A Joint Model of Entity Linking for RFC Protocols Knowledge Graph Construction

Shoubin Li [1,2] and Tao Luan [2]
[1]University of Chinese Academy of Sciences
[2]The Institute of Software, Chinese Academy of Sciences
{shoubin, luantao}@iscas.ac.cn

***Abstract*—Applying knowledge representation and reasoning to downstream tasks has been considered a promising research direction, as it enables semantic analysis of network protocols. Knowledge Graph is a new way of collecting knowledge, and building a protocol knowledge graph based on RFCs can help us study and analyze network protocols more effectively. However, automatically constructing a protocol knowledge graph from RFCs poses a major challenge, particularly in terms of extracting and linking protocol entities, due to the semi-structured nature of RFC documents. In this paper, we propose a model that combines a fine-tuned language model with an RFC Domain Model to link entities in RFCs to categories in the protocol knowledge base. Firstly, we design a protocol knowledge base as the schema for protocol entity linking. Secondly, we use heuristic methods to identify protocol entities and infer their descriptions from the nearby contexts of their header fields. Finally, we conduct comprehensive experiments on the RFC dataset using our joint model and baseline methods for protocol entity linking. Experimental results demonstrate that our model achieves state-of-the-art performance in entity linking on the RFC dataset, outperforming all baseline methods. In addition, we release a protocol knowledge graph, RFC-KG[1].**

***Index Terms*—Request for Comment, Entity Linking, Knowledge Graph, Protocol Analysis**

I. Introduction

Knowledge representation and reasoning based on ontology have become hot topics in research and application of network protocols. Garg et al [12] employed the ontological approach to expand the information on attacks and facilitate better prevention of such attacks by leveraging the advanced reasoning capabilities inherent in ontologies. Labonne et al [19] presented an unsupervised anomaly-based intrusion detection solution that can detect previously unseen network attacks, focusing on protocol header analysis. Zheng et al [43] generated an ontology using deep learning through neural network embeddings to enhance intelligent intrusion detection for cybersecurity. Khurat et al [17] proposed an ontology for SNORT rules to support the verification of SNORT rules using OWL ontology. Jero et al [16] learned protocol rules from RFC documents for six protocols (GRE, IPv6, IP, TCP, DCCP, and SCTP) to identify software vulnerabilities by injecting well-formed inputs generated based on rules that encode application semantics. However, existing methods for constructing network knowledge based on protocols are limited to specific protocols or specific application scenarios. The acquired knowledge has certain shortcomings, such as small scale, weak generalization ability, and dependence on manual expert experience. To address these issues, it is necessary to build a protocol knowledge base from a larger and more standardized protocol corpus using automated construction methods.

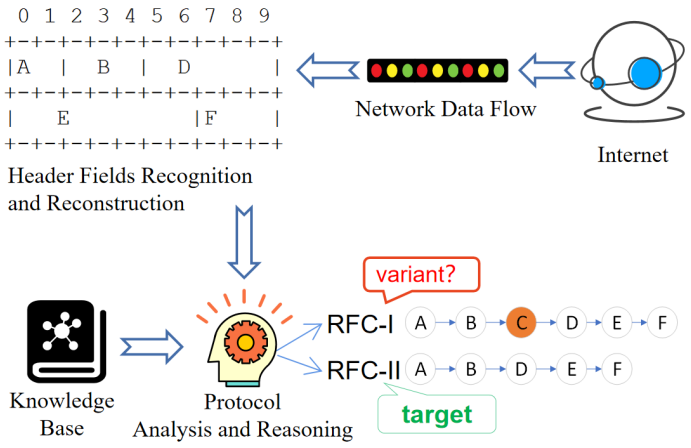

Fig. 1. Knowledge-based Protocol Analysis Process. This network data flow may be a variant of RFC-I or is the instance of RFC-II.

Utilizing the knowledge reasoning ability of the knowledge map, network protocol analysis tasks can be performed more intelligently. As shown in Figure 1, firstly, protocol classification methods ([5], [22], [23], [31], [38]) are used to analyze the protocol information of a network packet segment, such as the protocol field headers. Secondly, the knowledge-based protocol method is used to identify which RFC protocol the packets belong to (RFC-II). In particular, some protocol variants can also be identified (RFC-I). Building a knowledge base in the field of network protocols is significant for network analysis. Methods of knowledge base construction based on expert experience are time-consuming and labor-intensive, while automated construction methods require large-scale, standardized corpora. Request for Comments (RFCs)[2] have since become official documents for Internet specifications, communications protocols, procedures, and events. RFCs contain both structured and unstructured information and provide the fundamental corpus for automatically constructing a larger-scale network protocol knowledge base. Structured information includes RFC number, title, subtitle, abstract/introduction,

[1]url:http://39.104.126.169:7474/browser/,user:neo4j, pwd:123456

[2]https://www.rfc-editor.org/

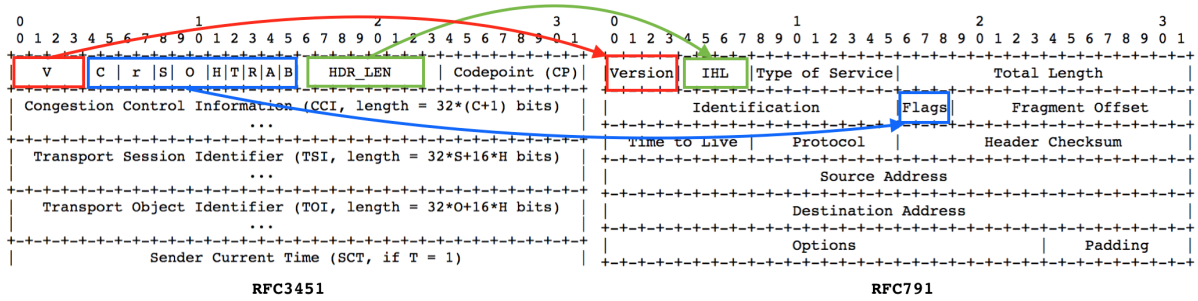

Fig. 2. Examples of Various Writing Styles in RFCs. Header field "Verion" is written as "Version" in RFC791, whereas "V" is used in RFC3451. Header field "Header Length" is written as "IHL" in RFC791, whereas "HDR_LEN" is used in RFC3451. Header field "Flag" is written as "Flag" in RFC791, whereas every flag bit is displayed in RFC3451.

and date of publication, while unstructured information refers to the natural language text describing the purpose/body of RFCs. Our main observation is that there is much hidden knowledge in the RFCs that can help us analyze network protocols. For example, mining the associations between protocols (field similarity, state machine similarity, etc.) can provide a clearer and more intuitive understanding of the development of specific protocols.

Knowledge graphs are an effective way to represent semantic real-world facts in a structured form. A knowledge graph is a collection of triplets, each indicating a piece of fact in the form of (Head-Entity, Relation, Tail-Entity). In the era of knowledge engineering, numerous KGs, such as YAGO [26], WordNet [25], DBpedia [20], and Freebase [4], have been developed. KGs contain a large amount of prior knowledge and can effectively organize data. Knowledge graphs have been widely adopted and have shown promising benefits in a wide range of applications, such as information retrieval, question answering, and knowledge reasoning. Additionally, facts extracted from specific domains convey domain knowledge that is usually only accessible to experts in those areas. Most existing works on Knowledge Graph Construction share several limitations, such as the requirement for sufficient external resources like large-scale knowledge graphs and concept ontologies as the starting point. However, such extensive domain-specific labeling is highly time-consuming and requires expertise, especially in the field of network protocol analysis. Therefore, the construction of knowledge graphs conveying domain knowledge, especially for network protocol analysis, is of great significance to the success of many domain-specific real-world applications related to protocols.

In this paper, we present the construction of a domain-specific knowledge graph, RFC-KG, which enhances the breadth and depth of knowledge in the field of network communication using the IETF database. We aim to fully exploit the potential knowledge within RFC documents to facilitate better analysis of network protocols. While RFC documents provide precise specifications, such as RFC5385 which defines the writing template, and RFC7322 which stipulates the writing style, the writing styles of authors differ significantly (Fig. 2). Additionally, some RFC documents even have distinct layouts, presenting significant challenges for Entity Linking (EL) in RFC-KG. EL is a vital process in the construction of RFC-KG, as it involves recognizing and disambiguating named entities in RFCs and linking them to a Protocol Knowledge Base (PKB) (Fig. 3), defined by domain-specific experts. To address this, we propose a novel model, RFC-BERT, which combines a fine-tuned model with an RFC Domain Model to tackle the EL task in RFCs. Our experimental results demonstrate that our model outperforms all baselines, achieving Precision, Recall, and F1-score of **73.70%, 74.70%, 74.20%** on the RFCs dataset.

In summary, our contributions are as follows:

1. Through a comprehensive analysis of over 8000 RFC specifications, we design a Protocol Knowledge Base (PKB) as the protocol domain schema, which serves as a guide for the subsequent protocol entity linking work.

2. To the best of our knowledge, RFC-KG is the first knowledge graph specific to RFCs, providing researchers with the ability to analyze and utilize protocols in a more granular manner.

3. We propose the RFC-BERT model, which combines a fine-tuned language model with an RFC domain model. Our model achieves superior performance in the RFC Entity Linking task, surpassing all baselines.

In the subsequent sections, we first provide an overview of related research, followed by a detailed description of the experimental process and an elaboration of the evaluation methods. Lastly, we provide concluding remarks.

II. Related Work

A. Entity Linking

Entity Linking (EL) is the task of linking entity mentions in text to their corresponding ontologies. Most EL approaches aim to link mentions to a comprehensive Knowledge Base (KB) [40]. Recent approaches utilize neural networks [11], [35] to capture the correspondence between a mention's context and a proposed entity in the KB. Graph-based [14], [41] and various joint methods [18], [24] are also widely used. In addition to linking to KB, there are also approaches that perform EL on ad-hoc entity lists [21]. In our experiment, we focus on the former, namely EL on KB. We aim to bridge the

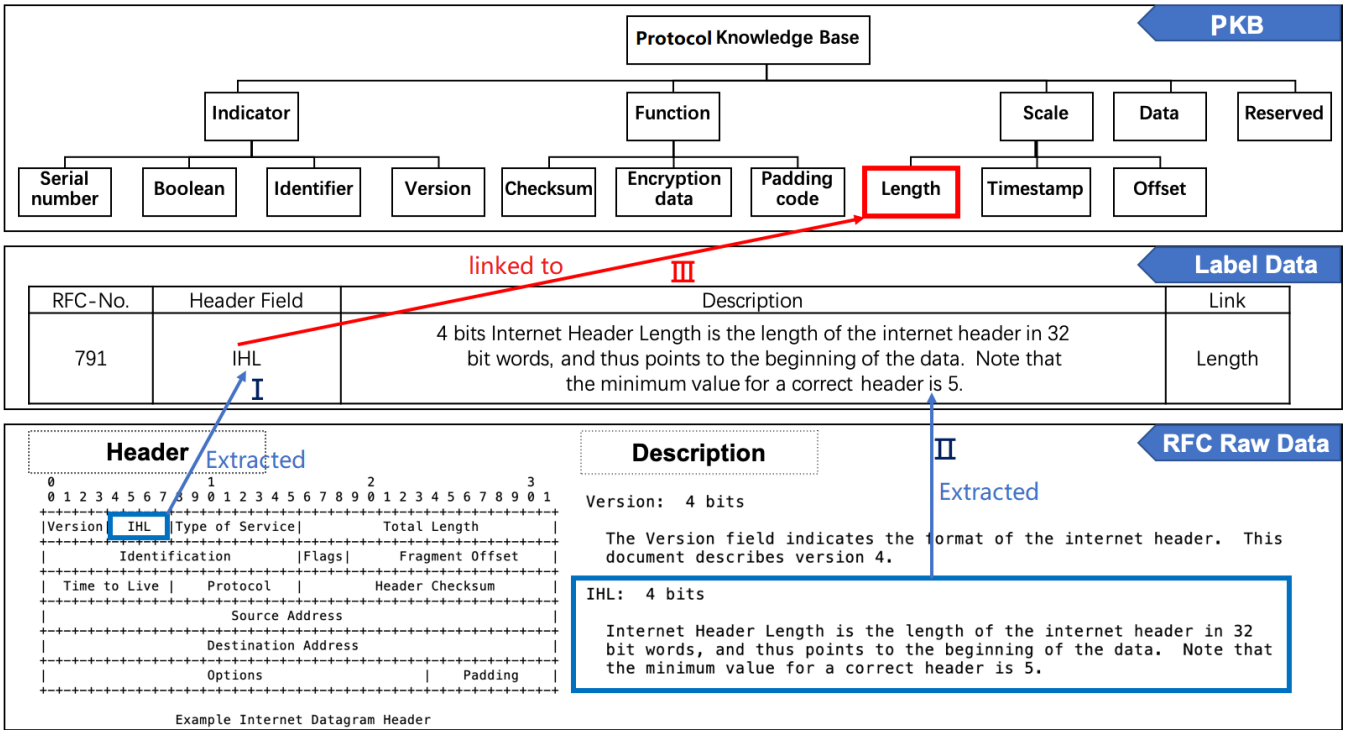

Fig. 3. Overview of Entity Linking in RFCs. I. Entity Extraction. II. Context Inference. III. Entity Linking.

gap between mentions in real-world text and entities in well-established theoretical schemas. Jero et al. [16] conducted a similar study, linking mentions from RFC documents to a list of ontologies to generate grammar-based fuzzing. With limited training data, they generalized this problem by assigning the property with the maximum keyphrase overlap to a header field.

B. Fine-tuning BERT for Classification

Pre-trained language models [10], [15], [30] have become a robust approach for EL. They capture rich language information from text by integrating pre-trained language representations with downstream tasks, thereby improving accuracy in many NLP applications. Among these models, BERT [10] has emerged as the most prominent in recent NLP studies. Through its self-attention mechanism, BERT encodes bidirectional contextual information at the character-level, word-level, and sentence-level, reducing discrepancies among single words. Fine-tuning BERT has also demonstrated optimal results in various downstream tasks, including Named Entity Recognition [32], Text Classification [34], and EL [6]. However, the initial BERT pre-training was conducted on generic datasets, such as Wikipedia and Book Corpus, for general purposes. It lacks domain-specific knowledge from RFCs. Our experiments have shown that standalone fine-tuned BERT is insufficient to achieve highly accurate EL in RFCs.

C. Learning with Scarce Annotations

In this paper, we also address the issue of data scarcity. Since our dataset is manually annotated, alleviating data scarcity enables us to train our model on a relatively small dataset, thus significantly reducing human effort. Transfer learning (TL) has been widely applied in previous research [1], [13], where classifiers trained on large datasets similar to, but not the same as, the target dataset are used to perform new tasks. Active Learning (AL) is another approach to dealing with data scarcity [3], [37]. AL selects queries or sub-spans that are most informative and improves its learning results through iterative processes. Bootstrapping, a widely used technique in Entity Set Expansion (ESE) [7], [39], enables classifiers to use their own predictions to enhance their performance, resulting in enriched datasets by acquiring new samples in each iteration. In our experiment, we utilize AL and Bootstrapping methods.

III. Preliminary

In this section, we introduce two important concepts used in our work.

a) Abstract Concept: The term ”abstract concept” refers to phrases that contain little semantic information. They are either short, consisting of a few words, or highly specialized terminologies in a specific domain. When trained using a pre-trained language model, the information from abstract concepts is often overshadowed by the dominant preferences in the general corpus where the language model is initially trained. However, these abstract concepts preserve highly valuable information about the specialized domain, which is crucial for solving domain-specific problems.

b) Narrative Description: The term ”narrative description” refers to a text chunk that conveys abundant semantic

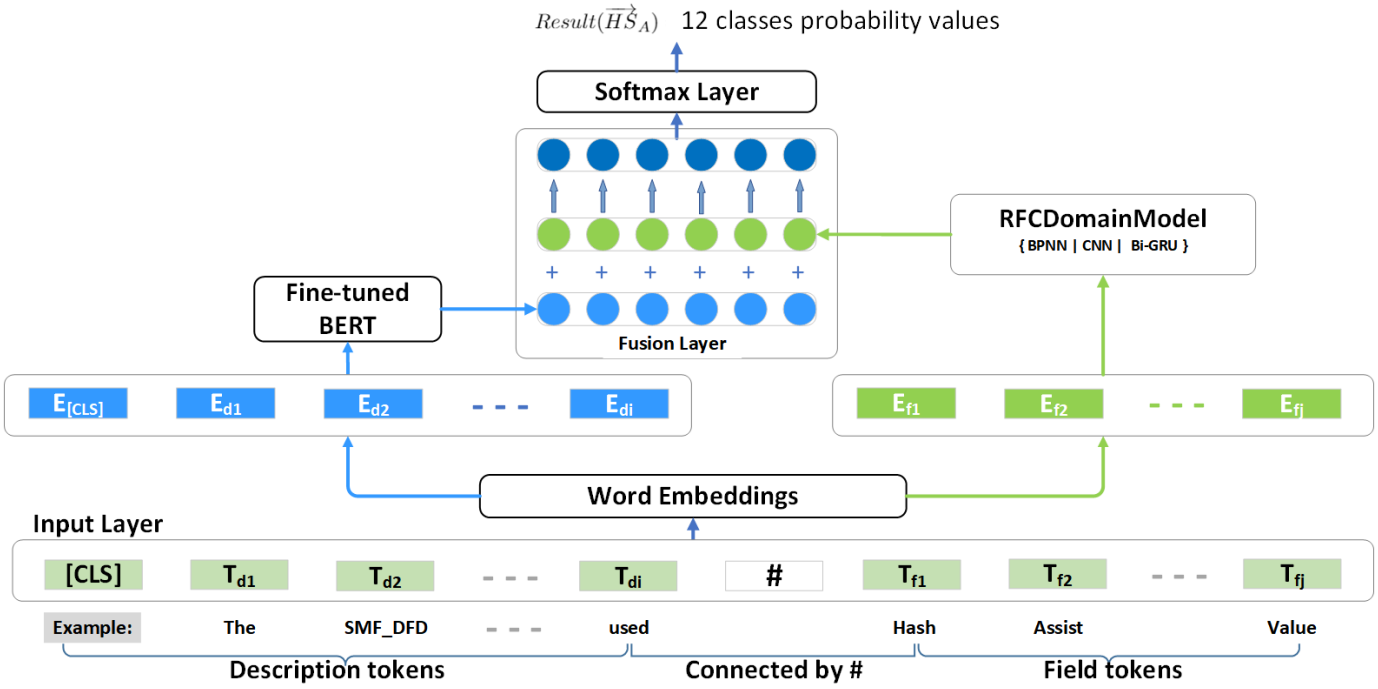

Fig. 4. RFC-BERT Model Architecture

information. However, this information is oftentimes too general and insufficient for performing domain-specific tasks.

In our task specifically, abstract concepts refer to RFC (Request for Comments) header fields, and narrative descriptions refer to the texts describing these header fields.

IV. Approach

In this section, we provide detailed explanations of the implementation details of our RFC-BERT model (Fig. 4).

I. Input: The input consists of header descriptions concatenated with header fields. The header field is parsed through header graphs (Fig. 3), while the description is the text chunk that references its corresponding header field. We infer the description from the nearby contexts of its header field using Zero-Shot Learning (ZSL) [28], similar to Jero [16].

II. Word Embeddings: We first tokenize the text into tokens as the basic units. To improve performance, we use 64-dimensional Glove word embeddings [29] that are pre-trained on a corpus consisting of 6 billion words from Wikipedia and Gigaword. These embeddings serve as the initial vectors for the corresponding words. Additionally, inspired by [33], we observe that there are obvious part-of-speech (POS) patterns or templates present in header field descriptions. Intuitively, incorporating POS tag information into word representations can enhance semantic understanding by introducing explicit lexical information. Therefore, we augment word representations with POS tag information to enhance their features. Specifically, each type of POS tag is initialized as a random vector with a uniform distribution and is optimized during training. Hence, each word can be represented as E_i = [$we_i \oplus pos_i$], where we_i represents the corresponding word embedding and pos_i represents the embedding of the POS tag of the word.

III. Fine-tuned BERT: The word embeddings for the description, denoted as $E_{(CLS)}$, E_{d1}, E_{d2}, ..., E_{dn}, $E_{(SEP)}$, are fed into the BERT model. The output, denoted as $\overrightarrow{HS}_{desc}$, represents the deep semantic features for the description.

$$\overrightarrow{HS}_{desc} = BERT(E_(CLS), E_{d1}, E_{d2}, ..., E_{dn}, E_{(SEP)}) \quad (1)$$

IV. RFC DOMAIN MODEL: The RFCDomainModel takes the word embeddings of header fields denoted as E_{h1}, E_{h2}, ..., E_{hn} as input, which can be one of BPNN [36], CNN [8], or Bi-GRU [9]. Its output is $\overrightarrow{HS}_{header}$.

$$\overrightarrow{HS}_{header} = RFCDomainModel(E_{h1}, E_{h2}, ..., E_{hn}) \quad (2)$$

where RFCDomainModel can be BPNN [36], CNN [8], or Bi-GRU [9].

V. FUSION LAYER: The fusion layer combines the outputs of BERT and RFCDomainModel, resulting in HS_A.

$$\begin{aligned} \overrightarrow{HS}_A &= \overrightarrow{HS}_{header} \bigoplus \overrightarrow{HS}_{desc} \\ &= [E_{d1} + E_{h1}, E_{d2} + E_{h2}, ..., E_{dn} + E_{hn}] \end{aligned} \quad (3)$$

VI. SOFTMAX LAYER: The output of the fusion layer is processed by the softmax function, resulting in multi-class probability values. The maximum probability value, *Result*, represents the classification result, belonging to one specific class in PKB, for the current inputs. This process can be formalized as follows:

$$Result(\overrightarrow{HS}_A) = \max(\text{softmax}(\overrightarrow{HS}_A)) = \max\left\{\frac{e^{score_1}}{\sum_{j=1}^{n} e^{score_j}}, ..., \frac{e^{score_n}}{\sum_{j=1}^{n} e^{score_j}}\right\} \quad (4)$$

where n is the total number of PKB classes (constant 12), and $score_i$ is the probability of belonging to one class in PKB.

V. Experiment

Our experimental data is extracted from RFCs. The workflow of constructing the knowledge-based dataset can be summarized as follows:

A. Experimental Setup

Protocol Knowledge Base:

We carefully analyzed more than 8000 RFC documents and designed a protocol knowledge base (Fig. 3).

Data Preparation: In our experiment, each training sample consists of a header field, a description text that describes this header field, and a label (belonging to PKB), represented as a triple: (Header Field, Description, Label). For example, (IHL, 4 bits Internet..., Length) (Fig. 3 Label Data). We extracted Header Fields and Descriptions from RFCs.

VI. Pre-process

We begin by downloading all plain text files of RFCs from the official website of RFC-editor. Despite the varied writing styles in RFCs, we have identified certain patterns in the description of header fields and the corresponding passage. Typically, these descriptions consist of two parts separated by different spacings. The first column indicates the field name, while the second column contains detailed explanations. Additionally, each RFC includes a graph providing an overview of the entire data frame for the protocol. Moreover, we observe consistent graphs of data packets across numerous RFC documents. For example, vertical bars act as delimiters between adjacent fields, and each line consists of 32 bits outlined with plus and minus signs.

Based on these patterns, we establish a set of binary features related to character-level characteristics, writing style, and context. These features encompass capitalization, term frequency, and indicative punctuations such as colons. By leveraging these binary features, we are able to extract field-description pairs. We narrow down our data source to 921 RFCs out of the initial set of over 8000 RFCs. By employing this rule-based approach, we successfully extract 11020 field-description pairs in a relatively standard format.

RFC documents are plain text files that include page breaks and page headers to facilitate printing. We identify and eliminate these page breaks and headers. RFCs follow a structured format where the document is divided into multiple sections, each focusing on a specific topic, such as a packet field, protocol state, or particular action. Our analysis reveals that this structure is highly beneficial for extracting entity types. Consequently, we maintain the structure by parsing the text itself into a hierarchical structure of sections. Each section includes a header line, body text, and potentially subsections.

VII. Entity Extraction

The task of Entity Extraction involves extracting all header fields in each RFC document. The preserved document structure from the pre-processing step enables us to extract this information using a simple rule-based system. Specifically, each packet field entity is sequentially described within a section, with the section title indicating the name and size of the entity. Therefore, we scan all section headers in the document. High-level section headers, which begin with numbers and lack function words, represent packet types. On the other hand, section headers without numbers and function words indicate packet fields and maintain a specific packet order. To parse each section header, we search for a colon separating the entity's name from its size and for commas or the word "and" indicating multiple entities within a single section header. This process generates a list of entities along with their sizes and order.

VIII. Ground-truth Labeling

Given the lack of an existing dataset for a similar protocol entity linking task, we manually label field-description pairs to establish the golden standard. To ensure accuracy in the labeling results, we form an inspection team consisting of four individuals. We establish criteria for each entity in our knowledge base, as shown in Figure 3, which we utilize for labeling field-description pairs. The labeling results from each team member undergo scrutiny from other group members. Only when full agreement is reached is a pair included in our dataset. In total, we collect 2858 valid pairs.

The distribution of samples for each feature is presented in Table 1.

IX. Introduction

This section presents a comparison of different implementation methods of domain models based on the following metrics: serial number, boolean, identifier, version, checksum, encryption data, padding code, length, timestamp, offset, data, and reserved. These metrics are commonly used in the field and provide valuable insights into the performance of the models.

X. Training

A. Model Configuration

All of our models (RFC-BERT-a, RFC-BERT-b, RFC-BERT-c) consist of 12 transformer blocks, 768 hidden units, and 12 self-attention heads. For RFC-BERT, we initialize it using BERT_{BASE} and then fine-tune the model for six epochs with a learning rate of $2e^{-5}$. During training and testing, the maximum text length is set to 10 tokens [42]. This limit is chosen because header fields often consist of short phrases in RFCs.

TABLE I
COMPARISON OF DIFFERENT IMPLEMENTATION METHODS OF DOMAIN MODELS

Serial Number	270
Boolean	290
Identifier	175
Version	520
Checksum	370
Encryption Data	61
Padding Code	300
Length	100
Timestamp	90
Offset	170
Data	302
Reserved	210

B. Baselines

In this subsection, we compare our method against several baselines including SVM [27], BPNN [36], CNN [8], Bi-GRU [9], and Adhikari et al [2]. We carefully tune the hyperparameters of the baselines to ensure fair comparisons. All the baselines take the word embeddings of the header and description as their inputs.

Adhikari et al [2] proposed a classification model based on a joint network of Bi-LSTM and Max-Pooling. We include this model as one of our baselines, as well as its succeeding model based on BERT.

1) SVM [27]: We train the SVM model using stochastic gradient descent. The constraints are adjusted using L2 Regularization, and the margin is set to 1.0. Other parameters are initialized randomly.

2) BPNN [36]: All parameters of the BPNN model are initialized randomly. We set the dropout rate to 0.1 to avoid overfitting and adopt an adaptive gradient descent strategy during training.

3) CNN [8]: We use three kernels during the convolution process, each with a size of 3 * 768. The size of the kernels in the max-pooling phrase is 2 * 2. The batch size is set to 1, and the dropout rate is 0.1. The output is sent into a linear layer and a softmax layer for predictions.

4) Bi-GRU [9]: Bi-GRU acts similarly to the memory cell in the LSTM network. All parameters are initialized randomly. We concatenate the output and use the same linear and softmax layers as the CNN model for post-processing the output.

5) Adhikari et al [2]: The model takes word embeddings as its input. We concatenate the outputs into a linear layer and a softmax layer. The size of the kernels in the max-pooling phrase is 2 * 2, and the dropout rate is set to 0.1.

To assess the impact of header fields on the experiment, all the baselines do not consider header information. Additionally, we use 8000 iterations to approximate the number of epochs in BERT for SVM, BPNN, CNN, and Bi-GRU. We set the learning rate for all the baselines to $2e^{-2}$ for faster convergence.

XI. EVALUATION AND ANALYSIS

In this section, we provide an evaluation of our model and analyze the results. We report the accuracy (Acc), average precision (Avg_P), average recall (Avg_R), and average F1-value (Avg_F) for all categories in the schema using 10-fold Cross Validation.

A. Evaluation Metrics

Acc represents the accuracy of the model on the training set. Avg_P, Avg_R, and Avg_F represent the average precision, recall, and F1-measure across all categories in PKB, respectively.

To calculate these metrics, we use the following formulas:

$$Acc = \sum_{a=1}^{C} \frac{TP(a)}{N} \tag{5}$$

$$Avg_P = \sum_{a=1}^{C} \frac{TP(a)}{TP(a) + FP(a)} \tag{6}$$

$$Avg_R = \sum_{a=1}^{C} \frac{TP(a)}{TP(a) + FN(a)} \tag{7}$$

$$Avg_F = \frac{2 \times Avg_P \times Avg_R}{Avg_P + Avg_R} \tag{8}$$

Here, $TP(a)$, $FP(a)$, and $FN(a)$ represent the true positives, false positives, and false negatives, respectively, for category a. C is the total number of categories, and N is the total number of samples.

B. Comparison on Different Implementations of RFC-BERT

We evaluate RFC-BERT using three different non-linear layers: BPNN [36], CNN [8], and Bi-GRU [9]. The statistics are shown in Table IV.

The results show that RFC-BERT based on Bi-GRU achieves the best performance on our dataset, achieving the highest accuracy of 72.9% and highest F1-value of 74.2%. However, since CNN is not able to capture contextual information effectively, its performance is inferior to that of BPNN and Bi-GRU. Bi-GRU, being derived from RNN, is better suited for handling segmented information. Therefore, we choose Bi-GRU as the RFCDomainModel in RFC-BERT.

TABLE II
THE PERFORMANCE ACHIEVED BY DIFFERENT APPROACHES

2*Exp. Group	Model	Performance				2*Learning Rate	
		Acc	Avg_P	Avg_R	Avg_F		
3*Our Approach	**RFC-BERT-a (BPNN)**		**72.4%**	**73.9%**	**74.3%**	**74.1%**	**2e^{-5}**
	RFC-BERT-b (CNN)		**49.6%**	**51.3%**	**53.0%**	**52.1%**	**2e^{-5}**
	RFC-BERT-c (Bi-GRU)		**72.9%**	**73.7%**	**74.7%**	**74.2%**	**2e^{-5}**

TABLE III
THE RESULTS OF ABLATION STUDY

Exp. Group	Model	Acc	Avg_P	Avg_R	Avg_F	Learning Rate
Baselines	**SVM [27]**	10.8%	10.4%	10.8%	10.6%	2e^{-2}
	BPNN [36]	55.8%	47.8%	48.7%	48.2%	2e^{-2}
	CNN [8]	48.0%	44.5%	45.2%	44.8%	2e^{-2}
	Bi-GRU [9]	53.6%	44.9%	41.3%	43.0%	2e^{-2}
	Adhikari [2]	57.6%	48.3%	48.3%	48.3%	2e^{-2}
Our Approach	**RFC-BERT**	**72.9%**	**73.7%**	**74.7%**	**74.2%**	2e^{-5}

TABLE IV
COMPARISON OF DIFFERENT IMPLEMENTATIONS OF RFC-BERT

Exp. Group Avg_F	Model Learning Rate	Acc	Avg_P	Avg_R
$BERT_{only}$ 72.4%	Fine-tuned BERT 2e^{-5}	69.8%	72.7%	72.2%
3*$RFCDomain_{only}$ 49.4%	BPNN 2e^{-5}	47.3%	50.2%	48.5%
 34.2%	CNN 2e^{-5}	35.5%	33.5%	34.9%
 64.1%	Bi-GRU 2e^{-5}	60.8%	64.1%	64.1%
1*$BERT+Domain_{Joint}$ 74.2%	RFC-BERT-c (Bi-GRU) 2e^{-5}	72.9%	73.7%	74.7%

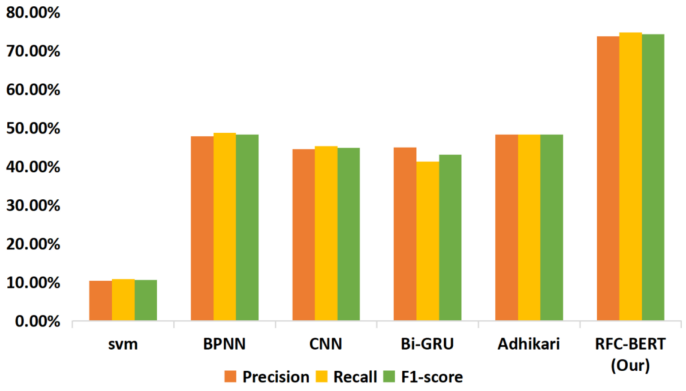

Fig. 5. Performance achieved by different approaches

C. Comparison with Baselines

Our approach, RFC-BERT, achieves the best performance, as demonstrated by the statistics presented in Table II and Figure 5. The Precision, Recall, and F1-score are **73.70%**, **74.70%**, and **74.20%**.

D. Analysis

We can draw an analogy between language models pre-trained on generic corpora and human beings. These models achieve decent results for descriptive texts because they can encode contextual information. However, our experimental results illustrate that they fail to comprehend **Abstract Concepts** like *header field*. For example, the header field "IHL" (Figure 2) is an abstract concept in the protocol domain. Domain-specific information cannot be inferred from its lexical presentation. Thus, we are unable to link it to specific categories in the Protocol Knowledge Base (PKB). In our approach, we not only use domain knowledge to fine-tune BERT but also design a domain model to explicitly learn domain-specific knowledge from these header fields. Finally, we combine these two models in our RFC-BERT model, which utilizes BERT Model to capture the deep semantic information of the **Narrative Description** (such as the description chunk text of the RFC header field) and uses the domain model to learn the semantic information of **Abstract Concepts** (such as the RFC header field). The experimental results showed that our approach significantly outperformed the five baseline approaches.

XII. ABLATION STUDY

A. Evaluation of the Performance of the Joint Model

This set of experiments is designed to evaluate the impact that header fields have on classification, specifically whether the RFCDomainModel contributes to increasing F_{Avg}. The results are shown in Table III.

The accuracy of fine-tuned BERT is 69.8%. The individual RFC Domain Model also achieves inferior results. However, when the RFCDomainModel is combined with BERT, the accuracy reaches 72.9%, which is higher than using only BERT or the RFCDomainModel independently. This suggests that the

fusion of the domain model and the fine-tuned language model can improve the performance by incorporating domain-specific knowledge. Therefore, we choose RFC-BERT-c (Bi-GRU) as our final model.

B. Using the RFC Domain Model to Handle Header Fields

Given the fact that BERT is suitable for processing texts with sequential relations, appending header fields to the descriptions will introduce false information, leading BERT to wrongly associate the header fields and descriptions appearing in adjacent contexts. This hinders BERT's performance. Therefore, directly concatenating header fields to descriptions is not appropriate. As a result, we do not solely rely on applying BERT to our entity linking task with header fields concatenated with descriptions as input.

C. Using Neural Network to Handle Header Fields

We consider using neural networks, such as Bi-GRU and BPNN, as the domain model to handle header fields independently, rather than a Pre-training Model. This is because our RFC dataset is much smaller compared to the large corpora that BERT is pre-trained on. The heuristic knowledge acquired by BERT from generic corpora largely compromises the valuable domain-specific information contained in our dataset. By training a model explicitly for header fields, we can fully exploit the domain knowledge contained in them.

XIII. Application

In this paper, we deploy our RFC-BERT model to perform the entity linking task on over 8000 RFCs. We extract all the header fields and feed the resulting RFC triplet data into the Graph Database neo4j[3] to construct the domain-specific knowledge graph of RFCs, RFC-KG, which contains 29,880 entities and 5,409,158 relationships.

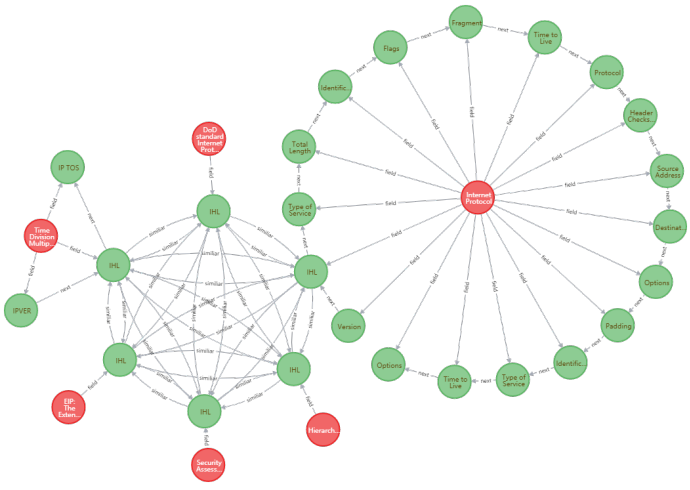

Fig. 6. Correlation analysis of header fields in RFC 791 (Internet Protocol), RFC: red color, Header Fields of RFC: green color

[3]https://github.com/neo4j/neo4j

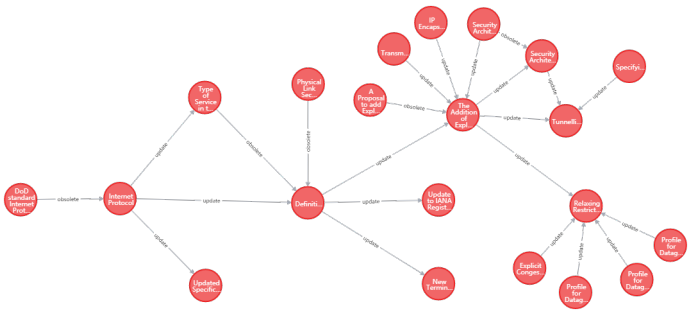

Fig. 7. The evolution of RFC 791 (Internet Protocol), RFC: red color

XIV. Conclusion and Future Work

In this paper, we propose a joint model to automatically extract protocol entities from RFCs and link them to PKB. The experimental results demonstrated that our approach significantly outperformed five baseline approaches with an average precision, recall, and F1-score of *73.70%, 74.70%, 74.20%*, respectively. By deploying the RFC-BERT model, we were able to generate the domain-specific knowledge graph of RFCs, referred to as RFC-KG. In the future, we plan to optimize the RFC-BERT model and evaluate the effectiveness of RFC-KG in real-world applications.

XV. Acknowledgments

References

[1] Titipat Achakulvisut, Chandra Bhagavatula, Daniel Ernesto Acuna, and Konrad P. Körding. Claim extraction in biomedical publications using deep discourse model and transfer learning. *ArXiv*, abs/1907.00962, 2019.

[2] Ashutosh Adhikari, Achyudh Ram, Raphael Tang, and Jimmy Lin. Rethinking complex neural network architectures for document classification. In *NAACL-HLT*, 2019.

[3] Alberto Bartoli, Andrea De Lorenzo, Eric Medvet, and Fabiano Tarlao. Active learning of regular expressions for entity extraction. *IEEE Transactions on Cybernetics*, 48:1067–1080, 2018.

[4] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA, 2008. Association for Computing Machinery.

[5] Pat Bosshart, Dan Daly, Glen Gibb, Martin Izzard, Nick McKeown, Jennifer Rexford, Cole Schlesinger, Dan Talayco, Amin Vahdat, George Varghese, and David Walker. P4: programming protocol-independent packet processors. *Computer Communication Review*, 44:87–95, 2014.

[6] Samuel Broscheit. Investigating entity knowledge in bert with simple neural end-to-end entity linking. *ArXiv*, abs/2003.05473, 2019.

[7] Aditi Chaudhary, Jiateng Xie, Zaid Sheikh, Graham Neubig, and Jaime G. Carbonell. A little annotation does a lot of good: A study in bootstrapping low-resource named entity recognizers. In *EMNLP/IJCNLP*, 2019.

[8] Yahui Chen. Convolutional neural network for sentence classification. 2015.

[9] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *ArXiv*, abs/1406.1078, 2014.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

[11] Matthew Francis-Landau, Greg Durrett, and Dan Klein. Capturing semantic similarity for entity linking with convolutional neural networks. *ArXiv*, abs/1604.00734, 2016.

[12] S. Garg, A. Garg, A. Kandpal, K. Joshi, R. Chauhan, and R. H. Goudar. Ontology and specification-based intrusion detection and prevention system. In *Confluence 2013: The Next Generation Information Technology Summit (4th International Conference)*, pages 154–159, Sep. 2013.

[13] Pablo Hernandez-Leal, Alban Maxhuni, Luis Enrique Sucar, Venet Osmani, Eduardo F. Morales, and Oscar Mayora-Ibarra. Stress modelling using transfer learning in presence of scarce data. In *AmIHEALTH*, 2015.

[14] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust disambiguation of named entities in text. In *EMNLP*, 2011.

[15] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *ACL*, 2018.

[16] Samuel Jero, Maria Leonor Pacheco, Dan Goldwasser, and Cristina Nita-Rotaru. Leveraging textual specifications for grammar-based fuzzing of network protocols. In *AAAI*, 2019.

[17] Assadarat Khurat and Wudhichart Sawangphol. An ontology for snort rule. *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pages 49–55, 2019.

[18] Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. End-to-end neural entity linking. In *CoNLL*, 2018.

[19] Maxime Labonne, Alexis Olivereau, Baptise Polvé, and Djamal Zeghlache. Unsupervised protocol-based intrusion detection for real-world networks. *2020 International Conference on Computing, Networking and Communications (ICNC)*, pages 299–303, 2020.

[20] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6:167–195, 2015.

[21] Ying Lin, Chin-Yew Lin, and Heng Ji. List-only entity linking. In *ACL*, 2017.

[22] Wei-Yin Loh. Classification and regression trees. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, 1:14–23, 2011.

[23] Mohammad Lotfollahi, Ramin Shirali Hossein Zade, Mahdi Jafari Siavoshani, and Mohammdsadegh Saberian. Deep packet: a novel approach for encrypted traffic classification using deep learning. *Soft Computing*, pages 1–14, 2020.

[24] Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. Joint learning of named entity recognition and entity linking. In *ACL*, 2019.

[25] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.

[26] H. Mosteghanemi and H. Drias. Bees swarm optimization for real time ontology based information retrieval. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 154–158, 2012.

[27] Edgar Elias Osuna, Robert M. Freund, and Federico Girosi. Support vector machines: Training and applications. 1997.

[28] Mark Palatucci, Dean Pomerleau, Geoffrey E. Hinton, and Tom Michael Mitchell. Zero-shot learning with semantic output codes. In *NIPS*, 2009.

[29] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[30] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *ArXiv*, abs/1802.05365, 2018.

[31] Yun Qu, Hao Zhang, Shijie Zhou, and Viktor K. Prasanna. Optimizing many-field packet classification on fpga, multi-core general purpose processor, and gpu. *2015 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, pages 87–98, 2015.

[32] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *ArXiv*, cs.CL/0306050, 2003.

[33] Lin Shi, Celia Chen, Qing Wang, Shoubin Li, and Barry W. Boehm. Understanding feature requests by leveraging fuzzy method and linguistic analysis. *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 440–450, 2017.

[34] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *CCL*, 2019.

[35] Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. Modeling mention, context and entity with neural networks for entity disambiguation. In *IJCAI*, 2015.

[36] Hemlata Tekwani and Mahak Motwani. Text categorization comparison between simple bpnn and combinatorial method of lsi and bpnn. *International Journal of Computer Applications*, 97:15–21, 2014.

[37] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, 2001.

[38] Matteo Varvello, Rafael P. Laufer, Feixiong Zhang, and T. V. Lakshman. Multilayer packet classification with graphics processing units. *IEEE/ACM Transactions on Networking*, 24:2728–2741, 2016.

[39] Lingyong Yan, Xianpei Han, Le Sun, and Ben He. Learning to bootstrap for entity set expansion. In *EMNLP/IJCNLP*, 2019.

[40] Weixin Zeng, Jiuyang Tang, and Xiang Zhao. Entity linking on chinese microblogs via deep neural network. *IEEE Access*, 6:25908–25920, 2018.

[41] Weixin Zeng, Xiang Zhao, Jiuyang Tang, and Haichuan Shang. Collective list-only entity linking: A graph-based approach. *IEEE Access*, 6:16035–16045, 2018.

[42] Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. Bridging the gap between training and inference for neural machine translation. In *ACL*, 2019.

[43] Huangjie Zheng, Yuchen Wang, Chen Han, Fangjie Le, Ruan He, and Jialiang Lu. Learning and applying ontology for machine learning in cyber attack detection. *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 1309–1315, 2018.