Building a Hybrid Model for Affected Vulnerable Entity Recognition Using BERT

Xiaotian Lu, Tao Du

***Abstract*— Common Platform Enumeration (CPE) is an enumeration division for product versions. CPE is associated with the Common Vulnerabilities and Exposures (CVE) released by National Vulnerability Database(NVD), so that the version enumeration set of all products affected by the specified vulnerability can be obtained.Automatically identifying the names of entities affected by vulnerabilities (extracting product, vendor) names facilitates the response to new cybersecurity threats and reduces the risk of attacks on related entities.In view of the latest released CVE data, this paper proposes a deep learning-based security vulnerability summary entity identification method, which identifies and labels two entities, vendor and device model, from the CVE summary.Experiments show that this method significantly improves the F1 value and recall rate of entity recognition in the vulnerability data provided by NVD.The results of this paper can also provide a reference for the automatic generation of CPE, and the affected suppliers can locate the security vulnerabilities in their products as soon as possible according to the CPE.**

***Index Terms*— Named Entity Identification, Vulnerability Information, Feature Extraction, Recurrent Neural Networks, BERT.**

I. INTRODUCTION

Despite the constant disclosure of vulnerability threats, common cyberattacks still pose a threat to some important enterprises or departments, because these organizations still rely on traditional methods to manage software, making updates and maintenance always lagging behind the release of vulnerabilities[1].

Among the many incidents that threaten network security, the scope of software vulnerabilities is very wide, which can directly affect all users who use the relevant software, even developers.In the process of software development, the use of third-party component software libraries has become an important means to improve development efficiency. If the third-party software used by developers has vulnerabilities, the security of the entire project will be affected.To reduce the security threats caused by software vulnerabilities, it is necessary to monitor and manage the third-party components in the project, which usually relies on the automatic software composition analysis (SCA) technology based on the open source vulnerability database.

Automated vulnerability matching requires standardized vulnerability data.Taking NVD as an example, it assigns a unique CVE number to each included vulnerability and includes a short summary describing the vulnerability.A study [2] pointed out that in 2018, the disclosure time of CPE lags behind CVE by an average of 35 days.

Using the semantic information of the vulnerability description summary text in the CVE, the product and vendor information of the entity affected by the vulnerability can be extracted.The core step of identifying the entities affected by the vulnerability is to extract the entity fields required to constitute the CPE from the textual information provided by the CVE.Previously, building a dictionary of mappings between vulnerabilities and CVEs was a common solution.A single mapping relationship cannot identify entities that have not appeared before, so the comprehensive recognition rate of the method of constructing a dictionary is not high, and manual proofreading is required in the later stage to achieve a better recognition effect.In recent years, machine learning has been widely used in classification tasks. [3] used the SVM model to classify the vulnerability text, and extracted the vulnerability-related text by training an SVM classifier.

In the current research, the use of deep learning methods to deal with complex texts has achieved relatively good results.As models such as bidirectional long short-term memory network and random conditional field are widely used in text recognition scenarios, the accuracy of entity recognition tasks in specific domains has been improved to a certain extent.One of the main advantages of sequence model-based natural language processing (NLP) methods is the ability of algorithms to understand the context of text. This makes the model no longer rely on a single word for classification, but has the ability to understand the context, so as to deal with complex text more accurately.For example, the application of models such as bidirectional long short-term memory network BLSTM and random conditional field CRF can improve the accuracy of entity recognition tasks in specific domains.However, the performance of such methods is largely dependent on feature selection.Feature vectorization models trained on regular corpora often fall short of expectations when dealing with text in the field of security vulnerabilities.This paper embeds the domain knowledge of security vulnerabilities into a large-scale pre-trained language model, and combines multiple text sequence processing algorithms to construct a hybrid model.Through transfer learning, the pre-trained model can extract high-quality features in more dimensions from the CVE to improve the overall performance of the model.When there is a CPE not given in the NVD database, the product and vendor information of the unrelated vulnerability-affected entities can be correctly classified through accurate identification of the relevant entities in the CVE summary.

The remainder content of the paper is organized as follows. Section 2 presents the related progress of vulnerability entity extraction work. Section 3 discusses the details and methodology of the proposed model for the identification of affected vulnerable entities. Sections 4 and 5 describe the experimental procedure in detail, and discuss and analyze the experimental results. Section 6 presents the conclusions and future related work.

Manuscript received November 13, 2024
Xiaotian Lu, Tiangong University, Tianjin, China
Tao Du, Tiangong University, Tianjin, China

II. Related Work

The work of extracting vulnerable entities has gone through multiple stages, from using simple mapping relationships to deep learning-based models, and methods using multi-feature parameters have shown significant advantages.

Building a mapping is the most intuitive solution.For example, in [4], a product-vendor dictionary is constructed by using the CPE with the corresponding relationship in the existing database, so as to find the product name mentioned in the vulnerability.This method can identify existing entities well, but cannot find entities that do not appear in the database.A single mapping relationship also cannot model complex situations.

Compared with dictionary mapping, a more comprehensive model can be built through machine learning.[5] proposes a classification method based on Naive Bayes to classify CVE entries with lack of information, and then determine their vulnerability categories.[2] proposes a TFIDF-based method to automatically extract the most likely affected software from newly disclosed zero-day vulnerability data to determine which systems are vulnerable.[6] classifies IoT device-related vulnerability data from public CVE/NVD databases, and trains an SVM classifier based on manually labeled data to achieve classification and prediction of new IoT device vulnerabilities.[7] builds a treebased FastXML algorithm to build machine learning models. The model is trained using vulnerability records and library names from NVD as training data to identify vulnerable software library names in XML format from NVD.[8] proposes a bootstrapping algorithm called PACE for NER in the field of vulnerability security. This algorithm was shown to be suitable for correctly extracting entities in smaller corpora.The machine learning method can be used to classify and identify vulnerable entities.However, when dealing with long vulnerability summaries, traditional machine learning models cannot model semantics, which reduces the recognition accuracy.In contrast, the hybrid model proposed in this paper can learn both the contextual and overall semantics of vulnerability summaries.Affected entity information can also be accurately identified when encountering long texts.

[9] proposed an automated system using a serialized text processing model to reduce the use of manual annotations by detecting inconsistencies between CVE descriptions and their cited vulnerability reports.[10, 11] construct feature engineering based on a large number of labeled corpora and train CRF classifiers to complete the identi-fication of vulnerability-related entities.[12] proposes a hybrid model TFI-DNN, which uses TFIDF to calculate word frequency information as the feature input of DNN network for automatic vulnerability classification.[13] proposed a document-level encoder based on Bidirectional Encoder Representations from Transformers (BERT), which achieved good results by encoding between documents, extracting corresponding features and using them for text extraction.

It is worth noting that some progress has been made in the application of BLSTM-CRF model to NER work in the field of security vulnerabilities. [14] selects common text features, uses multiple feature vectors spliced as the input of BLSTM, uses CRF to decode the output sequence and filter the prediction results to obtain the predicted label.The model achieves an F1 value of 86 %, but it uses a general pre-training model for feature vectorization, which will inevitably lead to missing features. BERT pre-training model can extract deep inter-word features, grammatical features, and contextual semantic features from the input sequence, providing high-quality word vectors.

Therefore, on the basis of previous research, this paper improves the feature engineering part, combines BERT and BLSTM-CRF model, and proposes a method of using BERT-BLSTM-CRF model to identify affected vulnerability entities.BERT is used to learn various features of text, and the semantic information is vectorized to the maximum extent as the feature input of BLSTM, thereby improving the overall accuracy of the model.

III. Affected Vulnerable Entity Identification

A. Problem Definition

Relevant companies or organizations usually associate CPEs for CVEs on a regular basis.Automated software management systems can use CPE to identify and warn of software affected by vulnerabilities.Reducing the time from vulnerability disclosure to correlation identification enables more timely processing of security-risk software.Therefore, improving the efficiency of CPE-associated CVE is the key.The purpose of this paper's proposed hybrid model is to:(1)Improve the efficiency of vulnerability entity identification. By automatically identifying affected vulnerability entities, the association process is less reliant on manual proofreading;(2)Improve the accuracy of vulnerability entity recognition. The CVE contains a large amount of complex text, and more targeted features need to be obtained from it to correctly identify the two types of entities, supplier and product.

B. Data and Labels

The CVE and CPE data used in this article come from NVD, which is obtained through the cve-search and imported into MongoDB, shown in 1.Local lookups are faster and more private than looking up public CVE databases directly.The CVE retrieved through the API provided by cve-search is in dictionary format, and different fields in the CVE are saved as required.

Among them, "description" is an overview of the overall vulnerability in the CVE, called the vulnerability summary.Use Natural Language Toolkit (NLTK) to segment the vulnerability summary to get the vulnerability corpus.

CPE can be regarded as a standardized expression of CVE, and there is a one-to-one correspondence between the two.The format of CPE is fixed, and its composition is as follows:

cpe : / < part >:< vendor >:< product >:< version >:< update >:< edition >:< language >

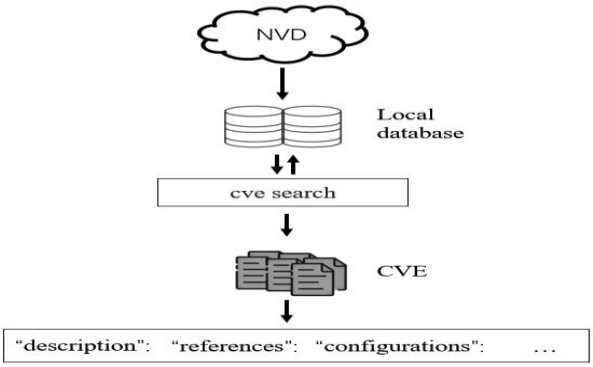

Fig. 1. The process of obtaining CVE-related field data

Table 1
Annotated summary with BIO

Summary:	Label:
Microsoft	B-VENDOR
Windows	B-PRODUCT
XP	I-PRODUCT
may	O
crash	O
when	O

Some fields in the CPE may overlap with entities in the vulnerability summary. For example, <Microsoft> as a vendor may appear in both CVE and CPE related to it, which is the basis for the association between CVE and CPE.

Sequence annotations are roughly divided into two categories: original annotations and joint segmentation markers, the latter being more common in practical applications.For example, when a named entity like "windows server 2008" appears in a sentence, it should be treated as a whole.Each affected vulnerability entity is labeled using the BIO rule [15], transforming the joint segmentation problem into the original labeling problem, which can ensure both the integrity of the entity and the efficiency of data processing.

In this paper, BIO rules are used to define tags, and five types of tags are obtained: B-VENDOR, I-VENDOR, BPRODUCT, I-PRODUCT, O.Labels are divided into B labels and I labels, which represent the start and end of a type of entity respectively, and O labels represent unrecognized entities. Thus, a piece of abstract text can be represented in the form in Table 1:

The CoNLL-2003 dataset is a classic English text dataset that has released many shared tasks including English NER.The training dataset for the NER task consists of three files, the training data file eng.train and two data files testa, testb for validation and testing.The eng.train format is shown in Figure 2

The file contains four columns of data, the first column is the single word obtained by dividing the original text by spaces, the second column is the part-of-speech tag, the third column is the grammar block tag, and the fourth column is the named entity tag.

There is a big difference between the vulnerability summary corpus and the existing dataset corpus, so it is necessary to build a security-oriented dataset for model training.Referring to the form of the CoNLL-2003 dataset,the self-built dataset also contains three files, namely the training set train, the validation set val and the test set test.

In specific tasks, no need to consider features at the grammar block tagging and part-of-speech level.Therefore,the self-built dataset consists of only two columns, the first column is the words that are separated by spaces for the vulnerability corpus, and the second column is the custom labels according to the BIO rules.The specific labels are:Vendor VENDOR, Product PROCUDT, and Unidentified O.

Germany NNP B-NP B-LOC
imported VBD B-VP O
47 · 600 CD B-NP O
sheep NN I-NP O
from IN B-PP O
Britain NNP B-NP B-LOC
last JJ B-NP O
year NN I-NP O
· · O O
nearly RB B-NP O
half NN I-NP O
of IN B-PP O
total JJ B-NP O
imports NNS I-NP O
. . O O

Fig. 2. CoNLL-2003 dataset format.

C. Modeling

In recent years, corpus preprocessing has been a hot research topic in the field of NLP. BERT [16], a new language model proposed by Google based on the Transformer model,provides higher-quality word feature vectors for downstream tasks by training large-scale parameters, thereby improving the accuracy of entity recognition and classification.The model is inspired by the work of [14] on feature extraction of vulnerable NER text.The model is inspired by the work of [15] on feature extraction of vulnerable NER text.Its main contribution is to use a largescale pre-trained language model for semantic feature extraction of vulnerable texts by means of transfer learning,thereby obtaining high-quality deep word vectors.These vectors containing a series of vulnerable text features are fed into the BLSTM for sequence labeling.The overall structure of the BERT-BLSTM-CRF model proposed in this paper is shown in Figure3. The model is mainly composed of two parts: the BERT module and the BLSTM-CRF module.First, the annotated vulnerability summary is converted into a vector form as the input data of the BERT pre-training model, and the output of the hidden layer of the last layer of the model is the word vector containing various semantics in the vulnerability corpus.Then the word vector is input to the BLSTM-CRF module for further processing.BLSTM will further extract the relationship between various features in the sequence, and finally use CRF to perform Viterbi decoding on the output value of BLSTM to obtain the predicted label sequence.Extracting and classifying the entities in the sequence can get different entity labels, and then identify the affected vulnerable entities from the CVE.

Before using the traditional Recurrent Neural Network (RNN) model to label text, the corpus is usually vectorized to form an input sequence and then further processed. The number of corpus features and the quality of vectorization will directly affect the final downstream task. For text annotation tasks in specific fields such as vulnerability entity recognition, the model trained on general corpus will cause the loss of text features in the process of word vectorization, resulting in the model not reaching the expected performance.Compared with the training text used by the mainstream word embedding pre-training model, the semantics of the security domain corpus is too large,resulting in the low quality of the obtained word embedding corresponding to the vulnerability text, which in turn affects the final results of downstream tasks.

The features learned by each layer in the BERT model are different, so the BERT model can obtain various features of

the sentence to form a comprehensive feature vector containing sentence context information[17].For example, some vendors require a unified text format for vulnerability disclosure, and learning text semantics can effectively distinguish such vulnerability information. Compared with the single contextual information obtained by the traditional RNN model, the information extraction ability of BERT is significantly stronger.Due to the introduction of the BERT preprocessing model, the model in this paper can directly perform multi-level feature extraction and semantic learning on the input vulnerability text sequence.This reduces the prediction error caused by context inconsistency to a certain extent, and further improves the accuracy and generality of the named entity recognition task in the security field.

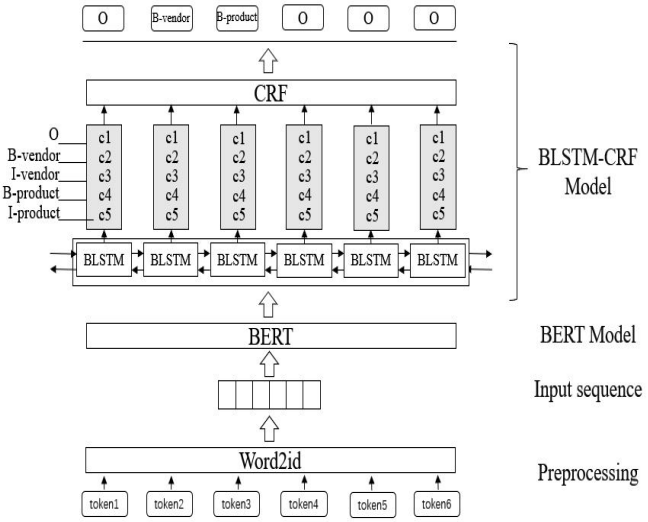

Fig. 3. The overall structure and data processing flow of the BERT-BLSTM-CRF model.

At present, the Transformer model has been widely recognized in the field of natural language processing. By fine-tuning the pre-trained language model to adapt to specific downstream tasks, such as classification, prediction and labeling tasks, the performance of the model can be effectively improved.

The core of BERT is the bidirectional Transformer encoding structure, which processes all words in parallel,which is the biggest difference from RNN.The Transformer uses Positional Encoding [16] to obtain the sequential information of the language, and models the text by introducing a Self-Attention mechanism.

Self-attention mechanism is the core part of Transformer encoder.

$$SelfAttention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

In the above formula, QK^T is the attention matrix, which is used to weight V so that each word contains the information of all other words in the current sentence.Under the self-attention mechanism, the correlation weight between vulnerable entities will be higher than the correlation weight between vulnerable entities and non-vulnerable entity texts.For example, in the attention matrix, the weight between "Microsoft" and "Windows XP" will be higher than the weight between "Microsoft" and "release".As a result, the vulnerability entities annotated as VENDOR and PRODUCT can be better distinguished from other texts.Furthermore, the self-attention mechanism also has advantages in distinguishing vulnerable entities composed of multiple words.For example, in the entity "Windows 10 Enterprise", "Windows" is labeled B-PRODUCT and the remaining two are I-PRODUCT.Although there may be cases where "Windows" and the latter two are closely weighted, in the attention matrix, the weight between "10" and "Enterprise" will be significantly higher than the weight between "Windows" and them.From this, the different components of the multi-word entity can be correctly distinguished.

BERT does not use the shallow splicing method in previous models for pre-training, but introduces two pretraining tasks, Masked Language Model (MLM) and Next Sentence Prediction (NSP) to build language models.This enables BERT to generate deep bidirectional language representations [18].

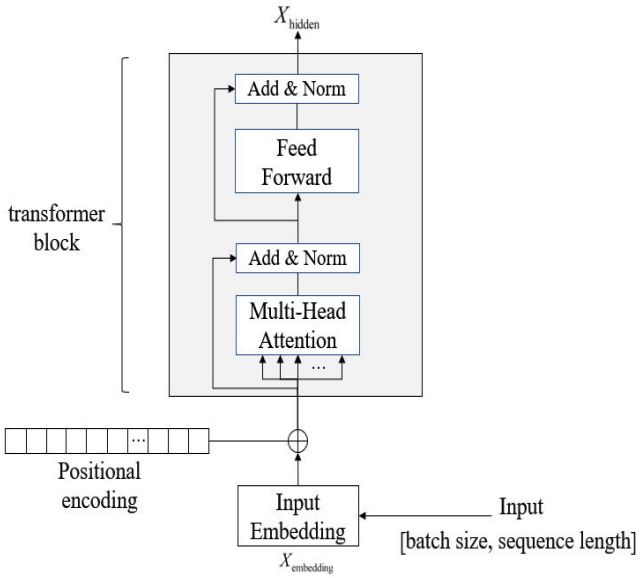

Fig. 4. Single Transformer Encoder Structure.

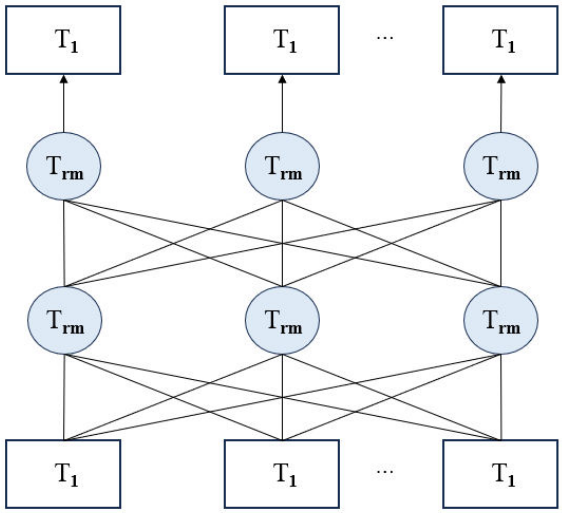

Fig. 5. Bidirectional structure in BERT.

Long Short Term Memory Network (LSTM) [19] is an improved RNN, relying on its unique "gate" structure, the network can theoretically effectively capture the association information in long sequences.Using BLSTM as part of the hybrid model is beneficial to further capture the context of vulnerability text data.The LSTM network solves the problems of vanishing gradient and exploding gradient in the RNN structure by introducing more states [20].An LSTM cell consists of a forget gate, an input gate, an output gate, and a

cell state.Figure X is the LSTM input-output relationship obtained by expanding the time dimension

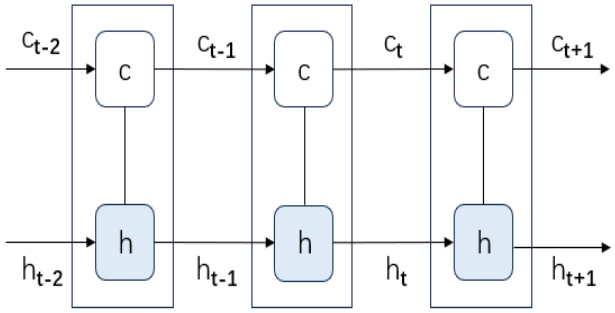

Fig. 6. LSTM sequence structure.

As shown in Figure X, the LSTM has three inputs at time t: the input value x_t obtained by the network at the current time, the unit state c_{t-1} at the previous time, and the output value h_{t-1} of the LSTM at the previous time.There are two outputs at this time: the output value h_t of the LSTM and the current cell state c_t.

LSTM can only process information in one direction, and cannot encode the feature information of text sequences from back to front.Bidirectional Long-Short Term Memory (BLSTM) network was proposed in [?], using two independent and identical LSTM layers to form a bidirectional structure, each layer passing the input sequence in its own direction.When dealing with classification problems that require more fine-grainedness, BLSTM can better capture bidirectional contextual semantic dependencies.For example, part of the vulnerability summary might read "The defective device is the A2021 and other products in the same series". BLSTM can capture the semantics of "and other products in the same series" and use it as a basis for identifying the corresponding vulnerability entity.

Conditional Random Field (CRF) is a probabilistic model for labeling and segmenting structured data such as sequences, trees, and grids [21].CRF can make constraints on the output of the model to reduce the weight of wrong predictions, which is equivalent to setting some prior specifications for the prediction behavior.For example: the label corresponding to the starting word of the vulnerability entity should be "B-" instead of "I-"; it is judged that the label "O I-label1" is wrong, and so on.This ensures the reliability of the final forecast results.The transition matrix A is a parameter of the BLSTM-CRF model. The CRF layer can obtain a transition score by learning the transition matrix, which represents the probability of transitioning from one label to another.The CRF layer can learn the preand post-dependency constraints of the sentence, and comprehensively use the tag state transition probability to get the score.For example, in a vulnerability summary, if the previous word is labeled "B-VENDOR", the probability that the current word is labeled "I-VENDOR" increases.

In summary, the calculation process of the final prediction result of the BLSTM-CRF model is as follows

$$\text{score}(x, y) = \sum_{i=1}^{n} P_{i,y_i} + \sum_{i=1}^{n+1} A_{y_{i-1},y_i}$$

For the entire sequence, the overall score for sequence annotation is equal to the sum of the scores for each position.$P_{i,\,yi}$ represents the label score with the subscript y_i of the LSTM output, and $A_{yi-1,yi}$ represents the probability score of transferring from the label with the subscript y_{i-1} to the label with the subscript y_i.

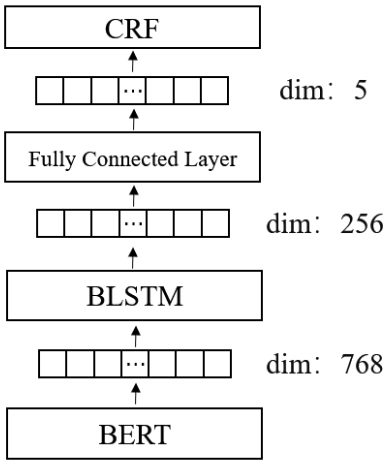

Fig. 7. Changes in data dimensions in each layer of the model.

The MLM language model is built in BERT.MLM can randomly cover or replace a certain word or word in the vulnerability summary, allowing the model to predict the covered or replaced part by understanding the context to better extract the affected vulnerability entity based on semantic information.This article uses the BIO rule to label the sequence, so when performing MASK, the token with the label label is used as the minimum unit to generate the mask.For the vulnerable corpus composed of multi-sentence texts, the NSP task is enabled to identify the relationship between sentences, and some tokens that constitute complete entities are added to two consecutive or random sentences, so that the model can predict whether there is a contextual relationship between the two sentences.The maximum sequence length that BERT can process is 512 (identifiers such as [CLS], [SEP] are not counted). For vulnerable corpora with a length of more than 512, the following part without entity tags is discarded and the text with a length of 512 is retained.If the entity appears at the end of the corpus, the text with a length of 512 is cut forward from the entity part, and the rest is discarded.

The BERT used in this paper is bert-base-cased, which consists of a twelve-layer network structure.In the lowlevel network, it focuses on acquiring phrase information, the middle-level network tends to learn grammatical information, and the high-level network learns semantic information.The extraction of text features by BERT is progressive, and the high-level network can learn deep-level composite features that integrate various features in the low-level network.Vulnerability corpus is essentially different from ordinary text corpus in terms of usage.Phrase information and English grammar information are relatively weakened, and various entities become more important information.Entity information and phrase information partially overlap, and the text that modifies the entity also needs to be grammatically learned. Therefore, the comprehensive extraction of the semantic features of the vulnerability corpus can cover the entity information in the vulnerability corpus to the greatest extent.In summary, this paper selects the output of the hidden layer of the last layer of

the BERT network as the result of BERT's feature vectorization of the vulnerability corpus.
In terms of parameter adjustment, BERT has been pre-trained based on large-scale text, so it is only necessary to perform partial training on the vulnerability corpus based on the original pre-training parameters.The updated part of the parameters can be regarded as supplementing the domain knowledge of security vulnerabilities into BERT,making the model more targeted.
The model in this paper works in an end-to-end manner, so the output of BERT is input to subsequent modules as a hidden layer in the overall BERT-BLSTM-CRF model.

Table 2
Training environment

OS	Linux
CPU	Intel Xeon E5-2680 v4 @ 2.40GHz
GPU	NVIDIA GTX1080ti
RAM	32 GB
Python	3.69
Pytorch	1.7.1

IV. Experiment

The experimental code is based on Pytorch, and the specific training environment configuration is shown in Table2
The Transformer part of the model uses the English pre-trained language model bert base model (cased) provided by Google, the scale is base, and it contains the features of word case information.bert-base-cased has a 12-layer structure, the hidden layer dimension is 768 dimensions, and the maximum sequence length is 128.
The multi-head attention mechanism is one of the core mechanisms of Transformer, but too many attention heads will lead to an increase in model parameters, which will affect the effect of the model when it is used for prediction [22].Try to train the model when the number of attention mechanism heads h is 3, 6, 9, and 12, respectively. In the case of considering the comprehensive training efficiency, h is set to 12.batch size is the size of a batch during training, which determines the number of samples input into the model in parallel.Referring to the work of Keskar et al. [23], the batch size is set to 24 after the comprehensive training time and loss converge.In the RNN part, the hidden layer dimension of LSTM is set to 128, the output layer dimension is 256, and the input and output parts use Dropout with a value of 0.5.
In the experiments, the following models are used for comparative experiments to demonstrate the superiority of the BERT-BLSTM-CRF model in vulnerability entity recognition
(1) BERT This model is an improvement of the Transformer language model. The Transformer encoder is formed by the bidirectional Transformer component so that the model can generate deep bidirectional text representations.Trained using a sequence of vulnerable text as model input.
(2) BLSTM-CRF This model is a classical model of sequence labeling based on RNN. The vulnerability corpus is converted into a vector representation using the glove pre-trained word embedding model [24],which is concatenated with the upper and lower case features and character-level features in the vulnerability text as input for training BLSTM.

There are certain changes in the naming rules and formats of CVEs released in different periods.In general, the CVE format with a recent release date is more standardized.This paper uses 100,000 original vulnerability CVEs from NVD before March 1, 2022, extracts the summary part and performs data cleaning and normalization to form a data set with 3,758,945 words.In this paper, the BIO three-dimensional notation method is used for sequence labeling, where "B" represents the initial part of the entity, "I" represents the rest of the entity, and "O" represents the non-entity part.A total of 5 tags are defined using this method, namely: "B-PRODUCT", "I-PRODUCT", "BVENDOR", "I-VENDOR", "O".
There are two training methods for BERT: updating all parameters of the model during training and updating only part of the parameters during training.In order to use the parameters trained by the pre-training model on largescale text to model the vulnerability corpus at the text level, this paper uses the method of updating only part of the parameters to conduct experiments.In the model training stage, 5% are randomly divided from the data set as the validation set and brought into the model for validation, and the remaining part is used as the training set for model training.

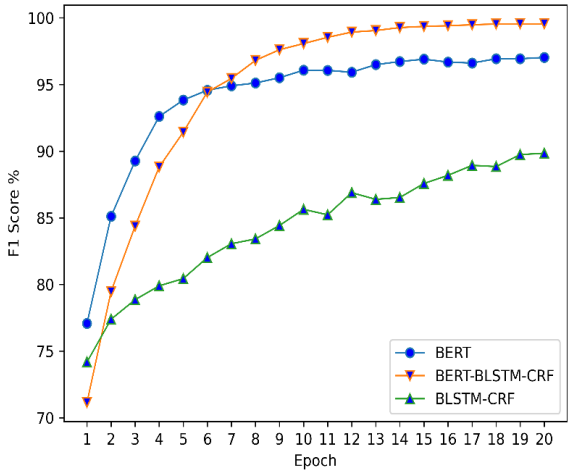

Fig. 8. Evaluate the model on the experimental dataset.

Figure8 shows the specific changes of the F1 value of each model with the number of training times.The BERTBLSTM-CRF model achieves a maximum F1 value of 99.55% when trained for 20 Epochs;The BLSTM-CRF model had a F1 value of 89.84 % at 20 Epochs, and reached a maximum F1 value of 93.62% at 35 Epochs;The BERT model reached a maximum F1 value of 96.95 % at 20 Epochs;
From the experimental results, the BERT-BLSTM-CRF model after introducing the BERT pre-training model for word vector preprocessing is superior to other models in various indicators, and the F1 value reaches 99.55%.The addition of the BERT pre-training model enables the overall model to obtain high-quality sentence-level, word-level and character-level relational features.A vulnerability summary is often composed of many sentences. BERT can synthesize the semantic information between sentences in a single summary

to generate contextual features for the entire summary, thereby improving performance.In addition, through the self-attention mechanism, the model can extract the relevance weight between each character, and the semantic information in different contexts is obtained,so it is a significant improvement compared with the RNN model.Compared with the traditional BLSTM-CRF model based on RNN, the improvement is 9.71 % under the same training number, and the comprehensive improvement is 5.93 %.This is mainly due to the fact that BERT provides better word vectors in depth and quality through the Transformer encoder structure and larger-scale parameters, so the model can obtain higher accuracy.It is worth noting that the earlier work of E Wåreus et al. [8] also used a similar RNN structure, they identified three categories of entities of vendor, product and version number and obtained an F1 value of 85.7% on the test set.The RNN model used in the comparison experiments in this paper uses a larger dataset, so the F1 value is better than the results obtained by E Wåreus et al.Furthermore, according to the test results without the version number entity, the model in this paper also outperforms the previous work in terms of F1, precision and recall.

BERT-BLSTM-CRF is also better than using the BERT pre-training model alone, and the former improves the F1 value by 2.6 %.It can be seen that BLSTM can use the information before and after the bidirectional structure learning sequence to further strengthen the features in the sequence, and CRF corrects the predicted probability value based on the correlation of adjacent labels to obtain the best label prediction.

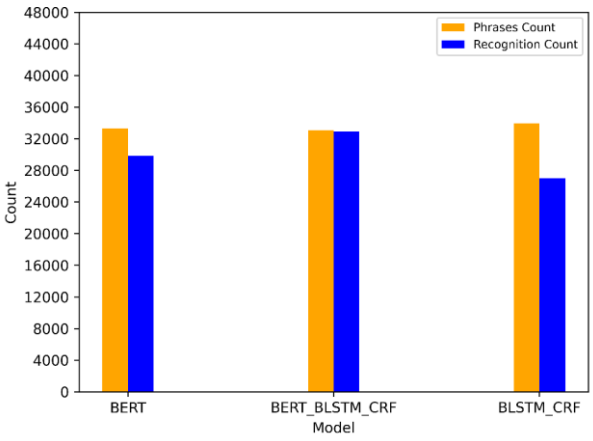

Fig. 9. Statistics of the number of phrases the model recognizes on the test set.

Table 3
Train Time of one epochs

Model	Time
BERT	758
BERT-BLSRM-CRF	1247
BLSTM-CRF	2340

Figure9 shows the number of phrases recognized by the three models on the test dataset and the number of phrases they were able to recognize correctly.The number of phrases recognized by BERT-BLSTM-CRF is the closest to the number of correctly recognized phrases, indicating that the model can capture contextual information more effectively.Because LSMT retains the information before and after the sequence, BLSTM-CRF recognizes the most phrases, but the recognition accuracy rate is not as good as the previous two models.

The total number of two types of labels in the dataset and the number of labels correctly identified by the three models are counted. Figure10 shows the recognition of the two types of labels by the model.In terms of recognition accuracy, BERT-BLSTM-CRF is significantly better than the other two models.

The time required for each model training round is shown in Table3.The training time of BERT-BLSTM-CRF is 1247s, and BERT is 758s.It is worth noting that BLSTM-CRF takes nearly twice as long to train one round as BERT-BLSTM-CRF, so its convergence speed is also slower than other models in the experiments.Judging from the time required for model training and the results obtained, BERT-BLSTM-CRF has higher training efficiency.

The F1 value, precision rate, and recall rate for the two entities of product name (PRODUCT) and supplier (VENDOR) are shown in Table4 below.It can be seen that in the BERT-BLSTM-CRF and BLSTM-CRF models, the prediction accuracy of supplier-type entities is low.This is because some supplier entities have acronyms, ambiguity and other interfering information, which are prone to prediction errors when more contextual information is not obtained.

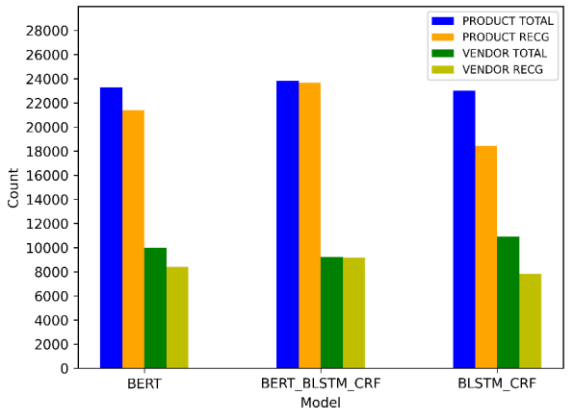

Fig. 10. The total number of labels and the number of recognitions in the test dataset.

Table 4
NER results for different types of entities

Model	Entity	F1	P	R
BERT-BLSRM-CRF	VENDOR	97.21	96.43	98.01
	PRODUCT	96.96	97.11	96.80
BERT	VENDOR	99.42	99.26	99.57
	PRODUCT	99.52	99.47	99.56
BLSTM-CRF	VENDOR	91.29	93.30	89.36
	PRODUCT	88.18	86.69	89.72

CONCLUSIONS

This paper studies the named entity recognition task in the unstructured security vulnerability summary text, and proposes a named entity recognition method combining the BERT model and BLSTM-CRF for the characteristics of the complex semantic information of the security vulnerability summary text.This method further improves the efficiency of entity labeling in the security domain.The experimental comparison shows that BERT-BLSTM-CRF has better accuracy.There are many irregularities in the description of version information in the vulnerability summary.For

example, various symbols such as " " and "-" are used to describe the version number range, and text is used to modify version number information, etc., which leads to low availability of the collected version number data.Therefore, the model in this paper only identifies two types of entities: suppliers and products.In the following research, in view of the above problems, it is planned to use a separately trained model to identify the version number in the data collection stage to accurately extract the version number data for model training, and add the version number as an entity to the recognition range of the model.On the basis of increasing the amount of data,the method is further optimized, and more features are added to describe the relationship between the text and the entity.Building a system that can automatically associate and complete CPE files is the focus of the next research.

REFERENCES

[1] S. Kalhoro, M. Rehman, F. Shaikh et al., Extracting Key Factors of Cyber Hygiene Behaviour Among Software Engineers: A Systematic Literature Review, IEEE Access (2021).

[2] C. Elbaz, L. Rilling and C. Morin, Automated Keyword Extraction from" One-day" Vulnerabilities at Disclosure, in: NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, IEEE, 2020, pp. 1–9.

[3] V. Mulwad, W. Li, A. Joshi, T. Finin and K. Viswanathan, Extracting information about security vulnerabilities from web text, in: 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, Vol. 3, IEEE, 2011, pp. 257–260.

[4] J. Sun, Z. Xing, H. Guo, D. Ye, X. Li, X. Xu and L. Zhu, Generating informative CVE description from ExploitDB posts by extractive summarization, arXiv preprint arXiv:2101.01431 (2021).

[5] S. Na, T. Kim and H. Kim, A study on the classification of common vulnerabilities and exposures using naïve bayes, in: International Conference on Broadband and Wireless Computing, Communication and Applications, Springer, 2016, pp. 657–662.

[6] G.J. Blinowski and P. Piotrowski, CVE based classification of vulnerable IoT systems, in: International Conference on Dependability and Complex Systems, Springer, 2020, pp. 82–93.

[7] Y. Chen, A.E. Santosa, A. Sharma and D. Lo, Automated identification of libraries from vulnerability data, in: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering in Practice, 2020, pp. 90–99.

[8] N. McNeil, R.A. Bridges, M.D. Iannacone, B. Czejdo, N. Perez and J.R. Goodall, Pace: Pattern accurate computationally efficient bootstrapping for timely discovery of cyber-security concepts, in: 2013 12th International Conference on Machine Learning and Applications,Vol. 2, IEEE, 2013, pp. 60–65.

[9] Y. Dong, W. Guo, Y. Chen, X. Xing, Y. Zhang and G. Wang, Towards the detection of inconsistencies in public security vulnerability reports, in: 28th USENIX Security Symposium (USENIX Security 19), 2019, pp. 869–885.

[10] S. Weerawardhana, S. Mukherjee, I. Ray and A. Howe, Automated extraction of vulnerability information for home computer security, in:International Symposium on Foundations and Practice of Security, Springer, 2014, pp. 356–366.

[11] A. Joshi, R. Lal, T. Finin and A. Joshi, Extracting cybersecurity related linked data from text, in: 2013 IEEE Seventh International Conference on Semantic Computing, IEEE, 2013, pp. 252–259.

[12] G. Huang, Y. Li, Q. Wang, J. Ren, Y. Cheng and X. Zhao, Automatic classification method for software vulnerability based on deep neural network, IEEE Access 7 (2019), 28291–28298.

[13] Y. Liu and M. Lapata, Text summarization with pretrained encoders, arXiv preprint arXiv:1908.08345 (2019).

[14] E. Wåreus and M. Hell, Automated cpe labeling of cve summaries with machine learning, in: International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, Springer, 2020, pp. 3–22.

[15] L.A. Ramshaw and M.P. Marcus, Text chunking using transformation-based learning, in: Natural language processing using very large corpora, Springer, 1999, pp. 157–176.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser and I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[17] G. Jawahar, B. Sagot and D. Seddah, What does BERT learn about the structure of language?, in: ACL 2019-57th Annual Meeting of the Association for Computational Linguistics, 2019.

[18] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[19] S. Hochreiter and J. Schmidhuber, Long short-term memory, Neural computation 9(8) (1997), 1735–1780.

[20] R. Pascanu, T. Mikolov and Y. Bengio, On the difficulty of training recurrent neural networks, in: International conference on machine learning, PMLR, 2013, pp. 1310–1318.

[21] C. Sutton, A. McCallum et al., An introduction to conditional random fields, Foundations and Trends® in Machine Learning 4(4) (2012),267–373.

[22] P. Michel, O. Levy and G. Neubig, Are sixteen heads really better than one?, Advances in neural information processing systems 32 (2019).

[23] N.S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy and P.T.P. Tang, On large-batch training for deep learning: Generalization gap and sharp minima, arXiv preprint arXiv:1609.04836 (2016).

[24] J. Pennington, R. Socher and C.D. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.