

Article

Multi-Relation Extraction for Cybersecurity Based on Ontology Rule-Enhanced Prompt Learning

Fei Wang ^{1,2}, Zhaoyun Ding ^{1,*}, Kai Liu ¹ , Lehai Xin ¹, Yu Zhao ¹ and Yun Zhou ¹ 

¹ National Key Laboratory of Information Systems Engineering, National University of Defense Technology, Changsha 410073, China; wangfei@nudt.edu.cn (F.W.); liukai18@nudt.edu.cn (K.L.); xinlehai@nudt.edu.cn (L.X.); yuzhao@nudt.edu.cn (Y.Z.); zhouyun@nudt.edu.cn (Y.Z.)

² Qingdao Branch of Naval Aeronautics University, Qingdao 266041, China

* Correspondence: zyding@nudt.edu.cn

Abstract: In the domain of cybersecurity, available annotated data are often scarce, especially for Chinese cybersecurity datasets, often necessitating the manual construction of datasets. The scarcity of samples is one of the challenges in researching cybersecurity, especially for the “no-relation” class. Since the annotation process typically focuses only on known relation classes, there are usually no training samples for the “no-relation” class. This poses a zero-shot classification problem, where during the classification process, there is a tendency to classify into a class with a relationship. Zero-shot classification tasks are particularly challenging in this context. Moreover, most relation classification models currently need to traverse all relations to calculate the class with the highest probability. Therefore, the problem of “computational redundancy” is another challenge faced. Thus, how to accurately and efficiently acquire cyberspace knowledge from heterogeneous data sources and address the challenges such as sample scarcity, zero-shot recognition, and computational redundancy is the main focus of this chapter. To address these problems, this chapter designs a multi-relation extraction model based on ontology rule-enhanced prompt learning, which is a parameter-sharing-based multi-task model. By introducing prompt learning, which has shown significant effectiveness in the few-shot domain, this chapter designs prompt templates combining discrete and continuous tokens and uses rule injection in prompt learning to solve the difficulties in zero-shot recognition of “no-relation” and computational redundancy issues, achieving efficient and accurate multi-relation extraction. Specifically, by constructing sub-prompts to achieve an efficient combination of templates, a parameter-sharing structure is used to implement knowledge extraction step by step: The first step constructs entity prompt templates combining discrete and continuous tokens, identifying the classes of two entities based on prompt learning. The second step involves rule injection, identifying whether it belongs to the “no-relation” class based on the combination of sub-prompts; if there is no connection between the classes of two entities, it is classified as “no relation”; if a connection exists, the candidate relation set is filtered out. The third step uses the pre-trained model and vectors from the first step, utilizing prompt learning and rule judgment to determine the relation class from the candidate relation set. Finally, the effectiveness of our model is validated on the general datasets TACRED, ReTACRED, and the cybersecurity dataset constructed in this paper.

Keywords: ontology rule; prompt learning; cybersecurity; multi-relation extraction



Citation: Wang, F.; Ding, Z.; Liu, K.; Xin, L.; Zhao, Y.; Zhou, Y. Multi-Relation Extraction for Cybersecurity Based on Ontology Rule-Enhanced Prompt Learning. *Electronics* **2024**, *13*, 2379. <https://doi.org/10.3390/electronics13122379>

Academic Editor: Aryya Gangopadhyay

Received: 12 May 2024

Revised: 6 June 2024

Accepted: 13 June 2024

Published: 18 June 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The internet contains a vast amount of valuable security information, such as attack events, security blogs, security intelligence, vulnerability databases, system logs, and other security data. Security data in cyberspace are massive and complex, generating countless new data every moment. However, in the field of cybersecurity, truly usable annotated data are very scarce, often requiring manual construction of training sets [1,2]. What appears to be big data are, in fact, a few shots, posing one of the challenges in researching cyberspace

security. Especially in the cybersecurity relation extraction, the annotation process typically focuses on labeling known relation classes, with very few or even no “no relation” samples, which belong to zero-shot classification. When classifying relations, there is a tendency to classify into a predefined set of relation classes, resulting in poor classification performance for “no relation” samples.

In Figure 1, there are four entities, three of which belong to the “organization” class, and one belongs to the “tool” class. The relationships between entities are judged separately, and the list of their real relations is as in Figure 2. It can be observed that

- (1) The relationships between real entity classes, including intra-group relationships, often involve multiple relations.
- (2) It is necessary to pay attention to zero-shot cases, that is, the “no relation” class. In many cases, there may be a relationship between classes, but according to real semantics, the two entities have no relation. In such cases, especially in imbalanced classifications, it is more inclined to classify into the pre-defined set of relation classes, making the classification performance of “no relation” samples poor.
- (3) The same type of relation exists between multiple classes. For example, “attacked” and “no relation” in the following example correspond to multiple classes.

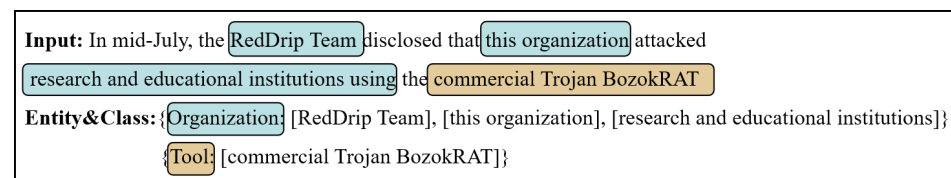


Figure 1. Cybersecurity Corpus example.

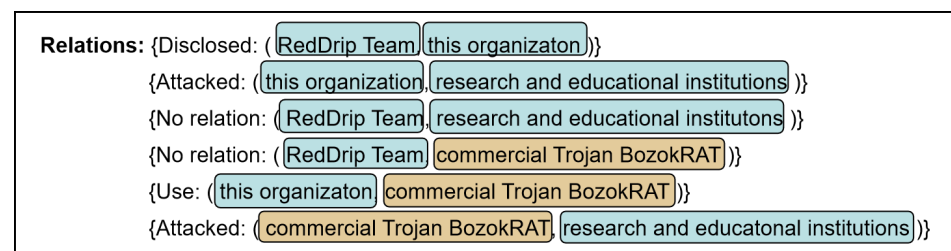


Figure 2. Relation list.

From this, it can be seen that in actual samples, there are often multiple relations between two entity classes, and these relations are complex and intertwined across multiple entity classes. Currently, most relation classification models need to traverse all relations to calculate the class with the highest probability. Therefore, the problem of “computational redundancy” is another challenge faced [3,4]. Thus, how to accurately and efficiently acquire cyberspace knowledge from heterogeneous data sources and address issues such as sparse samples, poor classification performance of “no relation” class, and computational redundancy is the main focus of this study.

To address this problem, this paper proposes a multi-relation extraction model based on ontology rule-enhanced prompt learning, as illustrated in Figure 3. To tackle the issue of small samples in the cybersecurity domain, a prompt template combining discrete and continuous tokens is designed. For zero-shot learning and computational redundancy problems, rule-injection prompt learning is constructed. Furthermore, a structure based on parameter sharing is designed for multi-relation extraction based on the pre-trained model. Specifically, this chapter achieves an efficient combination of templates by constructing sub-prompts and implements stepwise knowledge extraction using parameter-sharing structures: the first step is to identify the classes of two entities based on prompt learning. The second step involves rule injection, where, based on the rule table constructed from the combination of sub-prompts, if there is no connection between the classes of two entities,

it is directly classified as “no relation”; if there is a connection between the classes of two entities, the candidate relation set is filtered based on rules to reduce computational redundancy. The third step involves constructing a relation prompt template based on the pre-trained model and vectors from the first step and then determining the relationship type (including the “no relation” class) based on the candidate relation set.

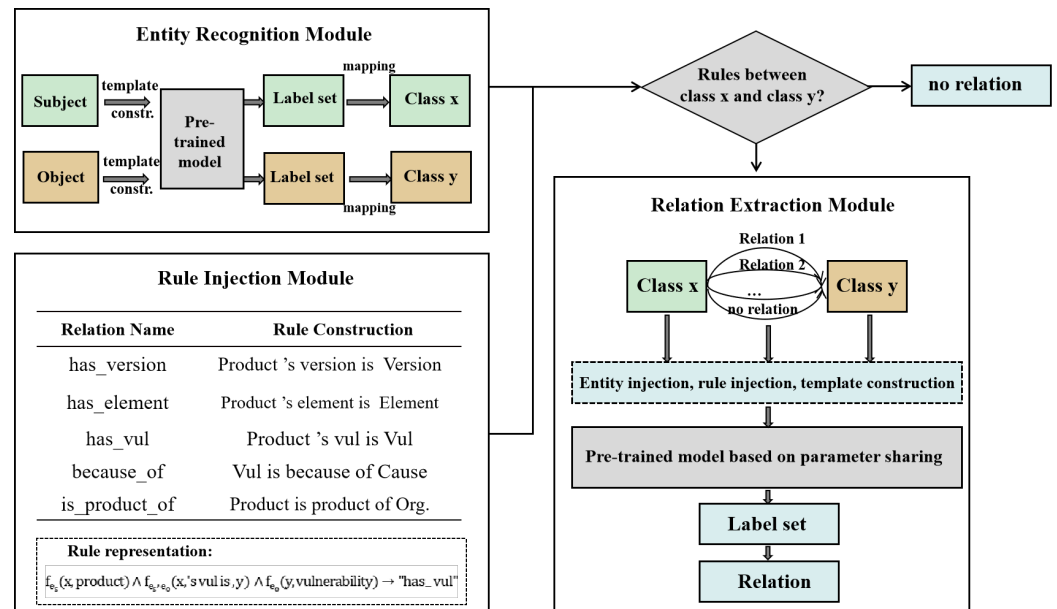


Figure 3. Cyberecurity relation extraction process based on ontology rule-enhanced prompt learning.

2. Literature Review

2.1. Development and Current Status of Information Extraction Techniques

Information Extraction (IE) is the process of extracting structured information from unstructured or semi-structured data, which is a crucial step in building knowledge graphs. The task of named entity recognition in the field of cybersecurity refers to extracting entities from complex multi-source data and determining their classes based on the classes specified by the ontology [5], with most scholars defining entity classes in the cybersecurity domain as vulnerabilities, organizations, and elements. The task of cybersecurity relation extraction involves determining the relation classes between entities from complex multi-source data based on the object properties specified by the ontology, such as has vul, uses, and exploit [6].

In the field of cybersecurity entity recognition, early approaches mostly focused on machine learning and deep learning techniques. With the advancement of pre-trained models such as BERT, GPT, and XLNet, new developments have emerged. Zhou et al. [7] combined BERT and its improved version BERT wwm with the BiLSTM-CRF architecture for named entity recognition tasks in the cybersecurity domain. Experimental results showed higher performance in precision, recall, and F1 score for both overall entities and individual entities. Ranade et al. [8] introduced CyBERT, a domain-specific BERT model based on Transformers, which was fine-tuned using a large amount of cybersecurity text data, achieving specialized entity recognition in the cybersecurity domain. Chen et al. [9] presented a joint BERT model for cybersecurity named entity recognition, where BERT served as the pre-trained model, with LSTM, Iterated Dilated Convolutional Neural Networks (ID-CNNs), and Conditional Random Fields (CRF) on the top to extract character and text features and predict sequence labels. Results showed that this joint BERT model outperformed state-of-the-art methods significantly.

Unlike entity recognition tasks, cybersecurity relation extraction relies more on semantic information between entities and sentence structure information, making it more complex and challenging. Early relation extraction often relied on manually designed

rule-based matching templates, which required significant human effort to construct a large number of templates. However, it was quickly replaced by neural network models, which are more efficient and less labor-intensive. Li Dongmei [10] utilized word embeddings, sentence embeddings, and part-of-speech features to jointly train a Support Vector Machine (SVM), analyzing the impact of different features on model performance, and pointed out that self-attention mechanisms can effectively enhance semantic features, which has become an indispensable structure for relation extraction models. To identify semantic features, some scholars used the Shortest Dependency Path (SDP) to obtain global dependency information. Gasmi et al. [11] performed entity recognition and relation extraction as two separate tasks to extract cybersecurity entities and their relationships. The NER task was completed using the CRF method for entity recognition, while relation extraction was accomplished by constructing an LSTM model and utilizing the SDP to reduce the influence of noise on semantic features. Wang et al. [12] first constructed a cybersecurity dataset, APTERC-DOC, which is a dataset containing APT intelligence entities, relations, and coreferences, and regarded relation extraction as a multi-classification task. They developed a joint learning framework named TIRECO, which can simultaneously perform threat intelligence relation extraction and coreference resolution. Pingle et al. [13] developed a threat intelligence semantic relation extraction system based on deep learning, which can extract semantic triples from open-source threat intelligence.

2.2. Development and Current Status of Prompt Learning

In recent years, the development of deep learning has brought about significant changes in natural language processing tasks. Liu et al. [14] summarized modern NLP techniques into four paradigms: Fully Supervised Learning (Non-Neural Network); Fully Supervised Learning (Neural Network); Pre-train, Fine-tune; and Pre-train, Prompt, Predict.

In the Pre-train, Fine-tune paradigm, the fine-tuning stage involves adjusting pre-trained models based on downstream tasks. Due to the differences in training objectives, there may be significant gaps between the pre-training stage and downstream tasks. To address this gap, a new paradigm called “Pre-train, Prompt, Predict” has been proposed, and currently, we are transitioning into this fourth paradigm. In this paradigm, instead of adapting pre-trained models to downstream tasks through target engineering, the gap is bridged by prompts, allowing downstream tasks to adopt the same format as the pre-training objective. The process of “Pre-train and Fine-tuning” is gradually being replaced by “Pre-train, Prompt, Predict”.

Prompt-based learning methods originated from GPT-3 (2020) [15], which achieved better results in many NLP tasks. Some studies based on manually crafted prompts, e.g., Schick et al. [16] introduced the Pattern Exploiting Training (PET) method, an effective ensemble method that combines manually crafted templates with BERT’s MLM model, achieving excellent zero-shot, few-shot, and even semi-supervised learning effects. In 2021, Ding et al. [17] enhanced prompt learning with fine-grained entity typing, constructing a fine-grained entity recognition model based on a masked language model.

To avoid extensive manual prompt construction, Gao et al. [18] first proposed automatic template and answer word generation in 2020. In 2021, Han [19] introduced PTR, which uses logical rules to construct sub-prompts. The method of injecting knowledge, learning virtual templates, and simulating answers to replace manual rule definition can generalize across multiple tasks. Injecting knowledge methods, learning virtual templates, and simulating answers to replace manually defined rules can generalize across multiple tasks. Jiang et al.’s [20] Mine approach is an automatic prompt construction method that can automatically find templates given a set of training inputs x and outputs y . Experimental results have shown that using the optimal prompt is on average better than manually crafted prompts, and integrating multiple prompts performs better than using a single prompt.

In the past two years, prompt learning has achieved excellent results in NLP tasks, especially in few-shot scenarios. Ding et al. [17] studied the application of prompt learning in fully supervised, few-shot, and zero-shot scenarios in fine-grained entity typing and

found that prompt learning methods significantly outperform fine-tuning baselines in few-shot entity recognition tasks. Cui et al. [21] proposed an NER method based on prompt learning, treating the NER task as a language model ranking problem in a sequence-to-sequence (Seq2Seq) framework, where the input consists of the original sentences and candidate named entity span, which also demonstrated the effectiveness of prompt learning in low-resource tasks.

In summary, the key techniques for information extraction in the field of cyberspace security are summarized in Table 1. Regarding entity extraction, machine learning-based methods are favored for their flexibility and good robustness, but they often require complex feature engineering and a large amount of manually labeled data, with relatively poor model portability. Deep neural networks based on pre-trained models can automatically capture features, but they rely on a large amount of labeled data, have complex model training, and have high demands on computational resources. Recently, prompt learning-based methods have gradually gained attention. They adapt to different tasks by constructing different templates without requiring a large number of samples, but the process of constructing all necessary templates is time-consuming and laborious.

Table 1. Information extraction technology.

Category	Method	Advantages	Disadvantages
Entity Extraction	Machine Learning-based	Flexibility, good robustness	Requires feature engineering, manual annotation, poor model transferability
	Pre-trained Model-based	Deep neural networks can automatically capture features	Depends on a large amount of labeled data, complex model training, high computational power demand
	Prompt-based Learning	No need for large samples, different templates can be built for various tasks	Cannot target multiple types of relations; exhaustive enumeration of all templates required
Relation Extraction	Rule-based Matching	High flexibility, high reliability	Requires manually constructing a large number of templates, time-consuming and labor-intensive
	Neural Network Model-based	Can automatically extract features	Poor scalability and portability

In terms of relation extraction, traditional rule-based methods have high flexibility and reliability but require manually constructing a large number of templates. Neural network-based methods can automatically extract features and usually require a large amount of labeled data for training, with relatively poor scalability and portability of the models. How to apply prompt learning to the field of cyberspace security is the focus and difficulty of this research paper.

3. Prompt Template Construction Strategy

One of the most important aspects of prompt learning is the construction of templates. High-quality templates can bridge the gap between pre-trained models and downstream tasks. To overcome the singularity of discrete templates and the randomness of continuous templates, this study combines the two types of templates. Specifically, discrete templates consist of three sub-templates: subject prompts, object prompts, and relation prompts, which can be flexibly combined to expand the template library, addressing the singularity of manually constructed templates while maintaining strong interpretability. On the other hand, continuous templates involve adding continuous tokens into template encoding and inserting pseudo-tokens at the beginning and end of each sub-prompt encoding. These

pseudo-tokens are learnable tokens initialized randomly through embedding, obtaining vector representations for each pseudo-token.

Recently, effective prompt fine-tuning in the domain of few-shot learning involves solving cloze-style tasks based on pre-trained language models. By providing prompts, downstream tasks can better adapt to the characteristics of pre-trained models without sacrificing the original advantages of pre-training. Formally, a prompt consists of a template T and a set of label words V . For each instance x , the template is first used to map x to the prompt input. At least one masked position is set in the template, indicated by $[MASK]$, where x is mapped. For example, a simplest template could be: $x_{prompt} = "x \text{ It was } [MASK]"$. The masked position requires a set of label words, which are eventually mapped to the actual relation class through predicted label words. To address the challenges of scarce samples and difficult multi-relation extraction in the field of cybersecurity, this paper improves the prompt templates in two aspects: flexible combinations of sub-prompts and integration of discrete and continuous prompts.

3.1. Flexible Combination of Sub-Prompts

In the example provided earlier, the prompt "It was $[MASK]$ " represents the simplest form of discrete prompt style. In previous template constructions, some would only build such a template for all corpora, which is straightforward but not flexible enough to adapt to complex corpora. Others design multiple templates based on specific tasks, which can yield better results in complex corpora, but designing different templates for different corpora can be time-consuming and laborious. Therefore, Xu et al. [19] proposed the effective method of constructing sub-prompts to address this issue, where the entire sentence prompt is broken down into multiple sub-prompts, and efficient template design is achieved through permutations and combinations of a small number of sub-prompt templates. Sub-prompts also consist of a template and a set of label words.

In this study, for the task of relation classification, prompts are split into three sub-prompts based on data characteristics, namely subject prompts, object prompts, and relation prompts. The subject and object are represented by e_s and e_o , respectively, and a function $f_{e_s}(\cdot, \{\text{product} \mid \text{entity} \mid \dots\})$ is defined, which calculates the probability of the function belonging to each class, with the class with the highest probability being the entity class. The label word set constructed in this paper for subject and object prompts are {"product", "version", "element", "vulnerability", "cause", "organization", "method", "impact", ...}. The sub-prompt template and label word set for the subject and object can be represented as

$$T_{f_{e_s}}(x) = "x \text{ the } [MASK]e_s" \quad (1)$$

$$V_{f_{e_s}}(x) = \{"product", "version", \dots\} \quad (2)$$

The twelve relation classes are defined as follows: has_version, has_element, has_vul, because_of, is_product_of, use_means_of, lead_to_consequence, exploit, develop, cooperate_with, target_at, and no_relation. They are represented by relation prompts, with the corresponding label word set {"'s version is ", "'s element is ", "'s vul is ", "is because of ", "is product of ", "used means of ", "led to ", "attacked ", "developed ", "cooperated with ", "targets at ", "is irrelevant to "}. Since relations involve subjects and objects, binary functions are commonly used for representation. Let $f_{e_s}(\cdot, \{"'s version is " \mid "'s element is " \mid "'s vul is " \mid \dots\}, \cdot)$ determine the type of entity. The sub-prompt templates and label word sets for relations can be expressed as follows:

$$T_{f_{e_s}, f_{e_o}}(x) = "x \text{ } e_s [MASK] e_o" \quad (3)$$

$$V_{f_{e_s}, f_{e_o}}(x) = \{"'s version is " , "'s vul is " , \dots\} \quad (4)$$

Therefore, the template representation of the entire sentence is as follows.

$$\begin{aligned} T(x) &= Tf_{e_s}(x); Tf_{e_s, e_o}(x); Tf_{e_o}(x) \\ &= the[MASK]_1 e_s [MASK]_2 the [MASK]_3 e_o, \end{aligned} \quad (5)$$

where the label word sets for the subject, relation, and object are, respectively,

$$V_{MASK_1} = V_{MASK_2} = \{ \text{"product", "version", "element", "vulnerability", "cause", "organization", "method", "impact", ...} \} \quad (6)$$

$$V_{MASK_3} = \{ \text{" 's version is ", " 's element is ", " 's vul is ", " is because of ", " is product of ", " used means of ", " led to ", " attacked ", " developed ", " cooperated with ", " targets at ", " is irrelevant to " } \} \quad (7)$$

The above is just one method for constructing prompt templates, using the order of "subject-relation-object" to build prompts. The order of the three sub-prompts can also be adjusted to efficiently combine into other candidate template forms. In this study, three prompt forms are constructed in total to inject entity and relationship knowledge. Prompt 1 is based on the order of "subject-relation-object", Prompt 2 is based on the order of "subject-object-relation", and Prompt 3 is based on the order of "relation-subject-object", with natural language added for connection in Prompts 2 and 3. The specific constructions are illustrated in Figure 4. Specifically, green represents the subject sub-prompt, orange represents the object sub-prompt, and blue represents the relation prompt. The three sub-templates can be freely combined and rearranged in any order.

T1 = " *the* [MASK]₁ *e_s* [MASK]₂ *the* [MASK]₃ *e_o* "
T2 = "the relation between *the* [MASK]₁ *e_s* and *the* [MASK]₂ *e_o* is [MASK]₃ "
T3 = " [MASK]₁ is the relationship between *the* [MASK]₂ *e_s* and *the* [MASK]₃ *e_o* "

Figure 4. Different template constructions.

3.2. Combining Discrete and Continuous Prompts

Prompt learning can be divided into two construction forms: discrete prompts and continuous prompts. Discrete prompt learning uses discrete tokens to represent features, typically constructing templates with discrete words and sentences, which requires the templates to be in natural language. However, converting discrete words to feature representations inevitably loses some information, and prompt templates may not necessarily be in natural language. This leads to continuous prompt learning, where soft prompts such as vectors are added to the semantic space of pre-trained models. To integrate the advantages of both, this study combines discrete prompts with continuous prompts. The form of discrete prompts is the combination of the aforementioned sub-prompts, including subject sub-prompts, object sub-prompts, and relation sub-prompts. Continuous prompts involve adding a pseudo token at the beginning and end of each sub-prompt, which is a learnable token initialized randomly through embedding, obtaining a vector representation for each pseudo token. As shown in Figure 5, green represents the subject sub-prompt, orange represents the object sub-prompt, and blue represents the relationship sub-prompt. These three sub-prompts are fixed and intuitively understandable discrete prompts that can represent semantic information. The gray mark represents a continuous token, indicating the continuous prompts, which is randomly initialized by inputting into the pre-trained model. It can dig out implicit features and has strong flexibility.

Finally, the representations of original input, discrete prompts, and continuous prompts are input together into the pre-trained model to obtain new representations.

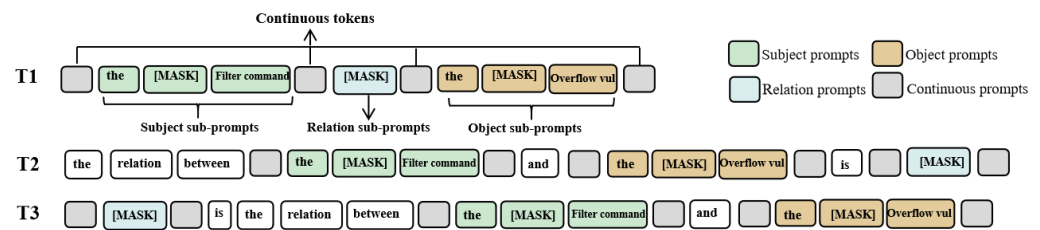


Figure 5. Construction of prompt templates.

4. Model Design

4.1. Overall Architecture

This chapter, based on prompt learning, designs a parameter-sharing-based relation extraction method using the rule-enhanced prompt learning, which is mainly divided into three major modules: entity recognition module, rule injection module, and relation extraction module. The model architecture is shown in Figure 6.

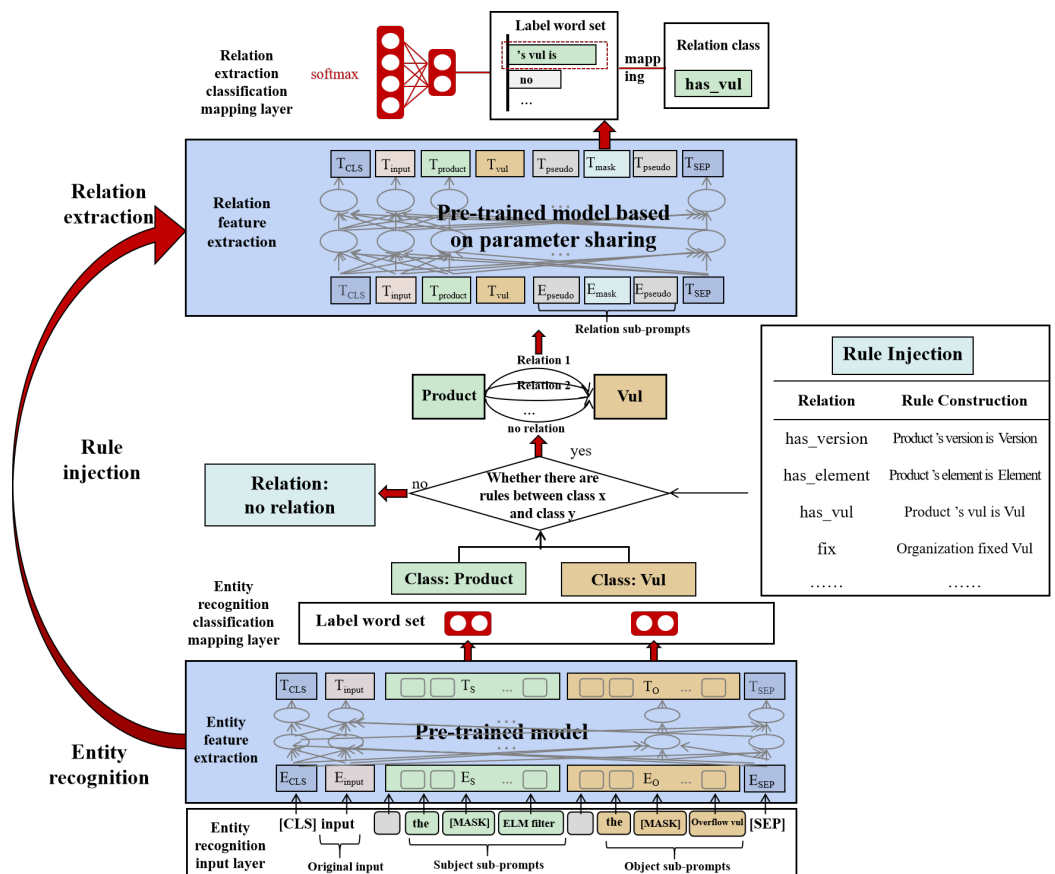


Figure 6. Architecture of multi-relation extraction model.

The first step of the model is entity recognition, which primarily involves identifying the entity classes of the subject and object. Due to the scarcity of samples in cybersecurity, this paper designs an entity recognition scheme based on prompt learning, which specifically includes the input layer, encoding layer, and classification layer. The input layer concatenates the original input, subject prompt templates, and object prompt templates. The encoding layer, based on large pre-trained models, embeds the input into token embeddings, position embeddings, and sentence embeddings. Stacking multi-head attention and feedforward modules enables the model to effectively utilize contextual information. Finally, the masked positions are classified for prediction, first classified into a set of label words, and then, the final entity classes are obtained based on a mapping function.

The second step of the model is a rule injection. After obtaining the entity classes of the subject and object in the first step, it is necessary to determine which relations exist between the two entities based on the injected rules. Rule injection is mainly aimed at addressing the issues of multi-relation extraction and unclear identification of negative samples. After constructing sub-prompts, without rule restrictions, each prompt sequence can form 768 templates. Therefore, based on the object properties in the ontology constructed in the previous text, rules are built for data augmentation prompt learning, incorporating cybersecurity knowledge into the prompts. These rules specify the types of relationships that may exist between two entities. In this study, a total of 12 relation examples are constructed, including “no relation”, for multi-relation extraction tasks between the same entities. This paper first judges “whether there are rules between two entities”. If there is no relationship between classes, it is directly judged as “no relation”. If rules exist, all possible relationships are identified and further predicted through the relation extraction module.

The third step of the model is relation extraction, with the primary task being to determine the specific relationship type based on the identified entities and rules using rule-enhanced prompt learning. The model consists of a feature extraction layer and a classification layer. There is no need to re-input the raw data for encoding and model training. Instead, entity embeddings and newly constructed relation sub-prompts are directly inputted into the model trained in the first step for fine-tuning. Finally, through the classification layer, the relation class is obtained by mapping the label word with the highest probability.

4.2. Entity Recognition Module

4.2.1. Input Layer

The input layer of entity recognition consists of two parts, the original input and the prompt template input, with sentence start and end separators [CLS] and [SEP] added at the beginning and end of the sentence. The template input includes subject prompts, object prompts, and continuous tokens, as shown in Figure 7.

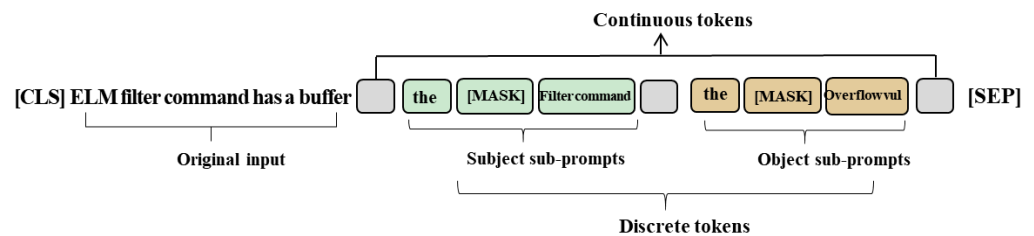


Figure 7. Entity recognition input template construction.

4.2.2. Feature Representation Layer

The feature representation layer converts text data into vector representations, undergoing feature selection and representation through deep neural networks. The feature representation layer mainly consists of an embedding encoding layer, network layer, attention layer, and feedforward layer, as shown in Figure 8.

The encoder adopts a large-scale pre-trained model as the base, and the input representation include token embeddings, position embeddings, and segmentation embeddings. Token embeddings encapsulate the semantic information of the text, position embeddings learn positional information among tokens to incorporate positional information into the encoding, and segmentation embeddings are used to represent the sentence structure.

$$PE(pos, 2i) = \sin(pos/10000^{2i/d_{model}}) \quad (8)$$

$$PE(pos, 2i + 1) = \cos(pos/10000^{2i/d_{model}}) \quad (9)$$

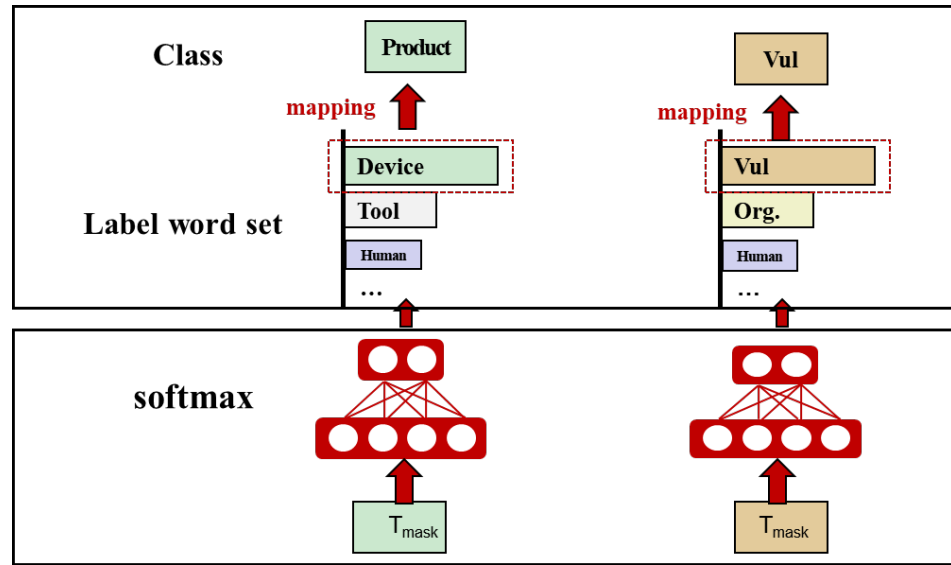


Figure 9. Structure of classification layer.

4.3. Rule Injection Module

In relation extraction tasks, many models perform well in classifying positive samples, but they often fall short in classifying negative samples, especially when two entity classes have a relation, yet the two entities have no relation in reality, leading to poor classification performance of the corpus. To address the issues of sample scarcity and poor classification of negative samples in cybersecurity relation extraction tasks, this study uses ontology rule-injected prompt learning for multi-relation extraction.

After constructing sub-prompts, if there are no rule restrictions, numerous templates can be formed for each prompt sequence. However, the subject and object corresponding to a relation are relatively fixed, not without limitations. Therefore, based on the ontology model constructed earlier, rules are built to enhance prompt learning, filling cybersecurity knowledge into the prompts. Based on the cybersecurity ontology constructed earlier, this study establishes the following rules, each consisting of “subject + relation + object”, which delimit the possible relationship types between two entities. The conditional functions of relations are set, where $f(x, \text{has_vul}, y)$ indicates whether x 's vulnerability is y . The relationship judgment requires the combination of entity conditional functions, and all conditional functions constitute a set of conditional functions $F(f \in F)$. Conditional functions are essentially predicates of first-order logic. Therefore, relations can be formally represented by three sub-prompts, where $f_{e_s}(\cdot, \cdot)$ is the conditional function determining the subject entity types, $f_{e_o}(\cdot, \cdot)$ is the conditional function determining the object entity types, and $f_{e_s, e_o}(\cdot, \cdot)$ is the conditional function determining the semantic connection between the subject and object.

$$f_{e_s}(x, \text{product}) \wedge f_{e_s, e_o}(x, \text{'s vul is}, y) \wedge f_{e_o}(y, \text{vulnerability}) \rightarrow \text{"has_vul"} \quad (15)$$

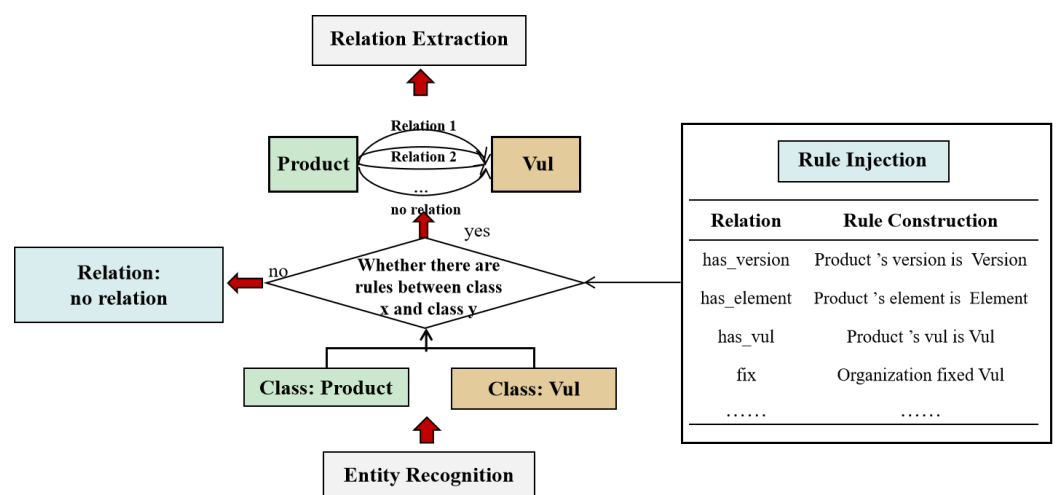
This study includes a total of 12 relation examples, including “no relation”, as shown in Table 2, for the multi-relation extraction task between the same entities.

To classify cybersecurity data with multiple relations more accurately and efficiently, this paper constructs a multi-relation extraction model based on rule-enhanced prompt learning, and builds a rule classifier based on prior knowledge. The entity recognition is performed first, and then multi-relation classification is conducted based on entity knowledge and injected rules. The specific process is shown in Figure 10.

Table 2. Rule construction table.

Relation	Rule Construction
has_version	$f_{e_s}(x, \text{product}) \wedge f_{e_s, e_o}(x, \text{'s version is, } y) \wedge f_{e_o}(y, \text{version})$
has_element	$f_{e_s}(x, \text{product}) \wedge f_{e_s, e_o}(x, \text{'s element is, } y) \wedge f_{e_o}(y, \text{element})$
because_of	$f_{e_s}(x, \text{vulnerability}) \wedge f_{e_s, e_o}(x, \text{is because of, } y) \wedge f_{e_o}(y, \text{cause})$
is_product_of	$f_{e_s}(x, \text{product}) \wedge f_{e_s, e_o}(x, \text{is product of, } y) \wedge f_{e_o}(y, \text{organization})$
has_vul	$f_{e_s}(x, \text{product}) \wedge f_{e_s, e_o}(x, \text{'s vul is, } y) \wedge f_{e_o}(y, \text{vulnerability})$
lead_to_consequence	$f_{e_s}(x, \text{vul}) \wedge f_{e_s, e_o}(x, \text{led to, } y) \wedge f_{e_o}(y, \text{impact})$
exploit	$f_{e_s}(x, \text{organization}) \wedge f_{e_s, e_o}(x, \text{exploited, } y) \wedge f_{e_o}(y, \text{product})$
use_means_of	$f_{e_s}(x, \text{organization}) \wedge f_{e_s, e_o}(x, \text{used means of, } y) \wedge f_{e_o}(y, \text{tool})$
develope	$f_{e_s}(x, \text{organization}) \wedge f_{e_s, e_o}(x, \text{developed, } y) \wedge f_{e_o}(y, \text{tool})$
cooperate_with	$f_{e_s}(x, \text{organization}) \wedge f_{e_s, e_o}(x, \text{cooperated with, } y) \wedge f_{e_o}(y, \text{organization})$
target_at	$f_{e_s}(x, \text{organization}) \wedge f_{e_s, e_o}(x, \text{targets at, } y) \wedge f_{e_o}(y, \text{organization})$
no_relation	$f_{e_s}(x, \text{entity}) \wedge f_{e_s, e_o}(x, \text{is irrelevant to, } y) \wedge f_{e_o}(y, \text{entity})$
Entities = product, version, vulnerability, cause, ...	

Based on the entity class obtained in the first step, the model searches the rule table to determine whether there are rules between these two entities. If no rule is satisfied, it is directly judged as “no relation”; if a relationship exists, the model proceeds to the next step for a specific relationship judgment.

**Figure 10.** Structure of rule injection.

4.4. Relation Extraction Module

After determining the existence of a relationship based on rules, the specific relationship is then determined. The entity knowledge obtained from the first step, the injected rule knowledge, and the relation sub-prompts are input into the trained model from the first step to continue training, resulting in the final relation class. The model representation is shown in Figure 11.

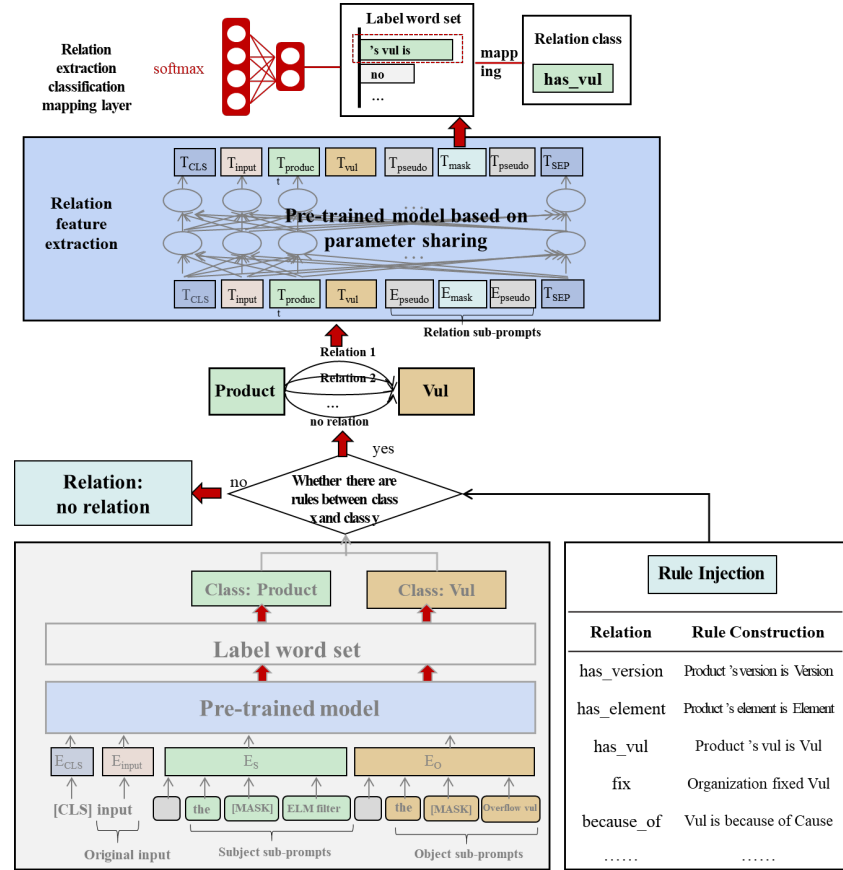


Figure 11. Relation extraction model.

4.4.1. Relation Feature Representation Layer

The input layer of relation extraction integrates knowledge from the entity recognition module and the rule injection module. The entity recognition module provides the subject and object classes, while rules provide potential relations between them. The relation sub-prompt consists of three [MASK] tokens and two continuous tokens. Therefore, the complete input consists of original corpus, subject class, object class, relation prompts, and special tokens marking the beginning and end. The feature representation layer comprises embedding, network, multi-head attention, and feedforward layers.

As shown in Figure 12, in the parameter-sharing-based model used in this paper, the relation extraction module does not require re-embedding. Instead, it directly uses the output representation T of the entity recognition module, connected with the relation sub-prompt E . The relation sub-prompt E consist of three [MASK] tokens and continuous tokens marking the beginning and end. The output representation of entity recognition comprises sentence representation and entity representation, along with special tokens [CLS] and [SEP], collectively forming the relation embedding representation, as follows:

$$E(r) = E_{pseudo}E_{mask}E_{mask}E_{mask}E_{pseudo} \quad (16)$$

$$T(e) = T_{CLS}T_{input}T_eT_oT_{SEP} \quad (17)$$

$$T(x) = T(e)E(r) = T_{CLS}T_{input}T_eT_oE_{pseudo}E_{mask}E_{mask}E_{mask}E_{pseudo}T_{SEP} \quad (18)$$

These representations are jointly input into the model trained by entity recognition, further fine-tuning with new inputs to obtain relation representations. The parameter-sharing-based model significantly reduces training costs and time, enabling efficient relation extraction.

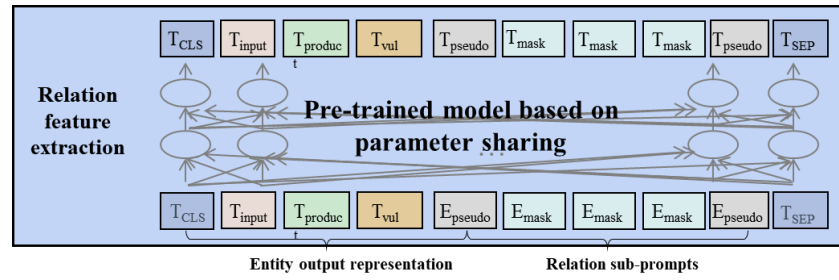


Figure 12. Relation feature representation layer.

4.4.2. Classification Layer

Firstly, all possible relations between two entity classes are searched in the rule table, including the “no relation” class. Likewise, using the Bert model to compute the hidden vector $h[MASK]$ of $[MASK]$ in the relation prompts, probabilities of the label words v at the masked positions are calculated sequentially. The label word with the highest probability is selected and mapped to the predetermined relation prompts. Since the relation extraction template contains multiple $[MASK]$ tokens, including one each for subject, object, and relation discrete prompts, as well as several continuous tokens, all masked positions need to be considered for prediction. Here, n is the number of masks in the template, $\phi_j(y)$ represents the j -th masked position $[MASK]_j$, and the cybersecurity class y is mapped to the set of label words V_{MASK_j} . The following equation represents the probability distribution on real cybersecurity class.

$$p(y|x) = \prod_{j=1}^n p([MASK]_j = \phi_j(y)|T(x)) \quad (19)$$

Therefore, for the rule-based prompt learning of cybersecurity constructed in this study, given the model $T(\cdot)$, label word set V , and mapping function ϕ , the learning objective is to maximize

$$\frac{1}{|X|} \sum_{x \in X} \log \prod_{j=1}^n p([MASK]_j = \phi_j(y)|T(x)) \quad (20)$$

In the context of rule-based prompt learning in this study, the loss function is refined into two types: pre-trained model loss (MLM loss) and relation classification task loss (CE loss). The MLM loss (Masked Language Model loss) is calculated based on the output encoding results of the Bert model. During pre-training, the Bert model learns contextual information of language by predicting masked words. In this study, MLM loss is used to calculate the cross-entropy between the predicted logits of each masked position and the true labels, measuring the model’s performance on word-level prediction. The CE loss (Cross Entropy loss) is employed for the relation classification task. In relation classification, the model needs to determine whether specific relations exist between entity pairs. For this, the model first calculates logits for each relation class and then transforms them into probability distributions through the softmax function. Finally, the cross-entropy between these probability distributions and the true sample labels is computed to obtain the loss for the relation classification task. By combining MLM loss and CE loss, this study comprehensively evaluates the model’s performance in rule-based prompt learning.

5. Experimental Design and Result Analysis

5.1. Experimental Environment and Parameters

The experiment in this paper is conducted on a server equipped with an NVIDIA V100 GPU, with a performance of 8 cores (vCPU) and 32 GiB of RAM. The software environment includes Ubuntu 18.04 64-bit operating system, PyTorch-1.7.1 development framework, and Python-3.7 programming language. The experiment is based on the base version of

the pre-trained Bert model for cybersecurity multi-relation extraction. This model contains approximately 22.7 M parameters, stacked into 12 layers of Transformers, with a hidden layer size of 768. Experimental parameter settings are shown in Table 3.

Table 3. List of experimental parameters.

Parameter Type	Value
gpu_train_batch_size	8
gradient_accumulation_steps	1
max_seq_length	512
warmup_steps	500
learning_rate	3×10^{-5}
learning_rate_for_new_token	1×10^{-5}
num_train_epochs	2
weight_decay	1×10^{-2}
adam_epsilon	1×10^{-6}

5.2. Construction of Cybersecurity Dataset

In order to verify the effectiveness of the model, this paper obtains a text intelligence training dataset from the mainstream Chinese information security community. The training data are cleaned, standardized, and structured to ensure clarity and accuracy.

(1) Data Crawling

All the data required for model training comes from text information on cybersecurity intelligence community websites. In this paper, the Python web crawler is used to crawl a large amount of data from major website platforms, including 10 Chinese sites such as “Anquanke”, “National Computer Network Emergency Response Technical Team/Coordination Center of China (CNCERT/CC)”, “WWW.YOUXIA.ORG”, “CNNVD”, “ZDNet”, “Hacker News”, as well as 6 English websites, totaling over 10,000 valid data entries. To further standardize the data format, the main focus is on obtaining information such as titles, dates, authors, sources, categories, views, comments, tags, content, links, and summaries.

(2) Data annotation

After the data text format cleaning, to form unified training set and test set samples and facilitate the subsequent annotation and classification, the data are imported into the MySQL database for shared storage.

To determine the features of the training set and test set, manual and machine annotations are employed to label object entities, attributes, relations, within the processed data text. For this model, a total of 13,903 text sentences have been annotated. In order to better reflect the characteristics of “big data, few shot” of cybersecurity data, this chapter does not adopt the traditional 8:1:1 split method for the training set–validation set–test set, but instead use a 1:1:8 ratio to better simulate real situations and verify the effectiveness of the model in small samples. The data are annotated with entity boundaries and relationship types, and the processed data format is shown in Figure 13.

```
{'token': ['There', 'is', 'a', 'vulnerability', 'in', 'the', 'CWD', '-', 'root', 'command', 'in', 'ftpd'],
'h': {'name': 'ftpd', 'pos': [11,11]},
't': {'name': 'CWD -root command', 'pos': [6, 9]},
'relation': 'has_element'}
```

Figure 13. Data format.

A total of 12 sets of relation examples are designed, including *has_version*, *has_element*, *because_of*, *is_product_of*, *has_vul*, *use_means_of*, *lead_to_consequence*, *exploit*, *develope*, *cooperate_with*, *target_at*, *no_relation*. The relationship definition is shown in Table 4.

Table 4. List of relation definitions.

Relation	Meaning	Quantity
has_version	Corresponding version(s) of involved product	3174
has_element	Subordination relationship between element(s) and the product	2252
because_of	Cause of existing vulnerability	71
is_product_of	A product belongs to a certain manufacturer	1697
has_vul	A product has vulnerability(ies)	4467
use_means_of	Attacker's use of a certain tool for network attacks	571
lead_to_consequence	Consequence caused by existing vulnerabilities	513
exploit	An organization exploits vulnerabilities for network attacks	117
develope	An organization develops a certain tool	58
cooperate_with	Cooperation and association between organizations and tools	27
target_at	Hostile relation between organizations	924
no_relation	No relation between two entities	32

has_version: This relation links the entities of a product and version number(s), indicating the corresponding version(s) of the product. For example, in the sentence “Sendmail versions 8.8.0 and 8.8.1 have a MIME buffer overflow vulnerability that can be exploited to gain root access”, the vulnerable versions of Sendmail are 8.8.0 and 8.8.1, thus there is a “*has_version*” relation between “Sendmail” and “versions 8.8.0 and 8.8.1”.

has_element: This relation connects a product and its element(s), indicating the subordinate relationship between the element(s) and the product. For example, in the sentence “Sendmail versions 8.8.0 and 8.8.1 have a MIME buffer overflow vulnerability that can be exploited to gain root access”, MIME is an element of Sendmail, thus there is a “*has_element*” relation between “Sendmail” and “MIME”.

because_of: This relation links a vulnerability with its cause, indicating the cause of the vulnerability. For example, in the sentence “Versions of Screen prior to 3.9.1.0 have a vulnerability related to multi-attach error”, the multi-attach error is a cause of the vulnerability, thus there is a “*because_of*” relation between “vulnerability” and “multi-attach error”.

is_product_of: This relation connects a product with its manufacturer, indicating that a certain product belongs to a certain manufacturer. For example, in the sentence “Microsoft Windows NT is a large-scale computer network operating system developed by the Microsoft Corporation”, Microsoft Windows NT is a product of Microsoft, thus there is an “*is_product_of*” relation between the entity “Microsoft Windows NT” and the entity “Microsoft Corporation”.

has_vul: This relation links a product with the vulnerability(ies), indicating the vulnerability(ies) present in the product. For example, in the sentence “Sendmail versions 8.8.0 and 8.8.1 have a MIME buffer overflow vulnerability that can be exploited to gain root access”, Sendmail has a buffer overflow vulnerability, thus there is a “*has_vul*” relation between “Sendmail” and “buffer overflow vulnerability”.

use_means_of: This relation connects an organization with a tool, indicating that the attacker uses a specific tool for network attacks. For example, in the sentence “The attacker caused a denial of service by sending excessive messages”, the means of attack by the “attacker” is “sending excessive messages”, thus there is a “*use_means_of*” relationship between them.

lead_to_consequence: This relation links a vulnerability with its consequence, indicating the consequence caused by the vulnerability. For example, in the sentence “There is a vulnerability in QMS CrownNet Unix Utilities version 2060, allowing root login without a password”, the vulnerability leads to the consequence of “allowing root login without a password”, thus there is a “*lead_to_consequence*” relation between them.

exploit: This relation connects an organization with a vulnerability, indicating that the organization exploits the vulnerability for network attacks. For example, in the sentence

“There is a buffer overflow vulnerability in the krb425_conv_principal function of Kerberos 5, allowing remote attackers to gain root privileges”, the “remote attackers” exploit the “buffer overflow vulnerability”, thus there is an “exploit” relation between them.

develope: This relation links an organization with a tool, indicating that an organization has developed a certain tool. For example, in the sentence “Fluxwire was created by the CIA to enable mesh networking”, the “CIA” created “Fluxwire”, thus establishing a “develop” relationship between the two.

cooperate_with: This relation connects organizations or tools, indicating cooperation or association between organizations or tools. For example, in the sentence “We believe that this malicious file is associated with some APT attack organizations in India, including Patchwork, BITTER, and Confucius”, the “malicious file” is associated with the “APT attack organizations Patchwork, BITTER, and Confucius”, thus there is a “cooperate_with” relation between them.

target_at: This relation links organizations, indicating hostile relation between them. For example, in the sentence “We believe that this malicious file is associated with some APT attack organizations in India, including Patchwork, BITTER, and Confucius”, the “malicious file” is associated with the “APT attack organizations Patchwork, BITTER, and Confucius”, thus there is a “target_at” relation between them.

no_relation: This relation indicates that there is no relationship between two entities.

5.3. Evaluation Metrics

This chapter deals with a multi-relation classification problem with imbalanced classes. The main evaluation metrics utilized are Precision and Recall, based on which the F1-score is computed. In the context of imbalanced multi-relation classification, this paper employs class-specific F1-score and Micro-F1 score to assess experimental results. These metrics are calculated based on the quantities of TP, FP, TN, and FN, as indicated in Table 5.

Table 5. Metrics for F1 calculation.

Name	Meaning
TP (True Positive)	Predicted as positive, the actual value is positive
FP (False Positive)	Predicted as positive, the actual value is negative
TN (True Negative)	Predicted as negative, the actual value is negative
FN (False Negative)	Predicted as negative, the actual value is positive

Precision represents the proportion of true positives among all positive predictions, while Recall denotes the proportion of predicted positives out of all positives. F1-score is the harmonic mean of the Precision and Recall. The subscript ‘*i*’ denotes the specific metric for the *i*-th class, and the formulas for calculation are as follows:

$$Precision_i = \frac{TP_i}{TP_i + FP_i} \quad (21)$$

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \quad (22)$$

$$F1_i = 2 \cdot \frac{Precision_i \cdot Recall_i}{Precision_i + Recall_i} \quad (23)$$

Micro represents the relevant indicators at the micro level. Micro-F1 score comprehensively takes into account the distribution of samples across classes, making it particularly suitable for handling imbalanced class distributions in the dataset. In such data environments, classes with a larger sample size have a more significant impact on the F1-score. The specific formulas for Micro-F1 score calculation are as follows:

$$Precision_{micro} = \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n TP_i + \sum_{i=1}^n FP_i} \quad (24)$$

$$Recall_{micro} = \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n TP_i + \sum_{i=1}^n FN_i} \quad (25)$$

$$F1_{micro} = 2 \cdot \frac{Precision_{micro} \cdot Recall_{micro}}{Precision_{micro} + Recall_{micro}} \quad (26)$$

5.4. Analysis of Model Method Experimental Results

Due to the few-shot nature of the cybersecurity domain, with limited annotated data, this paper adopts a data partitioning approach where the test set is larger than the training set. Initially, the dataset is split in a ratio of 1:1:8 for training, validation, and test sets, respectively, serving as the baseline partitioning method; while keeping the sizes of the validation and test sets constant, the training set is reduced to 0.1 times and 0.5 times its original size to evaluate the performance of the model under conditions of smaller training sets. The specific results are shown in Table 6.

Table 6. Experimental results.

Relation	Quantity	0.1:1:8			0.5:1:8			1:1:8		
		R	P	F1	R	P	F1	R	P	F1
no_relation	32	70.6	66.2	68.3	82.5	78.0	80.2	89.3	84.8	87.0
cooperate_with	27	0	0	0	0	0	0	0	0	0
develope	58	0	0	0	0	0	0	0	0	0
because_of	71	0	0	0	0	0	0	3.6	100	6.9
exploit	117	0	0	0	96.8	100	98.4	96.8	100	98.4
lead_to_consequence	513	90	88.5	89.2	99.8	88.3	93.7	99.5	88.5	93.7
use_means_of	571	8.8	85.1	15.9	90.4	82.7	86.4	89.0	87.3	88.2
target at	924	96.5	79.8	87.4	98.0	97.8	97.9	99.5	97.9	98.7
is_product_of	1697	85.6	74.7	79.8	92.2	87.4	89.7	95.4	84.3	89.5
has_element	2252	19.2	79.1	30.8	92.8	91.1	92.0	89.3	94.5	91.8
has_version	3174	97.8	62.4	76.2	96.6	99.2	97.9	97.6	98.3	97.9
has_vul	4467	99.8	94.8	97.3	99.9	99.9	99.9	99.9	99.9	99.9
micro	13,903	78.2	78.2	78.2	95.4	95.3	95.3	95.5	95.4	95.5

- (1) Effectiveness of extremely few shot. It can be observed that relations like “cooperate_with”, “develope”, “because_of”, and “exploit” belong to ultra-few-shot categories, with only about a hundred data points in total. Under the baseline partitioning, the training set for these relations is in single digits, and when reduced to 0.1 times, it becomes almost zero-shot. At zero-shot, the Recall and Precision for these relations are both zero, making it impossible to identify these relations. However, when the data are slightly increased to 0.5 times, the “exploit” relation can be accurately identified. With the sample size continuing to increase to the baseline dataset of 1:1:8, only two types of relations remain unidentified. This indicates that the model can still perform well in the context of extremely few shot, but it is unable to perform relation classification with completely zero samples.
- (2) Accurate identification of “no_relation”. Although “no_relation” constitutes an ultra-few-shot category in the dataset, it demonstrates a noticeable difference in classification effectiveness compared to other ultra-few-shot categories. It can be seen that on the 0.1:1:8 dataset, the F1-score for “no_relation” reaches 68.3%, steadily increasing with the increase in the training set. This indicates that the rules constructed for “no_relation” in this model are effective and can significantly improve the accuracy of the “no_relation” class.

- (3) Effectiveness of few shot. Classes like “*lead_to_consequence*” have relatively ample data but still fall under the few-shot category. Longitudinally, the increase in data volume leads to a noticeable improvement in the F1-score. Horizontally, increasing the training set size results in a significant improvement in performance, with F1-micro increasing from 78.2% to 95.5%. Overall, the F1-micro achieves outstanding results on the baseline dataset, reaching 95.5%, with the largest class “*has_vul*” achieving nearly 100% accuracy.

So, if the training set continues to increase, will the model have better performance? This paper conducts further comparison experiments by splitting the dataset into a ratio of 2:1:7, while keeping the sizes of validation and test sets constant, the size of the training set is changed to be three-fourths of its original, resulting in a dataset ratio of 1.5:1:7. The specific data are presented in Table 7.

Table 7. Performance differences with increased annotated samples.

Relation	Quantity	1.5:1:7			2:1:7		
		R	P	F1	R	P	F1
no_relation	32	93.7	89.5	91.5	94.3	90.5	92.4
cooperate_with	27	0	0	0	0	0	0
develope	58	2.4	16.7	4.3	0	0	0
because_of	71	40.8	76.9	53.3	57.1	75.7	65.1
exploit	117	96.3	98.7	97.5	93.8	100	96.8
lead_to_consequence	513	98.1	92.4	95.1	97.5	94.3	95.9
use_means_of	571	97.7	83.2	89.9	95.7	83.6	89.3
target at	924	98.9	98.6	98.8	98.8	98.2	98.5
is_product_of	1697	95.6	88.7	92.1	94.1	89.7	91.9
has_element	2252	91.3	95.6	93.4	92.3	93.8	93.1
has_version	3174	97.7	99	98.3	97.9	99	98.5
has_vul	4467	99.9	99.9	99.9	99.9	99.9	99.9
micro	13,903	96.3	96.2	96.3	96.3	96.2	96.3

From the table, it can be observed that with the increase in annotated data, the F1-micro under the new dataset remains at 96.3%, with an increase of 0.8 percentage points in F1, and a small number of relationships in the ultra-few-shot “*develope*” class can be identified. Comparatively, although the training set doubled, the F1 only increased by 0.8 percentage points from 95.5%. This precisely demonstrates the effectiveness of this model in the few-shot domain. Once a certain sample threshold is reached, the model can efficiently and accurately perform relation classification without the need for more annotated samples.

5.5. Analysis of Comparison Experiment Results

This chapter conducts comparison experiments on three models, K-adapter, SpanBert, and KnowPrompt, which have achieved excellent results in relation extraction tasks over the past two years. Firstly, a comparison is made on the general relation datasets TACRED and ReTACRED. Subsequently, using the cybersecurity dataset constructed in this paper and the same experimental parameters, the performance of each model in the cybersecurity domain is evaluated.

TACRED is one of the largest and most widely used general-purpose relation classification datasets, containing 42 relationship types, with 68,124, 22,631, and 15,509 samples in the training, validation, and test sets, respectively. ReTACRED is another version of the TACRED dataset, addressing some shortcomings of the original TACRED dataset, with 58,465, 19,584, and 13,418 instances in the training, development, and test sets, respectively, and it includes 40 relationship types.

K-adapter: K-adapter, proposed by Wang et al. [22] in 2021, is a classic method for injecting knowledge into PLMs (Pre-trained Language Models). Adapter acts as a “plugin”

loaded onto the outside of PLM. A pre-trained model can load multiple adapters, each representing a different type of knowledge. K-adapter designs two adapters: one for factual knowledge obtained from Wikipedia, and one for linguistic knowledge obtained from web text dependency parsing. This model adopts RoBERTa as its base model and demonstrates good performance in relation extraction tasks.

SpanBert: SpanBert, proposed by Joshi et al. [23] in 2020, is an excellent extension of BERT. The authors argue that span segments carry semantic information, thus changing the strategy of randomly masking tokens in BERT to masking a span segment instead. Additionally, they introduce the span-boundary objective (SPO), which stores span information in the representations of its boundary tokens. This model achieves state-of-the-art (SOTA) results on both the SQuAD and OntoNotes datasets.

KnowPrompt: KnowPrompt, proposed by Hu et al. [24] in 2022, is a model for relation extraction using prompt learning. The model constructs virtual entity words and virtual relation words and generates entities and relations by introducing external knowledge, thereby mapping to real labels. KnowPrompt adopts RoBERTa_large as its pre-trained model and achieves SOTA results in both fully supervised and few-shot scenarios.

We first compare Ours Model on two general relation extraction datasets, TACRED and ReTACRED. Since general datasets are not suitable for cybersecurity rules, the rule construction module has reconfigured rules for the 42 relations in the general datasets to enable model comparison. As shown in Table 8, the following can be observed:

- (1) Our model is highly effective on general datasets. On the TACRED dataset, the F1-score is 74.4%, which is 2 percentage points higher than KnowPrompt. On the ReTACRED dataset, the F1-score is 92.9%, which is 1.6 percentage points higher than KnowPrompt. Moreover, the baseline model used in this paper is Bert-base, which has fewer parameters compared to RoBERT-large used by KnowPrompt, and does not require external knowledge transfer.
- (2) Prompt learning paradigms are superior. It can be seen that fine-tuning methods based on pre-trained models such as Bert-base, Bert-large, ERNIE, and RoBERT achieve F1 scores ranging from 66% to 70% on the TACRED dataset. In comparison to KnowPrompt and our model, which are based on prompt learning paradigms, their performance is inferior.
- (3) Rule construction and knowledge injection are both effective. K-adapter utilizes Wikipedia knowledge base and web text information, KnowPrompt references external knowledge bases, and our model constructs rules, all achieving around 72% effectiveness on the TACRED dataset, proving the effectiveness of knowledge. However, compared to introducing large knowledge bases, this paper only needs to construct rules for the 42 relations to achieve similar results.

Table 8. Comparison experiment results on general datasets.

Model	TACRED	ReTACRED
Bert-base (Devlin et al., 2019)	66.0	-
Bert-large (Baldini Soares et al., 2019)	70.1	-
ERNIE (Zhang et al., 2019)	67.9	-
RoBERTa (Liu et al., 2019)	68.7	76
K-ADAPTER (Wang et al., 2020)	72.1	-
SPANBert (Joshi et al., 2020)	70.8	85.3
KnowPrompt (Chen et al., 2021)	72.4	91.3
Our Model	74.4	92.9

Next, this paper selects three models, K-adapter, SpanBert, and KnowPrompt, which perform well on the general dataset, for experimental comparison on the multi-relation cybersecurity dataset constructed in this chapter, and divides it into ten datasets based on the training set. It is worth noting that KnowPrompt and our model only require entity position information without the need for entity class information, while K-adapter and

SpanBert require both pieces of information. Therefore, the model uses a cybersecurity dataset with added class information. The comparison experiment results are shown in Table 9, and the visualization graph of the experimental results is shown in Figure 14, from which the following conclusions can be drawn.

Table 9. Comparison experiment results on cybersecurity dataset.

Model	0.1	0.3	0.5	0.7	1:1:8	1.2	1.4	1.6	1.8	2:1:7
K-adapter	0.02	0.03	0.03	0.14	32.7	32.5	32.2	32.2	32.2	35.4
SpanBert	34.1	46.2	59.4	62.1	72.4	73.6	75.4	78.1	81.1	88.8
KnowPrompt	26.2	37.6	75.1	79.9	89.0	91.1	95.5	95.7	97.4	96.2
Our Model	78.2	93.9	95.3	95.4	95.5	96.0	95.8	96.3	96.4	96.3

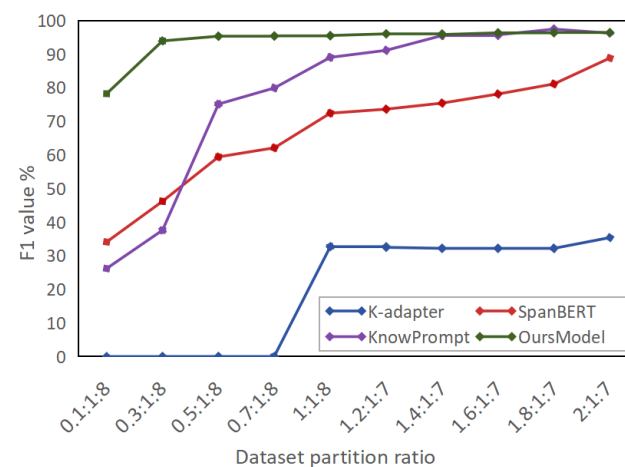


Figure 14. Line graph of comparison experiments on cybersecurity.

- (1) Our model is the most effective in the field of cybersecurity. Across almost all partitioned datasets, the multi-relation extraction model in this chapter achieved good results. Moreover, on the 0.1:1:8 dataset, only this model was effective, while the rest performed poorly with extremely few shots. This demonstrates the effectiveness of the rules and templates specifically built for cybersecurity data in this chapter, enabling efficient relation classification.
- (2) Significant improvement is observed with the increase in annotated samples. Overall, as the training set increases, all models show significant enhancement, reaching a critical point at the 1:1:8 dataset ratio where all models exhibit qualitative improvement. Therefore, this paper adopts 1:1:8 as the benchmark for few shot dataset partition.
- (3) Prompt-based paradigms are highly effective in the few-shot domain. Overall, the model effectiveness is ranked as follows: Our Model > KnowPrompt > SpanBert > K-adapter. In the field of cybersecurity, prompt learning paradigms are more effective than pre-training fine-tuning paradigms; while K-adapter achieved excellent results on the general dataset TACRED, it performed the worst on the dataset in this paper, with the F1-score of only 0.02% on the 0.1:1:8 dataset. As the number of annotated samples increases, the accuracy gradually improves, reaching 35.4% on the 2:1:7 dataset, but still significantly lagging behind the other three models. This indicates that pre-training fine-tuning paradigms like K-adapter are more suitable for large datasets and perform poorly in the few-shot domain. Both prompt learning-based models achieved around 96% effectiveness after reaching a training set size of 1.4. Further increasing the training set did not significantly improve performance, demonstrating the effectiveness of the prompt-based paradigm in the few-shot domain. The improvement in SpanBert's effectiveness is also significant, indicating the need for more attention to semantic information even in small samples.

- (4) Fast experimental speed. In summary, the multi-relation extraction model in this chapter has constructed cybersecurity rules and templates, using the Bert-base model. With fewer parameters and no need for annotated entity class information, it can quickly, efficiently, and accurately identify cybersecurity relations.

5.6. Analysis of Ablation Experiment Results

This subsection conducted ablation experiments on the model to evaluate the impact of different modules on the experiment's performance. Specifically, the rule injection module and the template construction module are removed separately to analyze the experimental effects.

- (1) Removal of the rule injection module. The cybersecurity rules are removed from the overall model to verify the effect of rules. The new model is represented as No_Rules.
- (2) Removal of the template construction module. The prompt templates are removed from the overall model, transforming the prompt learning model into a pre-trained fine-tuning model with rule injection. The new model is represented as No_Prompt.
- (3) Simultaneous removal of both modules. Both the cybersecurity rules and the prompt templates are removed from the overall model, transforming the prompt learning model into a pre-trained fine-tuning model. The new model is represented as No_RulesPrompt.

The results of the ablation experiments are shown in Table 10. The arrow indicates the decrease in F1 score after ablation experiments compared to the original model's performance. After removing the rule injection module, the F1-score of the No_Rules model decreased by 13 percentage points. After removing the template construction module, the F1-score of the No_Prompt model decreased by 21.9 percentage points. After removing both modules, the F1-score of the model decreased by 64.3 percentage points. This indicates the effectiveness of the rule injection and template construction modules, especially the significant improvement in relation extraction accuracy achieved by prompt template construction.

Table 10. Comparison experiment results on cybersecurity dataset.

Model	R	P	F1	Performance Decrease
No_Rules	84.7	80.4	82.5	13.0↓
No Prompt	74.9	72.4	73.6	21.9↓
No_RulesPrompt	31.9	30.5	31.2	64.3↓
Our Model	95.5	95.4	95.5	-

5.7. Model Performance Change Analysis

The performance will be further improved as the training set continues to grow, as evidenced by the experiment based on a 2:1:7 division ratio. At the same time, we have added an experiment time column to show the changes in experiment time as the training set increases. The training epoch is set to 2, and the changes are shown in Table 11. The visualization graph is shown in Figure 15.

Table 11. Model performance and experimental time.

Parameters	0.1:1:8	0.3:1:8	0.5:1:8	0.7:1:8	1:1:8
F1 Score	78.2	93.9	95.3	95.4	95.5
Experiment Time	3:01	3:10	3:41	4:13	4:53

As can be observed, with the increasing size of the training set, the F1 score gradually rises but at a decelerating rate, indicating that the model is gradually approaching its performance limit on this dataset. Simultaneously, the experimental time increases, and the growth rate accelerates as the training set expands, suggesting that as the dataset

size grows, the computational time required per unit of data also increases. This may be due to the saturation of computational resources, the complexity of the algorithm, or the characteristics of the dataset. The contrasting growth rates indicate that a larger dataset requires more computational resources for processing and analysis. Larger training sets often contain more features and more complex data patterns, thus necessitating greater computational resources for data loading, model training, and performance evaluation.

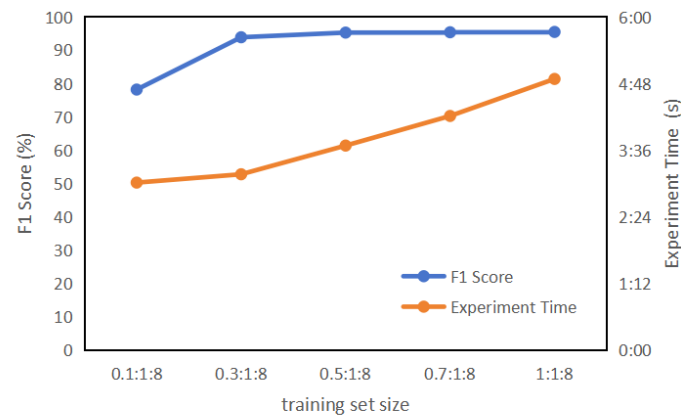


Figure 15. F1-Score experimental time graph.

6. Summary

This paper addresses the challenges of sample scarcity, zero-shot recognition of “no relation”, and computational redundancy in the field of cybersecurity. It constructs a multi-relational dataset for cybersecurity and a cybersecurity multi-relation extraction model based on parameter sharing. By introducing prompt learning, which is significantly effective in the few-shot domain, this chapter designs prompt templates combining discrete and continuous tokens and uses rule injection in prompt learning to filter out “no relation” and candidate set of relation classes. Finally, based on the Bert-base pre-trained model, a cybersecurity multi-relation extraction model based on parameter sharing is built. Specifically, it first constructs entity prompt templates combining discrete and continuous tokens and identifies the classes of two entities based on prompt learning. Then, the rule injection module mainly identifies whether it belongs to the “no relation” class. Based on the rule table constructed from sub-prompt combinations, if there is no connection between the classes of two entities, it is classified as “no relation”; if a connection exists, the candidate relation set is filtered out. Finally, based on the shared model parameters of the entity recognition model, relation prompt templates are constructed, and relation classes are determined from the candidate relation set based on prompt learning and rule judgment. In the experimental section, this chapter introduces the cybersecurity dataset constructed in this paper and conducts experimental analysis on the comparative models on the general datasets TACRED and ReTACRED, as well as the cybersecurity dataset constructed in this paper, significantly demonstrating the effectiveness of our model.

Author Contributions: Conceptualization, Z.D. and F.W.; methodology, K.L.; software, K.L.; validation, L.X., Y.Z. (Yu Zhao) and Y.Z. (Yun Zhou); formal analysis, L.X.; investigation, Y.Z. (Yu Zhao); writing—review and editing, F.W. and Z.D.; supervision, Y.Z. (Yun Zhou). All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Data Availability Statement: Dataset 1: TACRED is available at <https://nlp.stanford.edu/projects/tacred/> accessed on 13 June 2024; Dataset 2: ReTACRED is available at <https://paperswithcode.com/paper/re-tacred-addressing-shortcomings-of-the> accessed on 13 June 2024; Dataset 3: Baidu DUIE2.0 is available at <https://aistudio.baidu.com/datasetdetail/180082> accessed on 13 June 2024; Dataset 4: Ours data is available at <https://github.com/21wangfei/Cybersecurity-Database> accessed on 13 June 2024.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, H.; Han, W.; Lai, X.; Lin, D.; Ma, J.; Li, J. Overview of Cyberspace Security. *Sci. China Inf. Sci.* **2016**, *2*, 125–164.
2. Ding, Z.; Liu, K.; Liu, B.; Zhu, X. Research Review of Network Security Knowledge Graph. *J. Huazhong Univ. Sci. Technol.* **2021**, *49*, 79–91.
3. Xia, Z.; Qu, W.; Gu, Y.; Zhou, J.; Li, B. Review of Entity Relation Extraction based on deep learning. In Proceedings of the 19th Chinese National Conference on Computational Linguistics, Haikou, China, 30 October–1 November 2020; Sun, M., Li, S., Zhang, Y., Liu, Y., Eds.; Chinese Information Processing Society of China: Beijing, China, 2020; pp. 349–362.
4. Wang, X.; Peng, M.; Sun, M.; Li, P. OIE@OIA: An Adaptable and Efficient Open Information Extraction Framework. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 22–27 May 2022; Muresan, S., Nakov, P., Villavicencio, A., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2022; pp. 6213–6226. [\[CrossRef\]](#)
5. Van Nguyen, M.; Lai, V.D.; Nguyen, T.H. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. *arXiv* **2021**, arXiv:2103.09330.
6. Alsaedi, M.; Ghaleb, F.A.; Saeed, F.; Ahmad, J.; Alasli, M. Cyber threat intelligence-based malicious URL detection model using ensemble learning. *Sensors* **2022**, *22*, 3373. [\[CrossRef\]](#) [\[PubMed\]](#)
7. Zhou, S.; Liu, J.; Zhong, X.; Zhao, W. Named Entity Recognition Using BERT with Whole World Masking in Cybersecurity Domain. In Proceedings of the 2021 IEEE 6th International Conference on Big Data Analytics (ICBDA), Xiamen, China, 5–8 March 2021; pp. 316–320. [\[CrossRef\]](#)
8. Ranade, P.; Piplai, A.; Joshi, A.; Finin, T. CyBERT: Contextualized Embeddings for the Cybersecurity Domain. In Proceedings of the 2021 IEEE International Conference on Big Data (Big Data), Orlando, FL, USA, 15–18 December 2021; pp. 3334–3342. [\[CrossRef\]](#)
9. Chen, Y.; Ding, J.; Li, D.; Chen, Z. Joint BERT Model based Cybersecurity Named Entity Recognition. In Proceedings of the 2021 4th International Conference on Software Engineering and Information Management, Yokohama, Japan, 16–18 January 2021; ICSIM '21; pp. 236–242. [\[CrossRef\]](#)
10. Li, D.; Zhang, Y.; Li, D.; Lin, D. Review of Entity Relation Extraction Methods. *J. Comput. Res. Dev.* **2020**, *57*, 25.
11. Gasmı, H.; Laval, J.; Bouras, A. Information Extraction of Cybersecurity Concepts: An LSTM Approach. *Appl. Sci.* **2019**, *9*, 3945. [\[CrossRef\]](#)
12. Wang, X.; Xiong, M.; Luo, Y.; Li, N.; Jiang, Z.; Xiong, Z. Joint Learning for Document-Level Threat Intelligence Relation Extraction and Coreference Resolution Based on GCN. In Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Guangzhou, China, 29 December–1 January 2020; pp. 584–591. [\[CrossRef\]](#)
13. Pingle, A.; Piplai, A.; Mittal, S.; Joshi, A.; Holt, J.; Zak, R. RelExt: Relation Extraction using Deep Learning approaches for Cybersecurity Knowledge Graph Improvement. In Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), Vancouver, BC, Canada, 27–30 August 2019; pp. 879–886. [\[CrossRef\]](#)
14. Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; Neubig, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.* **2023**, *55*, 1–35. [\[CrossRef\]](#)
15. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
16. Schick, T.; Schütze, H. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics, Online, 19–23 April 2020.
17. Ding, N.; Chen, Y.; Han, X.; Xu, G.; Xie, P.; Zheng, H.T.; Liu, Z.; Li, J.; Kim, H.G. Prompt-learning for fine-grained entity typing. *arXiv* **2021**, arXiv:2108.10604.
18. Gao, T.; Fisch, A.; Chen, D. Making Pre-trained Language Models Better Few-shot Learners. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Virtual Event, 1–6 August 2021.
19. Han, X.; Zhao, W.; Ding, N.; Liu, Z.; Sun, M. PTR: Prompt Tuning with Rules for Text Classification. *arXiv* **2021**, arXiv:2105.11259.
20. Jiang, Z.; Xu, F.F.; Araki, J.; Neubig, G. How Can We Know What Language Models Know? *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 423–438. [\[CrossRef\]](#)
21. Cui, L.; Wu, Y.; Liu, J.; Yang, S.; Zhang, Y. Template-based named entity recognition using BART. *arXiv* **2021**, arXiv:2106.01760.
22. Wang, R.; Tang, D.; Duan, N.; Wei, Z.; Huang, X.; Ji, J.; Cao, G.; Jiang, D.; Zhou, M. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online, 1–6 August 2021; Zong, C., Xia, F., Li, W., Navigli, R., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 1405–1418. [\[CrossRef\]](#)

23. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [[CrossRef](#)]
24. Hu, S.; Ding, N.; Wang, H.; Liu, Z.; Wang, J.; Li, J.; Wu, W.; Sun, M. Knowledgeable Prompt-tuning: Incorporating Knowledge into Prompt Verbalizer for Text Classification. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, 22–27 May 2022; Muresan, S., Nakov, P., Villavicencio, A., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2022; pp. 2225–2240. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.