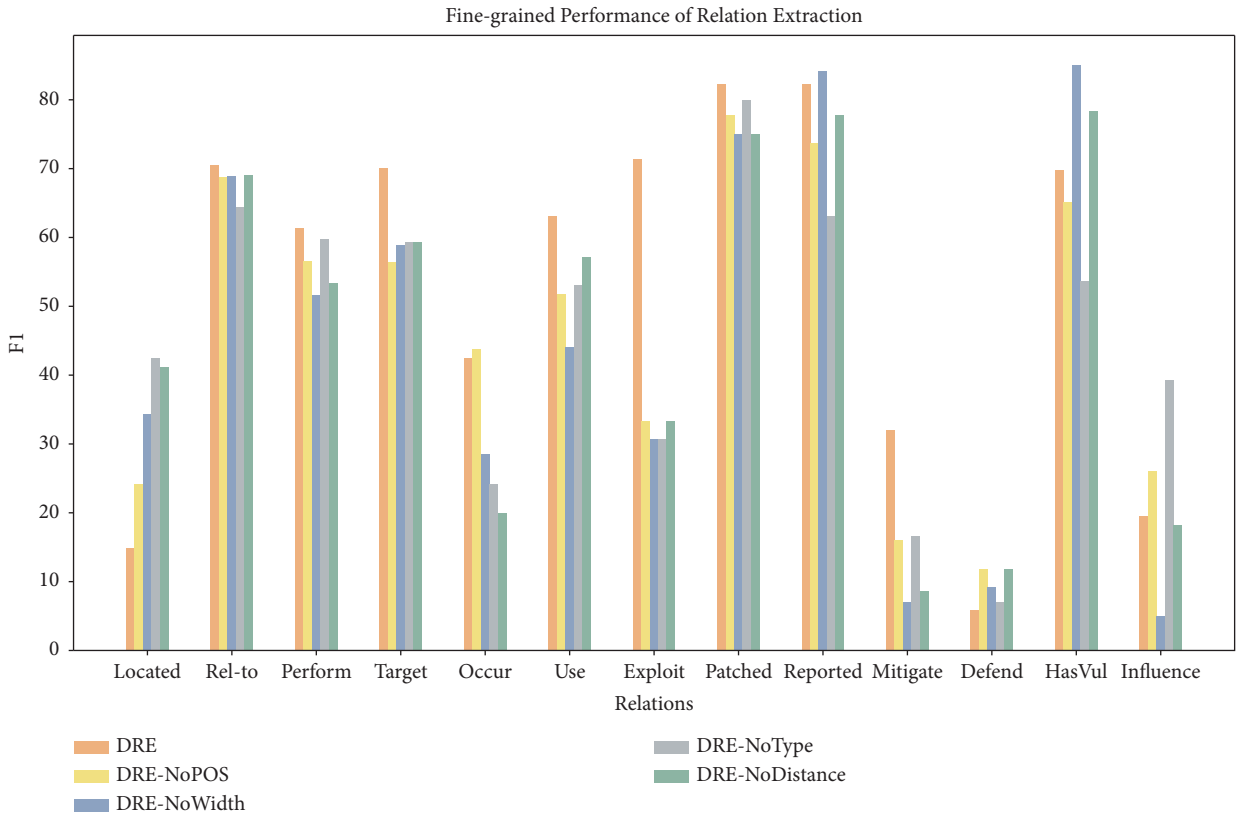


Figure 6: Single-category fine-grained performance of relation extraction.

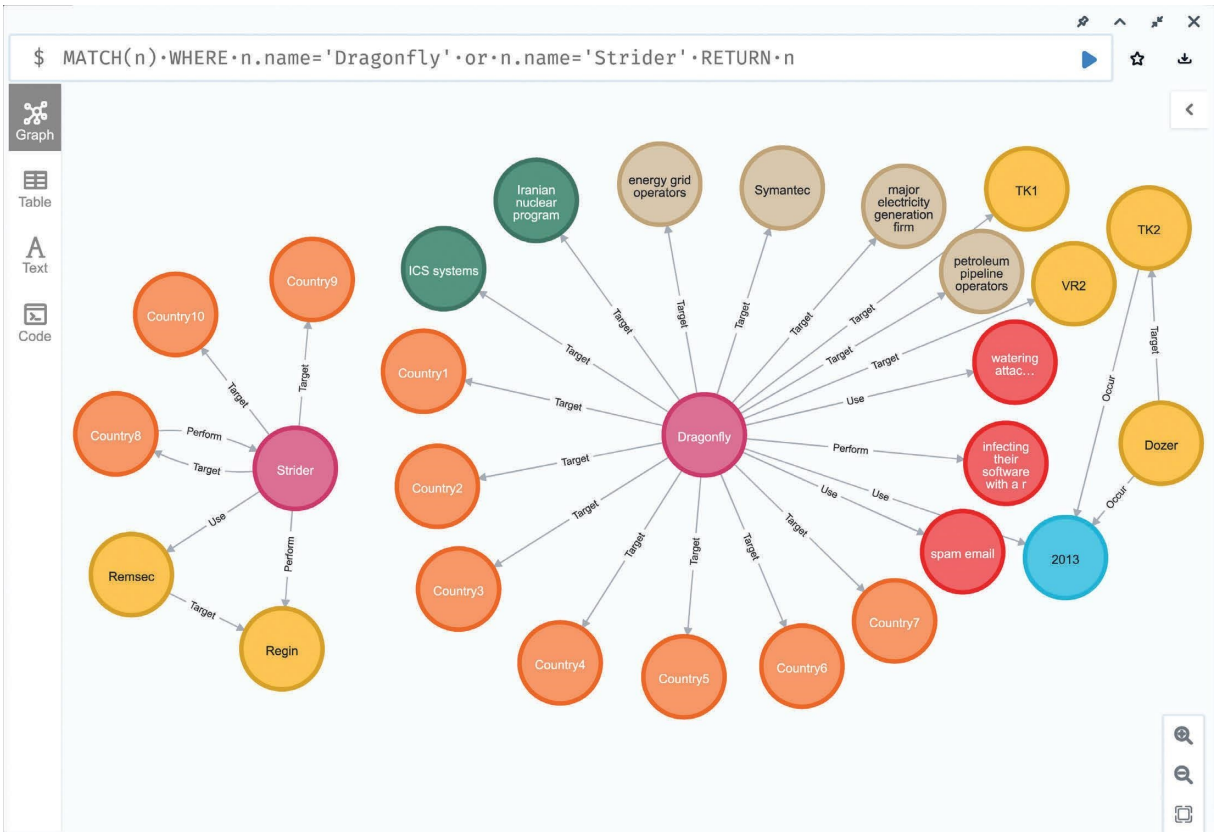


Figure 7: A part of our threat intelligence knowledge graph.

5. Conclusion

This paper designs a hybrid model to implement information extraction in the field of threat intelligence, including entity extraction, coreference resolution, relation extraction, and knowledge graph construction. First, the multihead self-attention mechanism is introduced into the entity extraction model to obtain important contextual vectors. Meanwhile, in the coreference resolution model, the contextual information is fused with mention embedding. Furthermore, the features of multiple dimensions are acquired by a convolutional neural network. In the relation extraction model, parts of speech, mention width, entity type, and the distance between entity pairs are introduced as additional features to improve the representation. Experimental results indicate that, compared to baselines, our model improves the $F1$ score by at least 8.87, 9.82, and 10.56 on the tasks of entity extraction, coreference resolution, and relation extraction, respectively. Finally, a threat intelligence knowledge graph is constructed to illustrate the potential semantic connections between entities. In summary, our model can automatically extract knowledge from multiple documents and display the internal relationships between crucial elements. It lays a firm foundation for situational awareness and attack detection. In future work, we will further refine entity and relationship classification and collect more examples to extend our dataset. Also, we will perform knowledge reasoning through a knowledge graph to obtain more knowledge.

Data Availability

The experiment dataset used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (grant nos. 61501515 and 61601515).

References

[1] R. Kumar, P. Kumar, R. Tripathi, G. P. Gupta, S. Garg, and M. M. Hassan, "A distributed intrusion detection system to detect DDoS attacks in blockchain-enabled IoT network," *Journal of Parallel and Distributed Computing*, vol. 164, pp. 55–68, 2022.

[2] P. Kumar, G. P. Gupta, and R. Tripathi, "Design of anomaly-based intrusion detection system using fog computing for IoT network," *Automatic Control and Computer Sciences*, vol. 55, no. 2, pp. 137–147, 2021.

[3] P. Kumar, R. Tripathi, and P. G. Gupta, "P2IDF: a privacy-preserving based intrusion detection framework for software defined Internet of Things-fog (SDIoT-Fog)," in *Proceedings of the 2021 International Conference on Distributed Computing and Networking*, pp. 37–42, Nara Japan, January 2021.

[4] Y. Zhou, Y. Tang, M. Yi, C. Xi, and H. Lu, "CTI view: APT threat intelligence analysis system," *Security and Communication Networks*, vol. 202215 pages, Article ID 9875199, 2022.

[5] H. Jo, Y. Lee, and S. Shin, "Vulcan: automatic extraction and analysis of cyber threat intelligence from unstructured text," *Computers & Security*, vol. 120, Article ID 102763, 2022.

[6] X. Liu and J. Li, "Key-based method for extracting entities from XML data," *Journal of Computer Research and Development*, vol. 51, no. 1, pp. 64–75, 2014.

[7] A. Joshi, R. Lal, T. Finin, and A. Joshi, "Extracting cybersecurity related linked data from text," in *Proceedings of the 2013 IEEE Seventh International Conference on Semantic Computing*, pp. 252–259, IEEE, Irvine, CA, USA, September 2013.

[8] V. Mulwad, W. Li, A. Joshi, T. Finin, and K. Viswanathan, "Extracting information about security vulnerabilities from web text," in *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, pp. 257–260, IEEE, Lyon, France, August 2011.

[9] R. A. Bridges, K. M. Huffer, C. L. Jones, M. D. Iannacone, and J. R. Goodall, "Cybersecurity automated information extraction techniques: drawbacks of current methods, and enhanced extractors," in *Proceedings of the 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 437–442, IEEE, Cancun, Mexico, December 2017.

[10] 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.

[11] N. Dionísio, F. Alves, P. M. Ferreira, and A. Bessani, "Cyberthreat detection from twitter using deep neural networks," in *Proceedings of the 2019 international joint conference on neural networks (IJCNN)*, pp. 1–8, IEEE, Budapest, Hungary, July 2019.

[12] Y. Wu, Q. Liu, X. Liao et al., "Price tag: towards semi-automatically discovery tactics, techniques and procedures of E-commerce cyber threat intelligence," *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2021.

[13] H. Gasmi, J. Laval, and A. Bouras, "Information extraction of cybersecurity concepts: an lstm approach," *Applied Sciences*, vol. 9, no. 19, p. 3945, 2019.

[14] J. Zhao, Q. Yan, X. Liu, B. Li, and G. Zuo, "Cyber threat intelligence modeling based on heterogeneous graph convolutional network," in *Proceedings of the 23rd international symposium on research in attacks, intrusions and defenses (RAID 2020)*, pp. 241–256, San Sebastian, Spain, October 2020.

[15] D. Ye, Y. Lin, J. Du et al., "Coreferential reasoning learning for language representation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7170–7186, November 2020.

[16] Y. Lu, H. Lin, J. Tang, X. Han, and L. Sun, "End-to-end neural event coreference resolution," *Artificial Intelligence*, vol. 303, Article ID 103632, 2022.

[17] H. M. Tran, D. Phung, and T. H. Nguyen, "Exploiting document structures and cluster consistencies for event coreference resolution," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 4840–4850, Bangkok, Thailand, August 2021.

[18] J. Bai, H. Zhang, Y. Song, and K. Xu, "Joint coreference resolution and character linking for multiparty conversation," in *Proceedings of the 16th Conference of the European Chapter*

of the Association for Computational Linguistics, pp. 539–548, April 2021.

[19] Q. Fu, L. Song, W. Du, and Y. Zhang, "End-to-end AMR coreference resolution," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 4204–4214, Bangkok, Thailand, August 2021.

[20] Y.-M. Shang, H. Huang, X. Sun, W. Wei, and X.-L. Mao, "A pattern-aware self-attention network for distant supervised relation extraction," *Information Sciences*, vol. 584, pp. 269–279, 2022.

[21] Z. Guo, G. Nan, W. Lu, and S. B. Cohen, "Learning latent forests for medical relation extraction," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 3651–3657, Yokohama, Japan, January 2021.

[22] J. Liu, S. Chen, B. Wang, J. Zhang, N. Li, and T. Xu, "Attention as relation: learning supervised multi-head self-attention for relation extraction," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 3787–3793, Yokohama, Japan, January 2021.

[23] S. Zhao, M. Hu, Z. Cai, and F. Liu, "Modeling dense cross-modal interactions for joint entity-relation extraction," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 4032–4038, Yokohama, Japan, January 2021.

[24] S. Zeng, R. Xu, B. Chang, and L. Li, "Double graph based reasoning for document-level relation extraction," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1630–1640, November 2020.

[25] Q. Sun, T. Xu, K. Zhang et al., "Dual-Channel and hierarchical graph convolutional networks for document-level relation extraction," *Expert Systems with Applications*, vol. 205, Article ID 117678, 2022.

[26] C. Yuan, H. Huang, C. Feng, G. Shi, and X. Wei, "Document-level relation extraction with entity-selection attention," *Information Sciences*, vol. 568, pp. 163–174, 2021.

[27] N. Zhang, X. Chen, X. Xie et al., "Document-level relation extraction as semantic segmentation," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, Montreal, Canada, August 2021.

[28] W. Zhou, K. Huang, T. Ma, and J. Huang, "Document-level relation extraction with adaptive thresholding and localized context pooling," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 14612–14620, Vancouver, Canada, February 2021.

[29] P. Zhou, W. Shi, J. Tian et al., "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 207–212, Berlin, Germany, August 2016.

[30] K. Lee, L. He, M. Lewis, and L. Zettlemoyer, "End-to-end neural coreference resolution," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 188–197, Copenhagen, Denmark, September, 2017.