Hu *et al. Cybersecurity* (2024) 7:19
https://doi.org/10.1186/s42400-024-00206-y

Cybersecurity

RESEARCH

Open Access

Joint contrastive learning and belief rule base for named entity recognition in cybersecurity

Chenxi Hu[1], Tao Wu[1,2*], Chunsheng Liu[1] and Chao Chang[1]

Abstract

Named Entity Recognition (NER) in cybersecurity is crucial for mining information during cybersecurity incidents. Current methods rely on pre-trained models for rich semantic text embeddings, but the challenge of anisotropy may affect subsequent encoding quality. Additionally, existing models may struggle with noise detection. To address these issues, we propose JCLB, a novel model that Joins Contrastive Learning and Belief rule base for NER in cybersecurity. JCLB utilizes contrastive learning to enhance similarity in the vector space between token sequence representations of entities in the same category. A Belief Rule Base (BRB) is developed using regexes to ensure accurate entity identification, particularly for fixed-format phrases lacking semantics. Moreover, a Distributed Constraint Covariance Matrix Adaptation Evolution Strategy (D-CMA-ES) algorithm is introduced for BRB parameter optimization. Experimental results demonstrate that JCLB, with the D-CMA-ES algorithm, significantly improves NER accuracy in cybersecurity.

Keywords Named entity recognition, Cybersecurity, Contrastive learning, Belief rule base

Introduction

As cybercrimes and cyber-espionage incidents continue to escalate, cybersecurity has gained increasing significance for individuals, businesses, and governments (Ashraf et al. 2023). In the event of a cybersecurity incident, analysts need to swiftly identify entities from diverse incident logs, sourced from host log data, cyber traffic data, security alarm data, and threat intelligence data. These entities impact the cybersecurity situation, yet they are not directly observable in the actual cyber environment. Instead, they manifest within various cybersecurity events. To respond efficiently and effectively to cybersecurity incidents, it is essential to model and recognize entities across a vast array of cybersecurity data. With the development of Named Entity Recognition (NER), neural networks have been applied to entity extraction in the cybersecurity field (Gao et al. 2021). Whether utilizing pre-trained models in the representation process or employing encoders in the encoding process, these approaches allow for a comprehensive consideration of the contextual influence on each word.

However, there are still challenges within NER for cybersecurity data. Firstly, embeddings derived from pre-trained language models such as BERT often exhibit excessive clustering and uneven distribution in vector space (Gao et al. 2021). This phenomenon can lead to semantically similar tokens or token sequences being positioned further apart, while semantically unrelated tokens or sequences may end up with closely aligned vectors. The suboptimal representation of semantic similarity can skew the model's ability to accurately identify entities, potentially impacting its overall performance by favoring certain directional biases. Furthermore, present methods exhibit a deficiency when it comes to ensuring

*Correspondence:
Tao Wu
wutao20@nudt.edu.cn
[1] College of Electronic Engineering, National University of Defense Technology, Hefei, China
[2] Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China

the accuracy of entity recognition. There seems to be a significant amount of noise in cybersecurity data. For instance, an IP address that appears in the text may be incorrect either in format or in content. Despite this, current models continue to label such instances as IP addresses, demonstrating a lack of judgment on their validity.

In this paper, we propose JCLB, which **J**oins **C**ontrastive **L**earning and **B**elief rule base, designed for NER in cybersecurity. Inspired by the successful application of contrastive learning in text clustering (Hu et al. 2024), we use contrastive learning to fine-tune BERT, with the purpose of closely aligning token sequence representations for the same type of entities in vector space, while keeping them distinct from those of other token sequences. Specifically, we devise objectives based on span and position to enhance the representation similarity of both token sequence and tokens at the boundary for entities of the same type in the vector space. Additionally, to effectively filter noise and discern entity correctness, we establish regexes as rules. We learn the confidence of each rule to create a Belief Rule Base (BRB), which filters entity categories and simultaneously assesses their correctness. The BRB mitigates potential errors associated with relying solely on regexes. Furthermore, while the Covariance Matrix Adaptation Evolutionary Strategies (CMA-ES) algorithm is a robust optimization algorithm for BRB, it may not perform optimally for larger-scale or high-dimensional optimization problems (Hansen 2006; Yao et al. 2004). To address these challenges, we propose the Distributed CMA-ES (D-CMA-ES) algorithm that divides the high-dimensional search space into various subspaces with relatively lower dimensions and uses the CMA-ES algorithm to search in these subspaces. Finally, the solutions in the low-dimensional subspaces are combined to obtain the solution to the original problem.

Our contributions are as follows.

- We apply contrastive learning to fine-tune BERT, enhancing the similarity of the same type of entities in the vector space.
- We establish a BRB combining qualitative information with its capacity to define various types of uncertain information to filter noise and verify entity accuracy. Additionally, we develop the D-CMA-ES algorithm to address the high dimensions in the parameter optimization of the BRB.
- We conduct extensive experiments on two cybersecurity datasets, and the experimental results demonstrate the superiority of JCLB over existing models.

In the rest of the paper, we cover the related work in section "Related work" and then present the JCLB in section "Methodology". After reporting the experimental study in section "Experiments". We finally conclude our work in section "Conclusion".

Related work

Traditional methods for NER in cybersecurity

Early NER approaches primarily fall into two categories: rule-based and statistical machine-learning models. Rule-based methods rely on expert-crafted rules, incorporating gazetteers and syntactic lexical patterns (Etzioni et al. 2005; Bridges et al. 2017). Statistical approaches leverage machine learning algorithms such as Hidden Markov Models (Morwal et al. 2012), Support Vector Machines (Mansouri et al. 2008), Perceptrons (Jin et al. 2020), and Conditional Random Fields (CRFs) (Joshi et al. 2013; Jia et al. 2018). Mulwad et al. (2011) extracted specific vulnerabilities and attack knowledge from Wikipedia, generating machine-understandable assertions but did not consider temporal factors. Lal (2013) trained a model using Stanford NER's Conditional Random Fields, automating and enhancing zero-day attack security, yet its performance is limited on cybersecurity data. Weerawardhana et al. (2015) proposed a machine learning and part-of-speech tagging strategy to extract intelligence from online vulnerability databases.

Neural networks for NER in cybersecurity

In recent years, deep neural networks have been considered potential alternatives to traditional NER methods due to the rapid development of deep learning (Altalhi and Gutub 2021; Kashihara et al. 2022; Zhu et al. 2021; Zhang et al. 2022). Collobert et al. (2011) proposed a neural network architecture and learning algorithm that reduces reliance on prior NLP knowledge, albeit with only moderate improvements in feature representation. Huang et al. (2015) integrated BiLSTM and CRF, effectively performing sequence labeling tasks, and establishing the dominance of RNN-based sequence models in NER tasks. Kim et al. (2020) utilized a deep BiLSTM-CRF network to automatically extract key information from CTI reports, enriching feature representation by adding bidirectional dense layers. Qin et al. (2019) first extracted character features using CNN, then input them into BiLSTM to learn global word representations. They combined feature templates to extract feature vectors, obtaining more meaningful representations. Simran et al. (2020) developed a model based on multiple deep neural networks, employing a linear stack of Bi-GRU and CNN to learn hidden representations, preserving context information from different time sequences to enhance performance. Zhou et al. (2021) applied BERT-BiLSTM-CRF to cybersecurity NER tasks, using an improved BERT with full-word masking to represent

word embeddings. Gao et al. (2021) designed a data- and knowledge-driven NER network, introducing external dictionaries collected from cybersecurity-related blogs, vulnerability repositories, and Wikipedia. This enhanced word representation accurately reflects cybersecurity patterns. In addition, Li et al. (2021) introduced an adversarial active learning approach, employing BiLSTM for word embedding encoding in cybersecurity NER tasks. Another LSTM layer decoded dynamic attentional hidden representations, generating pseudo-labels to address the issue of limited annotated samples. Sarhan and Spruit (2021) constructed an open CyKG model utilizing a neural Open Information Extraction (OIE) structure based on attention mechanisms, extracting network threat data from unstructured APT reports without the need for predefined information extraction sets. Alam et al. (2022) designed an open-source Python library named CyNER, using transformer-based models and heuristic methods to extract cybersecurity-related entities and Indicators of Compromise (IOC). This framework offers good portability and scalability while providing multiple trained models.

In this paper, we introduce contrastive learning into NER in Cybersecurity to bring the span representations for similar entities closer in the embedding space. Additionally, we utilize BRB to mitigate the impact of noisy entities.

Methodology

We begin by offering an overview of JCLB, along with an illustration of the framework in Fig. 1. Sentences are initially transformed into embedding matrices via BERT

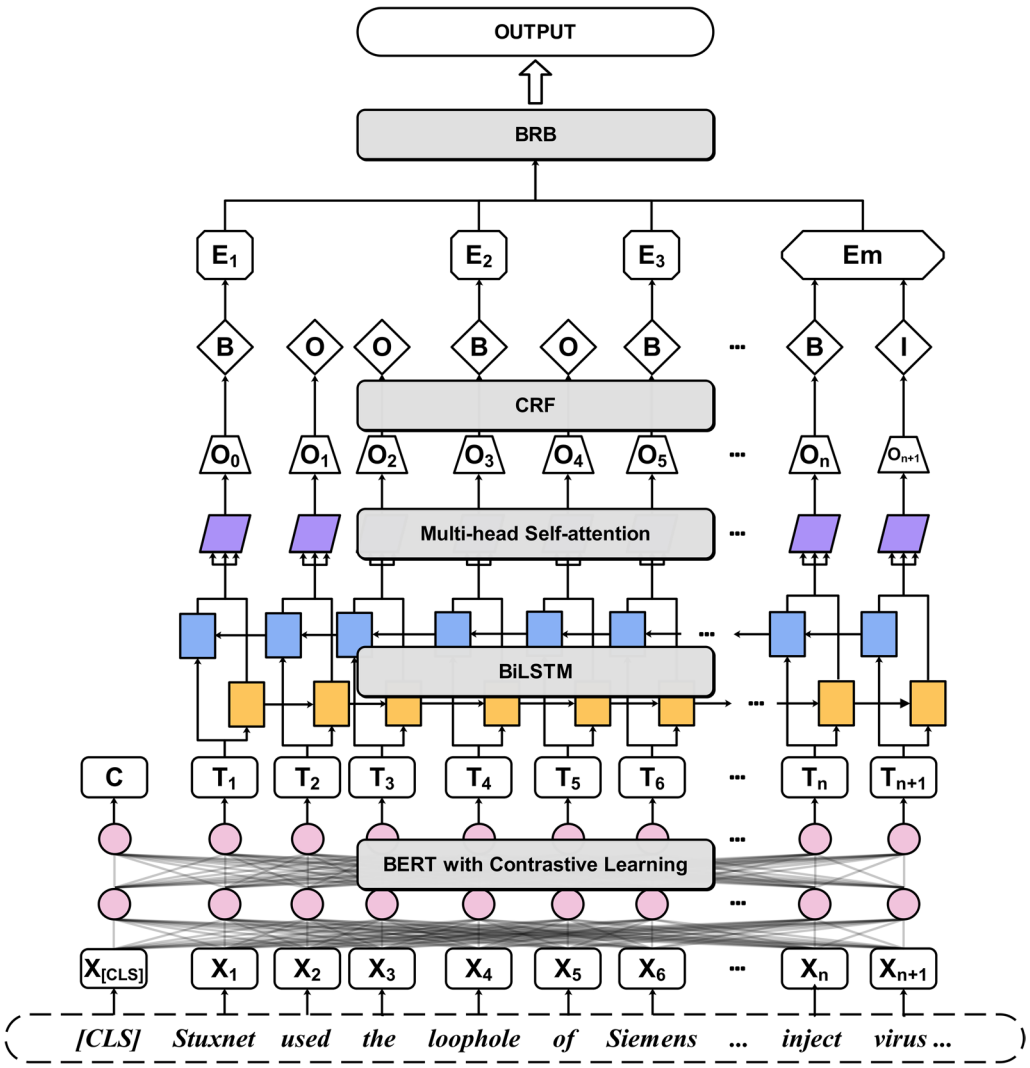

Fig. 1 The overview of the JCLB

(Section "Initialize embedding"). In this process, we employ contrastive learning to fine-tune BERT (Section "Contrastive learning for NER"). Specifically, we obtain span representations for entities in each sentence. We then generate prototypes of span, the initial token, and the final token representations for the same type of entity in a mini-batch. Based on that, we introduce three objectives via contrastive learning for NER. Then, we use BiLSTM to splice the forward and backward hidden vectors, allowing for better long-distance bidirectional semantic dependency capture (Section "BiLSTM layer"). To enable JCLB to selectively focus on more crucial parts in the input sequence when there is noise, we also introduce the MS (Section "Multi-head self-attention layer"). The CRF model is then used to predict the likelihood of each token belonging to various labels (Section "CRF layer"). Finally, a BRB is implemented to filter out inaccurately recognized entities, enhancing the precision of recognizing cybersecurity entities, particularly those typed with fixed-format phrases lacking semantics (Section "BRB layer").

Initialize embedding

JCLB first uses BERT to transform each token contained in the sentence into a vector $v \in \mathbb{R}^d$ that consists of two parts: word embedding $v_w \in \mathbb{R}^{d_w}$ and position embedding $v_p \in \mathbb{R}^{d_p}$, where d, d_w, and d_p are dimensions of v, v_w, and v_p, respectively. Hence, v is denoted as $v = [v_w, v_p]$, concatenating word embedding and position embedding. Suppose that there are n tokens in a certain sentence, it can be transformed into a sentence matrix $T \in \mathbb{R}^{d \times n}$.

Contrastive learning for NER

After obtaining the initial token vector, we introduce a contrastive learning objective to fine-tune the BERT. Contrastive learning is primarily applied in representation learning to alleviate the various idiosyncrasies of BERT. Its main purpose is to bring closer the embeddings of similar texts in the vector space while pushing apart those of dissimilar texts. As seen in Fig. 2, in NER, we aim for BERT's representations of token sequences belonging to the same entity type to be closer in the vector space, while being farther away from token sequences of other types. Based on this, we derive the vector representation for a contiguous sequence of tokens in a certain sentence with a start token in position i and an end token in position j as

$$span_{i,j} = \text{Linear}(v_i \oplus v_j \oplus l(j-i)), \tag{1}$$

where Linear is a learnable linear layer, $\oplus$ denotes the vector concatenation, $l(j-i) \in \mathbb{R}^d_l$ is the $(j-i)$-th row of a learnable span width embedding matrix $l \in \mathbb{R}^{n \times d_l}$. Assuming predefined entity types, within a mini-batch, we obtain vector representations for all sequences representing the k-th entity type e_k. The set of the vector representations is denoted as $\{span^i\}_{i=1}^K$. Then, the prototype of the vector representation set is calculated as

$$p_k = \frac{\sum_{i=1}^K span^i}{K}. \tag{2}$$

Accordingly, the span-based infoNCE (Oord et al. 2019) can be defined as

$$\mathcal{L}_{\text{span}} = -\log \frac{\exp(\text{sim}(span_{i,j}, p_k))}{\sum_{span' \in S_k^- \cup s_{i,j}} \exp(\text{sim}(span', p_k))}, \tag{3}$$

where $span_{i,j}$ denotes the sequence vector representation for an entity of type e_k, S_k^- is the set of negative sequences that all exist in the mini-batch.

The span-based objective uniformly penalizes all non-entity token sequences. Therefore, to identify the boundaries of token sequences representing a specific entity, we propose a position-based objective.

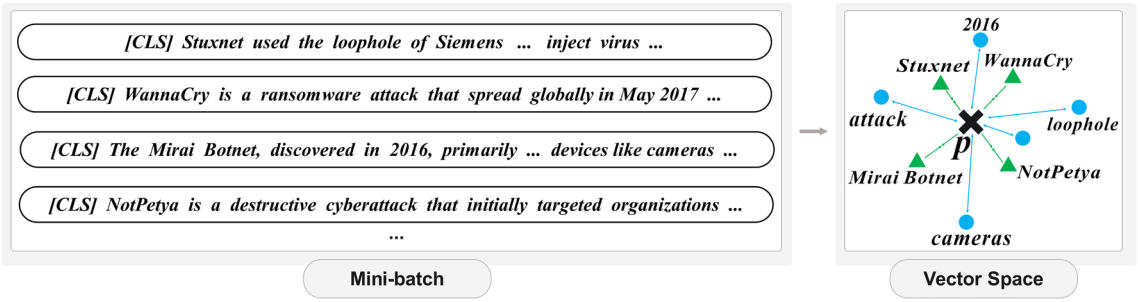

Fig. 2 The contrastive learning objectives. In the mini-batch, entities such as *Stuxnet*, *WannaCry*, *Mirai Botnet*, and *NotPetya* are predefined "malware" entities. In the vector space, for "malware" entities, we define the prototype of corresponding entity representations as anchors, denoted by cross marks. Positive samples are representations of "malware" entities, indicated by green triangles, while negative samples are representations of other token sequences, represented by blue circles

Intuitively, we want the initial (or final) tokens of entities of the same type to be closer to the embedding space. Specifically, we find the prototype of the initial (or final) token in the token sequence of entities of the same type in a mini-batch as

$$p_k^{\text{start}} = \frac{\sum_{i=1}^{K} span_{0,0}^{i}}{K}, \tag{4}$$

$$p_k^{\text{end}} = \frac{\sum_{i=1}^{K} span_{n,n}^{i}}{K}, \tag{5}$$

where n is the number of the tokens in $span^i$. Using p_k^{start} and p_k^{end} as anchors, the position-based objectives are defined by

$$\mathcal{L}_{\text{start}} = -\log \frac{\exp(\text{sim}(span_{0,0}, p_k^{\text{start}}))}{\sum_{i=1}^{n} \exp(\text{sim}(span_{i,i}, p_k^{\text{start}}))}, \tag{6}$$

$$\mathcal{L}_{\text{end}} = -\log \frac{\exp(\text{sim}(span_{n,n}, p_k^{\text{end}}))}{\sum_{i=1}^{n} \exp(\text{sim}(span_{i,i}, p_k^{\text{end}}))}. \tag{7}$$

Finally, we achieve our overall contrastive objective by integrating the three discussed objectives as

$$\mathcal{L}_{\text{cl}} = \alpha \mathcal{L}_{\text{span}} + \lambda \mathcal{L}_{\text{start}} + \gamma \mathcal{L}_{\text{end}}, \tag{8}$$

where α, λ, and γ are all hyper-parameters.

BiLSTM layer

In this section, we use BiLSTM for encoding the sentence matrix. In t-th time step, we first calculate a forgetting gate to determine what information to discard as $f_t = \sigma(W_f \cdot [h_{t-1}, v_t] + b_f)$. Secondly, we calculate the memory gate to select the information to be memorized as $i_t = \sigma(W_i \cdot [h_{t-1}, v_t] + b_i)$. The temporary cellular state is calculated as $\tilde{C}_t = \tanh(\mathrm{W_C} \cdot [\mathrm{h_{t-1}}, \mathrm{v_t}] + \mathrm{b_C})$. Thirdly, we calculate the current cell state to integrate the memory and forgetting gates, along with the temporary cell state and the previous cell state as $C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$. Finally, we calculate the output gate as $o_t = \sigma(W_o \cdot [h_{t-1}, v_t] + b_o)$, and the hidden layer state as $h_t = o_t \odot tanh(C_t)$. We can get the hidden layer state sequence with the same length as the sentence $\{h_0, \cdots, h_{n-1}\}$.

Multi-head self-attention layer

After the encoding of embeddings is completed by BiLSTM, we use the MS layer to further capture the dependency between tokens in the sequence $X = \{v_1, \ldots, v_n\}$ (Manikandan et al. 2018; Jin et al. 2020; Liao et al. 2019) and improve the robustness of JCLB.

The specific calculation of the attention mechanism is described as the mapping from a query token $Q = XW_q$ to a series of key tokens $K = XW_k$ and value tokens $V = XW_v$ in the sentence, where W_Q, W_K, and W_V are parameter matrices. The weight corresponding to each value token is obtained by calculating the similarity between the query token and each key token. The similarity between the query token and the key token is calculated by the dot product, and the attention score of the scaled dot product is as follows,

$$\text{Attention}(\mathrm{Q}, \mathrm{K}, \mathrm{V}) = \text{softmax}\frac{\mathrm{QK^T}}{\sqrt{\mathrm{d}}} \cdot \mathrm{V}. \tag{9}$$

To obtain the MS score, we perform the scaled dot product attention calculation process for h times, the input is mapped to h different subspaces through the parameter matrix, the scaled dot product attention score is calculated in turn, and the final result is spliced as the final attention score. The i-th self-attention vector is calculated as

$$u_i = \text{Attention}(\mathrm{Q_i}, \mathrm{K_i}, \mathrm{V_i}) = \text{softmax}\left(\frac{\mathrm{Q_i K_i^T}}{\sqrt{\mathrm{d}}}\right)\mathrm{V}. \tag{10}$$

Finally, the MS score is calculated as

$$\text{MS} = (u_1, u_2, \cdots, u_h) \cdot W_o, \tag{11}$$

where W_o is a wight matrix.

CRF layer

In this layer, we regard the extraction of entities in cybersecurity as a sequence marking task. We assume the sequence as $O = \{O_1, O_2, \cdots, O_n\}$. After the processing of the MS layer, we get an $n \times m$ matrix P, where n is the number of input tokens and m is the number of label types. The entry is the probability that the label i of the token j appears in the sentence. We represent $y = \{y_1, y_2, \cdots, y_n\}$ as a marker sequence, so the model calculates the corresponding score:

$$Score(x, y) = \sum_{i-1}^{n} P_{i,y_i} + \sum_{i-1}^{n+1} D_{y_{i-1}y_i}, \tag{12}$$

where, D_{ij} is the transition probability from y_i to y_j. Then, we apply softmax to obtain the normalized probability:

$$P(y|x) = \frac{\exp(\text{score}(\mathrm{x}, \mathrm{y}))}{\sum_{y'} \exp(\text{score}(\mathrm{x}, \mathrm{y}'))}. \tag{13}$$

After that, we use the maximum logarithm function for training:

$$\log P(y^x|x) = score(x, y^x) - \log\left(\sum_{y'} \exp(\text{score}(\text{x}, \text{y}'))\right). \tag{14}$$

Finally, in the prediction process, the Viterbi algorithm is used to calculate probability:

$$y^* = \text{argmax}_{\text{y}'}\text{score}(\text{x}, \text{y}'). \tag{15}$$

BRB layer

The difference between the BRB layer and the data-driven models mentioned above is that the BRB's internal structure can be explained (Yang et al. 2006). Additionally, compared to the aforementioned data-driven models, the BRB model has the ability to comprehensively utilize semi-quantitative information and describe all kinds of uncertain information (Yang et al. 2004).

Construction of BRB

We construct a BRB that comprises multiple rules. In these rules, we consider the CRF output as one of the premise attributes and also incorporate the use of powerful and easy-to-understand regexes as another necessary attribute. This helps to accurately identify cybersecurity entities with a fixed format but of no particular semantic relevance in cybersecurity incidents. Table 1 showcases some of the regexes we develop. $R_{i,j}$ refers to the j-th regex of the i-th entity category. Each rule and the premise attribute of the rule have a certain weight, and the latter part of the rule is matched with confidence to express the credibility of the conclusion. The BRB model can be described in the following form,

$$\begin{aligned} R_k:\ &\text{If } (\text{x}_1 \text{ is } \text{A}_1^{\text{k}}) \wedge (\text{x}_2 \text{ is } \text{A}_2^{\text{k}}) \wedge \cdots \wedge (\text{x}_\text{M} \text{ is } \text{A}_\text{M}^{\text{k}}), \\ &\text{Then } (\text{D}_1,\ \beta_{1,\text{k}}), \ldots, (\text{D}_\text{N},\ \beta_{\text{N,k}}), \\ &\text{With a rule weight } \theta_\text{k} \text{ and attribute weight } \delta_1, \delta_2, \ldots, \delta_\text{M}, \end{aligned} \tag{16}$$

where "$\wedge$" denotes that the rule is based on the intersection assumption. $x_i (i = 1, 2, \ldots, M)$ denotes the i-th premise attribute of BRB model, and M denotes the number of premise attributes. $R_k (k = 1, 2, \ldots, L)$ denotes the k-th rule of BRB model, $A_i^k (i = 1, 2, \ldots, M)$ denotes the reference value of the i-th premise attribute in the k-th rule, and $D_j (j = 1, 2, \ldots, N)$ denotes the j-th category, $\beta_{j,k}$ denotes the confidence of the j-th conclusion in the k-th rule, θ_k denotes the weight of the k-th rule, and δ_i denotes the weight of the i-th premise attribute. The structure of the BRB model is shown in Fig. 3. For example, the rule *If the identification result of the category of the entity output by CRF is "Identifier", and the entity can match* $R_{1,1}$, *then the confidence that the entity is "Identifier" is 100%* can be expressed as *If(Identifier is true)*$\wedge$ *(*$R_{1,1}$ *is true), then (Identifier, 100%)*.

Inference of BRB

After modeling the BRB, the input will activate the corresponding rules, and the inference results will be obtained by integrating the activation rules through the Evidential Reasoning (ER) algorithm.

Firstly, if the input of the m-th premise attribute is $x_m (m = 1, \ldots, M)$, its matching degree with the reference is calculated as follows,

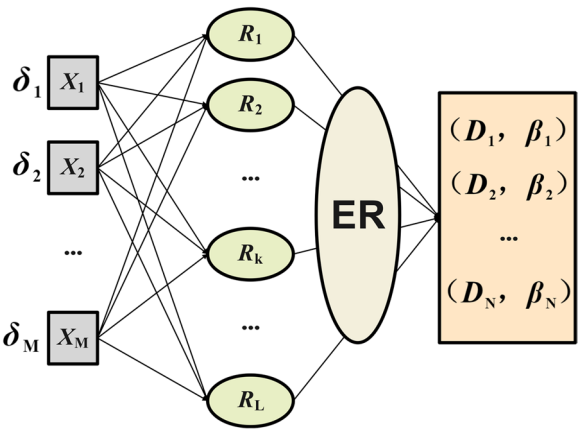

Fig. 3 Structure of the BRB

Table 1 Regexes for entities in CSS

Rule	Regex	Example
$R_{1,1}$	((([A-Za-z]{3,9}:(?:\/\/)?)(?:[-;:&=\+$,\w]+@)?[A-Za-z0-9.-]+(:[0-9]+)?(?:ww?w.[-;:&=\+$,\{\w}]+@)[A-Za-z0-9.-]+)((?:\/[\+~%\/.\w-_]*)?\??(?:[-\+=&;%@.\w_]*)#?(?:[\w]*))?)	www.google.com
$R_{1,2}$	((2[0-4]\d—25[0-5][01]?\d\d?)\.){3}(2[0-4]\d25[0-5][01]?\d\d?)	192.168.0.1
$R_{1,3}$	([0-V9a-fA-F]{2})(([0-9a-fA-F]{2}){5})	00-16-EA-AE-3C-40
$R_{1,4}$	(w+([-+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+))	xxx@163.com
$R_{1,5}$	((?:(?:25[0-5]2[0-4]d[01]?d?d).)3(?:25[0-5]2[0-4]d[01]?d?d))	255.255.255.0
$R_{2,1}$	(CVE-(19992\d{3})-(0\d{2}[1-9][1-9]\d{3,}))	CVE-2019-16201
$R_{3,1}$	exec\s+(sx)p\w+	xp_cmdshell
...	...	...

$$a_k^m = \frac{\varphi(x_m, A_{m,j}^k)\varepsilon_m}{\sum_{j=1}^{J_m}\varphi(x_m, A_{m,j}^k)}, \tag{17}$$

where ε_m represents the certainty of the input. For example, (x_m, $\varepsilon_m = 90\%$) represents that the certainty of x_m is 90%. $\varphi(x_m, A_{m,j}^k)(j \in (1, 2, \ldots, J_m))$ denotes the matching degree between input information x_m and reference value $A_m^k(A_m^k \in A_{m,1}^k, A_{m,2}^k, \ldots, A_{m,J_m}^k)$, where J_m represents the number of reference values of the m-th premise attribute. Since the input of our BRB model is qualitative information and a is in the form of semantic fuzzy value, the matching degree can be obtained directly. If entities in cybersecurity have 10 reference values (10 types), it must be one of them for any input.

Afterward, we calculate the activation weight of the rule. w_k is the activation weight, that is, the activation degree of the input information to the rule. The calculation process of activation weight is as follows,

$$w_k = \frac{\theta_k \prod_{m=1}^{M}(\alpha_m^k)^{\bar{\delta}_m}}{\sum_{\tilde{k}=1}(\theta_{\tilde{k}} \prod_{m=1}^{M}(\alpha_m^{\tilde{k}})^{\bar{\delta}_m})}, \bar{\delta}_m = \frac{\delta_m}{\max\limits_{m=1,\ldots,M}\delta_m}, \tag{18}$$

where, α_m^k is the matching degree of the input information relative to the reference value, θ_k is the initial rule weight, and δ_m is the initial attribute weight. If w_k=0, the rule is not activated.

Finally, after converting the input information into the matching degree with the reference value and obtaining the activation degree of the corresponding rules, the ER algorithm is used to integrate the activation rules.

Parameter optimization

According to the calculation process in the previous section, an objective function for optimizing BRB model parameters can be established, which is expressed as follows:

$$\begin{aligned} &\min\ f(\omega) \\ &s.t.\ 0 \le \theta_k \le 1,\ k = 1, \ldots, K \\ &\quad 0 \le \beta_{j,k} \le 1,\ j = 1, \ldots, N, k = 1, \ldots, K \\ &\quad \sum_{j=1}^{N} \beta_{j,k} = 1 \\ &\quad A_i^k \in [0, 1], i = 1, \ldots, M, k = 1, \ldots, K, \end{aligned} \tag{19}$$

where $\omega = [\theta_1, \ldots, \theta_K, \beta_{1,1}, \ldots, \beta_{N,K}, A_{1,1}, \ldots, A_{M,K}]^T$ represents the parameter vector of the BRB, $\{\theta_1, \ldots, \theta_K\}$ represent the weights of the K rules, $\{\beta_{1,1}, \ldots, \beta_{N,K}\}$ represent the confidences of the output conclusion, and $\{A_{1,1}, \ldots, A_{M,K}\}$ represent the reference values of the premise attributes. Let E_u represent the error of classification results, if there is $\hat{j} = j$, then $E_u = 0$, otherwise $E_u = 1$. Then $f(\omega)$ can be described as

$$f(\omega) = \frac{1}{m'}\sum_{u=1}^{m'} E_u^2. \tag{20}$$

To address the challenge of optimization with constraints and high dimensions, we propose the D-CMA-ES algorithm. This algorithm initially divides the high-dimensional search space into several subspaces having relatively lower dimensions and then applies the

Algorithm 1 D-CMA-ES algorithm.

1: Initialize Gaussian distribution mean μ_w, covariance matrix Σ, global search step s, and other parameters required by CMA-ES,
2: Divide the original space into N subspaces randomly, set $sub = 1$, and start an optimization round.
3: **for** $sub = 1$ to N **do**
4: The Gaussian distribution parameters in the subspace are constructed from the Gaussian distribution in the original space,
5: Sampling λ samples with Gaussian distribution $N(\mu_{sub}, s^2 \cdot \Sigma_{sub})$,
6: The constraints in a solution are transformed into specific constraint objective functions, and then the CMA-ES algorithm is used to optimize it independently to ensure that the parameters returned each time meet the constraints,
7: Calculating sample fitness with greedy posture strategy,
8: Update $\mu_{sub}, \Sigma_{sub}, \mu_w, \Sigma$.
9: **end for**

CMA-ES algorithm for searching in these low-dimensional subspaces. Subsequently, the solutions from each search are integrated to obtain the solution to the original problem. For example, if the dimension of the original space is 4, the algorithm divides the space into two subspaces, with each subspace having a dimension of 2. Each constraint is transformed into a specific unconstrained objective function that is independently optimized in each iteration to ensure that the solution always satisfies constraints. The D-CMA-ES algorithm is shown as Algorithm 1.

Experiments

Experiment settings

In this section, we describe the datasets, baseline models, implementation details, and evaluation metrics of experiments.

Datasets

JCLB is evaluated on two datasets as follows.

- Bridges et al. (2014): This dataset, encompassing data from various cybersecurity platforms like Microsoft Security Bulletins, Metasploit, and the National Vulnerability Database, features multiple entities including Applications, Vendors, Operating Systems, and Relevant Terms.
- **OpenCS**: We collect and summarize a large number of open-source unstructured cybersecurity data. The data sources include Threat Intelligence of client vault, Amazon's network security blog, CVE vulnerability description entries, and APT reports disclosed in recent years. We select 13218 sentences from the collected cybersecurity data, with a total of 248832 words. Based on this, according to the entity categories defined in the UCO ontology, the word frequency statistics, and analysis results of the filtered cybersecurity data, ten entity categories are ultimately defined to label entities in the cybersecurity data, including Organization (ORG), software (SOF), malware (MAL), vulnerability (VUL), identifier (IDE), tool (TOO), protocol (PRO), system (SYS), equipment (EQU) and attack methods (MET). The quantity statistics of different categories of elements in the data set we constructed are shown in Table 2.

In the labeling task, BIO mode is adopted, in which "B" (Begin) identifies the starting position of entities, "I" (Inside) identifies the token inside entities, and "O" (Outside) identifying the token is not in any entities. After data labeling, we divided the cybersecurity data into a 70% training set, a 10% validation set and a 20% test set for scientifically and reasonably evaluating our method proposed.

Implementation details

As the framework includes the necessary hyper-parameters required for model training, this section outlines the main hyper-parameters employed. To embed tokens, the dimension is set to 768. For sequence coding, the hidden layer of both the forward and reverse LSTM is comprised of 300 neurons, and the dropout strategy is utilized in the BiLSTM feature coding layer to prevent over-fitting. The MS module consists of a size of K and V at 64 and the number of heads at 3. During model training, the epoch is set to 100 with a batch size of 128. Further, the updated model parameters are trained through random gradient descent, with the initial learning rate being 0.001. Table 3 exhibits the specific values for these hyper-parameters. The hyper-parameters α, γ, and λ are determined through a grid search, where the step size is 0.1, and the range is [0.1,2].

Table 2 Size of each type of entities

Category	Train	Val	Test
Organization	864	238	276
Software	3682	412	1028
Malware	955	394	187
Vulnerability	1092	298	399
Identifier	6990	756	2113
Tool	2638	218	635
Protocol	517	159	114
System	1637	48	497
Equipment	1019	259	319
Attack method	2563	152	704

Table 3 Hyper-parameters in the JCLB

Parameters	Value
Initial learning rate	0.5
Embedded layer dimension	768
LSTM dimension	300
Epoch	100
Batch size	128
Dropout rate	0.5
Learning rate	0.001
Initial rule weight	1
Initial premise weight	0.5
Initial confidence	0.1

Baseline models

We compare JCLB with six baseline models as follows.

- Abdullah et al. (2018) introduce a CRF model, a statistical-based conditional probability distribution widely used for sequence labeling.
- Jie and Lu (2019) encode dependency trees using a dependency-guided LSTM-CRF architecture. Subsequently, the resulting word representation is input to the BiLSTM layer.
- Zhou et al. (2021) employ a BiLSTM-CRF architecture for the cybersecurity NER task. The BiLSTM layer extracts contextual features from input embeddings, and the subsequent CRF layer decodes sequences to predict labels.
- Gao et al. (2021) introduce a data and knowledge-driven NER model for cybersecurity. The input layer incorporates an external dictionary as an auxiliary knowledge database to enhance word representation.
- Wu et al. (2022) utilize BiLSTM, CNN, and CRF for NER. Specifically, they employ a linear stack of LSTM and CNN in the deep neural network layer for a more efficient global and local feature representation.
- Wang and Liu (2023) develop a graph RNN, GARU, integrating diverse features extracted from GNNs and RNNs. Additionally, they introduce an entity boundary detection module for predicting entity heads and tails.

Metrics

To evaluate the performance of the model, the common evaluation values in the information extraction tasks are used, Precision (P), Recall (R), and F1. P represents the percentage of correct samples identified by the model in all identified samples, and R represents the percentage of correct samples identified by the model in all identified samples. The F1 value is the harmonic average of accuracy and recall, which is used to evaluate the comprehensive performance of the model. Each evaluation index is formally expressed as follows:

$$\boldsymbol{P} = \frac{TP}{TP + FP}, \tag{21}$$

$$\boldsymbol{R} = \frac{TP}{TP + FN}, \tag{22}$$

$$\boldsymbol{F1} = \frac{2 \times P \times R}{P + R}, \tag{23}$$

where TP (True Positive) is the number of positive samples with a correct prediction, FP (False Positive) is the number of positive samples with a wrong prediction, and FN (False Negative) is the number of negative samples with a wrong prediction.

Main results

Table 4 presents the performance of various models on the Bridges et al.'s collected dataset and our collected dataset OpenCS. We report P, R, and F1. Our JCLB achieves state-of-the-art performance on the two datasets. JCLB outperforms all previous NER models in Cybersecurity, with F1 scores of 94.73% and 91.13% on the respective datasets. Notably, compared to the prior best model (Wang and Liu 2023), our approach demonstrates an improvement in F1 on the OpenCS dataset, with an absolute increase of +0.54%. It's worth noting that the previous models are built on different encoders such as LSTM, BERT, and PERT. In summary, our proposed JCLB, involving contrastive learning for fine-tuning BERT and utilizing BRB for noise filtering, represents a substantial advancement over previous models.

Ablation study

In this section, we degenerate our JCLB into several models for ablation study using the two datasets. The models are CRF (C), BiLSTM-CRF (BIC), BERT-CRF (BEC), BiLSTM-MS-CRF (BIMC), BERT-MS-CRF (BEMC), BiLSTM-BiLSTM-CRF (BBC),

Table 4 Comparison of different models on two datasets

Model	Bridges et al.			OpenCS		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
Abdullah et al. (2018)	88.92	82.27	85.47	79.35	71.59	75.27
Jie and Lu (2019)	93.50	93.00	93.25	88.43	87.62	88.02
Zhou et al. (2021)	91.94	90.79	91.36	85.64	84.19	84.90
Gao et al. (2021)	94.14	93.69	93.92	89.62	88.36	88.99
Wu et al. (2022)	91.88	93.18	92.53	80.71	78.92	79.80
Wang and Liu (2023)	94.80	94.32	94.56	91.06	90.13	90.59
JCLB (Ours)	95.16	94.30	**94.73**	91.59	90.68	**91.13**

F1 scores in bold indicate the best results

Table 5 Ablation study on two datasets

Framework	Bridges et al.			OpenCS		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
JCLB	95.16	94.30	**94.73**	91.59	90.68	**91.13**
C	87.53	81.69	84.51	79.11	71.07	74.88
BIC	88.63	84.34	86.43	83.80	73.51	78.32
BEC	89.20	85.12	87.11	86.99	75.40	80.78
BIMC	90.52	87.25	88.85	84.27	73.51	78.52
BEMC	92.15	88.63	90.36	88.93	75.40	81.61
BBC	89.62	85.82	87.68	87.95	78.14	83.63
BBMCB	93.62	92.86	93.24	90.07	89.26	89.66

F1 scores in bold indicate the best results

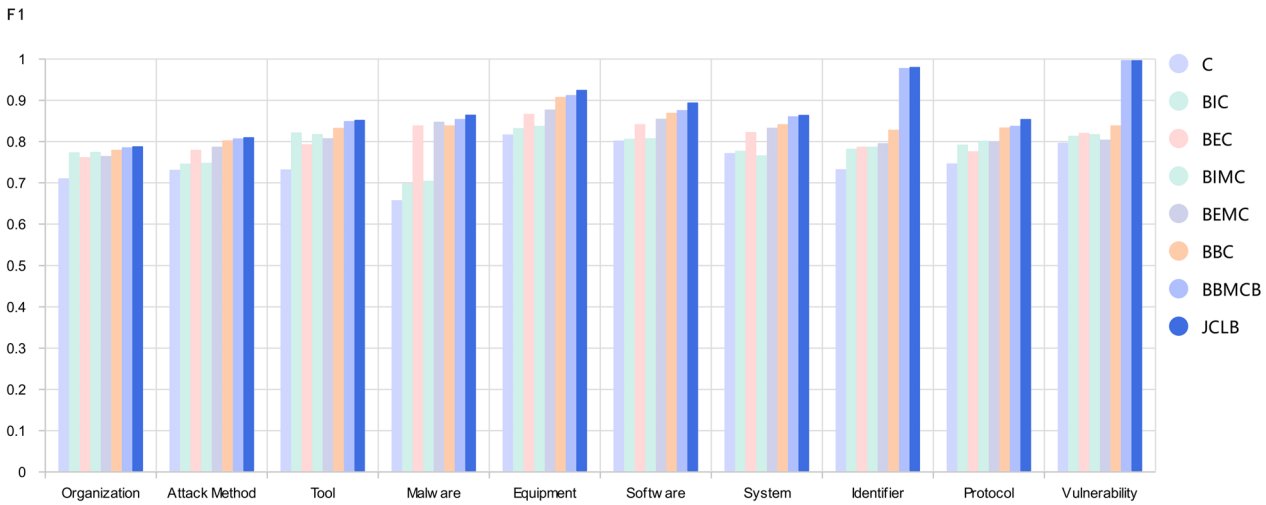

Fig. 4 Performance of recognition for ten categories of cybersecurity entities achieved by various models

and BERT-BiLSTM-MS-CRF-BRB (BBMCB). C takes into account the content information of data and the change information between data labels when modeling. Its related models have achieved good results in many natural language processing tasks. The bi-directional structure in BIC can be determined according to the context at the same time (Wu et al. 2019). In BEC, the words of each word position can be encoded directly regardless of direction and distance. BERT in BBC can embed tokens through its rich semantic knowledge (Cai et al. 2020). We compare the effect of MS and BRB on the performance of the model. We keep the hyper-parameters unchanged during the training of each model.

The experimental results are in Table 5. Figure 4 illustrates the F1 for ten different entities on the OpenCS dataset. We can observe that JCLB surpasses other models and achieves the highest F1 on the two datasets. Additionally, BIC performs significantly better than C, with an F1 improvement of 3.44%. This is because BIC addresses the issue of long-distance dependence on long-sequence modeling. In contrast, compared to using BiLSTM as a feature extractor, adding the BERT model yields a better F1. Furthermore,

Table 6 Ablation study on contrastive learning

Framework	P (%)	R (%)	F1 (%)
JCLB	91.59	90.68	**91.13**
w/o span-based objective	90.35	90.68	90.51
w/o position-based objectives	91.02	90.71	90.86

F1 score in bold indicates the best result

when compared with BIC and BEC, BIMC and BEMC show improvements of 0.20% and 0.83% in F1, respectively. Since the MS layer is employed to capture the dependency weight of feature coding between any two tokens.

Analysis on contrastive learning

We present a comparative analysis of variants of our contrastive learning method, detailing their test performance on the OpenCS dataset in Table 6. We observe an F1 decline across all these variants. We speculate that when contrastive learning focuses solely on the initial and final tokens of an entity, neglecting its span-based token sequence, it might result in the loss of semantic information contained within the intermediate tokens. In certain scenarios, these middle tokens could provide key insights into how the entity interacts with the context. Ignoring these tokens could diminish the model's understanding of the entity's meaning. Conversely, when contrastive learning focuses solely on span-based token sequences of entities, ignoring the initial and final tokens, the model overlooks the semantic information of entity boundaries carried by these tokens, leading to an inadequate understanding of the entity's integrity.

In Fig. 5, we visualize the average similarity of token sequence representations for in-batch positives and negatives sampled in span-based and position-based contrastive learning. For position-based contrastive learning, we show the mean similarity between the initial and final tokens. We observe that, as training progresses, the similarity for in-batch negatives rapidly decreases, indicating that in-batch negatives provide limited gradient signals. In contrast, the similarity for in-batch positives remains high and is distinctly separated from the negatives, suggesting that our method effectively enhances the similarity of token sequence representations for the same type of entities in the vector space.

Table 7 Ablation study on BRB

Framework	BRB	P (%)	R (%)	F1 (%)
BBMCB	✓	90.07	89.26	**89.66**
BBMCB	×	87.92	82.89	85.33
BEMC	✓	89.65	84.76	**87.20**
BEMC	×	88.93	75.40	81.61
JCLB	✓	91.59	90.68	**91.13**
JCLB	×	89.21	88.68	88.94

F1 scores in bold indicate better results obtained w/ or w/o BRB

Analysis on BRB

We evaluate the impact of using BRB on the OpenCS dataset. The experimental results are shown in Table 7, where "✓" indicates that the BRB is combined and "×" indicates that the BRB is not used. Integrating BRB enhances BBMCB with an 89.66% F1, contrasting with 85.33% when BRB is absent. In BEMC, BRB integration results in an 85.20% F1, while its removal leads to a 3.59% F1 decrease. Similarly, JCLB achieves a 91.13% F1 with BRB, but a 2.19% F1 decrease occurs without BRB. These experiments reveal BRB's notable recognition capability, particularly for entities with fixed formats.

We employ several constrained optimization algorithms, Sequential Quadratic Programming (SQP) and Differential Evolution (DE), to optimize BRB in JCLB. SQP is a traditional yet effective optimization algorithm addressing constraint problems through sequential quadratic programming subproblems. DE is an intelligent evolutionary algorithm solving constraint problems through

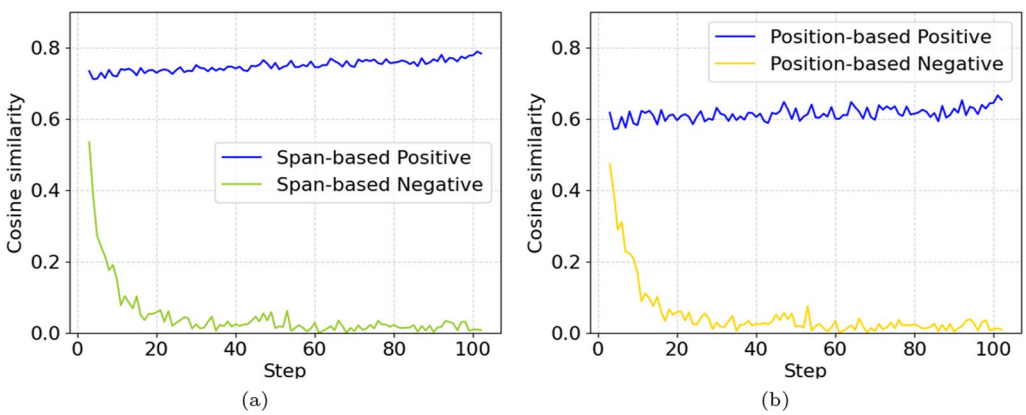

Fig. 5 **a** Variation for similarity of in-batch positive and negative pairs in span-based contrastive learning. **b** Variation for similarity of in-batch positive and negative pairs in position-based contrastive learning

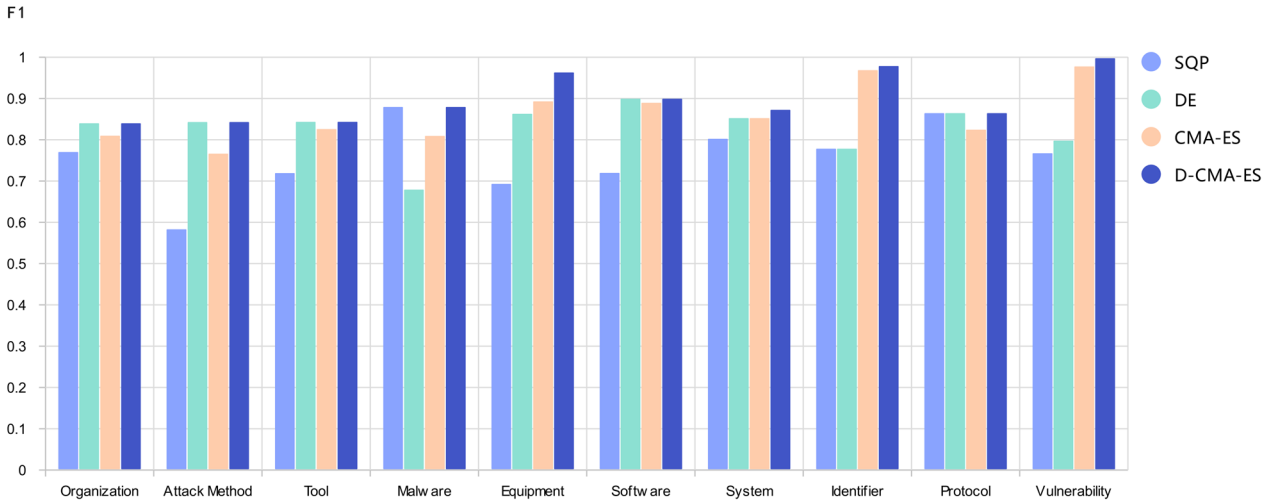

Fig. 6 The performance of JCLB with different optimization algorithms for BRB

Table 8 Case study on recognition of three types of entities

Model	BBC	JCLB (Ours)
Abbreviation	Using the POODLE vulnerability can weaken the strength of encryption protocols... BGP hijacks NAC to bypass VLAN jump attacks, DHCP starvation attacks, and rogue DHCP servers...	Using the POODLE vulnerability can weaken the strength of encryption protocols... BGP hijacks NAC to bypass VLAN jump attacks, DHCP starvation attacks, and rogue DHCP servers...
Proper noun	The MSF file system can view the details of 12 related Eternal Blue through Kali, and the MS17-010 utilized module also details the vulnerability in the system of XP...	The MSF file system can view the details of 12 related Eternal Blue through Kali, and the MS17-010 utilized module also details the vulnerability in the system of XP...
Fixed-format	First to find the corresponding network adapter, the address to 1.1.1.1, mask 255.255.255.256, gateway blank...	First to find the corresponding network adapter, the address to 1.1.1.1, mask 255.255.255.256, gateway blank...

The underlined texts denote recognized entities by two models

specialized operations. Figure 6 displays the results of these optimization algorithms, revealing that the model attains optimal recognition performance for each category when utilizing D-CMA-ES.

Case study

As seen in Table 8, we take the proposed JCLB and BBC models as examples to analyze the recognition results of entities in CSS, which are typed using abbreviations, proprietary nouns, or fixed format phrases without semantics. In the second line, we list some abbreviations like "POODLE" and "BGP". "POODLE" is the abbreviation of "Padding Oracle On Downgraded Legacy Encryption" and "BGP" is the abbreviation of "Border Gateway Protocol". The highlighted characters in Table 4 indicate that "BGP" is correctly recognized as entities, but BBC does not correctly identify "POODLE". This is because the BBMCB model uses an MS module, which encodes the input words and pays attention to other words in the context at the same time so that the label will not have the problem of difference. In the third line, "Eternal Blue" is a network attack tool that the BBC did not correctly identify. Given noise cybersecurity entities in the text, as seen in the fourth line, the BBC did not correctly identify the IP address "1.1.1.1", but it identified the malformed "255.255.255.256" as a subnet mask. The BBC can not identify the noise in the text.

Conclusion

In this paper, we propose JCLB, a novel model for NER in cybersecurity. JCLB employs contrastive learning to establish objectives based on span and position, thereby fine-tuning BERT. This method enhances the similarity of token sequence representations for the same type of entities in vector space, reducing the impact of anisotropy on encoding quality. We also demonstrate the feasibility of applying BRB to filter noise and the advantages of improving the recognition of fixed format entities. When optimizing BRB parameters, compared with the CMA-ES algorithm, we propose the D-CMA-ES algorithm, which adaptively divides samples into multiple subspaces for sampling, effectively avoiding the negative impact of

high-dimensional samples on training results. Experimental evaluations on two cybersecurity datasets affirm the efficacy of JCLB for NER in cybersecurity.

Acknowledgements
This work is supported by NSFC with No. 62002377, in part by the Hong Kong Scholars Program with No. 2021-101, in part by NSFC with Nos. 62072303, 62072424, 61872178, 62272223. This work is supported by NSFC with Nos. 62372456, 62072424, 62172063, 62272223, in part by the Hong Kong Scholars Program with No. 2021-101.

Author contributions
C.H., T.W., C.L., and C.C. conceived and designed the experiments. C.H. and C.L. performed the experiments. C.H. and T.W. prepared the figures and processed the data. C.H. wrote the main manuscript. All the authors contributed to the discussion and revision of the content.

Availability of data materials
Data will be made available on request.

Declarations

Competing interests
We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work. There is no professional or other personal interest of any nature or kind in any product, service or company that could be construed as influencing the position presented in or the review of the manuscript entitled.

Received: 1 November 2023 Accepted: 12 January 2024
Published online: 03 April 2024

References

Abdullah MS, Zainal A, Maarof MA, Nizam Kassim M (2018) Cyber-attack features for detecting cyber threat incidents from online news. In: 2018 cyber resilience conference (CRC), pp 1–4. https://doi.org/10.1109/CR.2018.8626866

Alam MT, Bhusal D, Park Y, Rastogi N (2022) CyNER: a python library for cybersecurity named entity recognition

Altalhi S, Gutub A (2021) A survey on predictions of cyber-attacks utilizing real-time twitter tracing recognition. J Ambient Intell Humaniz Comput 12(11):10209–10221. https://doi.org/10.1007/s12652-020-02789-z

Ashraf I, Park Y, Hur S, Kim SW, Alroobaea R, Zikria YB, Nosheen S (2023) A survey on cyber security threats in iot-enabled maritime industry. IEEE Trans Intell Transp Syst 24(2):2677–2690. https://doi.org/10.1109/TITS.2022.3164678

Bridges RA, Huffer KMT, Jones CL, Iannacone MD, Goodall JR (2017) Cybersecurity automated information extraction techniques: Drawbacks of current methods, and enhanced extractors. In: 2017 16th IEEE international conference on machine learning and applications (ICMLA), pp 437–442. https://doi.org/10.1109/ICMLA.2017.0-122

Bridges RA, Jones CL, Iannacone MD, Testa KM, Goodall JR (2014) Automatic labeling for entity extraction in cyber security

Cai L, Song Y, Liu T, Zhang K (2020) A hybrid bert model that incorporates label semantics via adjustive attention for multi-label text classification. IEEE Access 8:152183–152192. https://doi.org/10.1109/ACCESS.2020.3017382

Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P (2011) Natural language processing (almost) from scratch. J Mach Learn Res 12(null):2493–2537

Etzioni O, Cafarella M, Downey D, Popescu A-M, Shaked T, Soderland S, Weld DS, Yates A (2005) Unsupervised named-entity extraction from the web: an experimental study. Artif Intell 165(1):91–134. https://doi.org/10.1016/j.artint.2005.03.001

Gao C, Zhang X, Liu H (2021) Data and knowledge-driven named entity recognition for cyber security. Cybersecurity 4(1):9. https://doi.org/10.1186/s42400-021-00072-y

Gao T, Yao X, Chen D (2021) SimCSE: Simple contrastive learning of sentence embeddings. In: Moens M-F, Huang X, Specia L, Yih SW-t (eds) Proceedings of the 2021 conference on empirical methods in natural language processing, pp 6894–6910. Association for computational linguistics, Online and Punta Cana, Dominican Republic. https://doi.org/10.18653/v1/2021.emnlp-main.552. https://aclanthology.org/2021.emnlp-main.552

Hansen N (2006) In: Lozano JA, Larrañaga P, Inza I, Bengoetxea E (eds) The CMA evolution strategy: a comparing review, pp 75–102. Springer, Berlin. https://doi.org/10.1007/3-540-32494-1-4

Hu C, Wu T, Liu S, Liu C, Ma T, Yang F (2024) Joint unsupervised contrastive learning and robust GMM for text clustering. Inf Process Manage 61(1):103529. https://doi.org/10.1016/j.ipm.2023.103529

Huang Z, Xu W, Yu K (2015) Bidirectional LSTM-CRF models for sequence tagging

Jia Y, Qi Y, Shang H, Jiang R, Li A (2018) A practical approach to constructing a knowledge graph for cybersecurity. Engineering 4(1):53–60. https://doi.org/10.1016/j.eng.2018.01.004. (**Cybersecurity**)

Jie Z, Lu W (2019) Dependency-guided LSTM-CRF for named entity recognition. In: Inui K, Jiang J, Ng V, Wan X (eds) Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), pp 3862–3872. Association for Computational Linguistics, Hong Kong, China. https://doi.org/10.18653/v1/D19-1399 . https://aclanthology.org/D19-1399

Jin L, Chen M, Chunjiang Z, Xian F (2020) Strategic path and countermeasures for developing internet plus modern agriculture. Strateg Study Chin Acad Eng 22(4):50. https://doi.org/10.15302/J-SSCAE-2020.04.006

Jin Y, Wu D, Guo W (2020) Attention-based lstm with filter mechanism for entity relation classification. Symmetry. https://doi.org/10.3390/sym12101729

Joshi A, Lal R, Finin T, Joshi A (2013) Extracting cybersecurity related linked data from text. In: 2013 IEEE seventh international conference on semantic computing, pp 252–259. https://doi.org/10.1109/ICSC.2013.50

Kashihara K, Sandhu HS, Shakarian J (2022) Automated corpus annotation for cybersecurity named entity recognition with small keyword dictionary. In: Arai K (ed) Intelligent systems and applications. Springer, Cham, pp 155–174

Kim G, Lee C, Jo J, Lim H (2020) Automatic extraction of named entities of cyber threats using a deep bi-lstm-crf network. Int J Mach Learn Cybern 11(10):2341–2355. https://doi.org/10.1007/s13042-020-01122-6

Lal R (2013) Information extraction of security related entities and concepts from unstructured text. Master's thesis. Ebiquity Lab

Li T, Hu Y, Ju A, Hu Z (2021) Adversarial active learning for named entity recognition in cybersecurity. Comput Mater Continua 66(1):407–420. https://doi.org/10.32604/cmc.2020.012023. (**Cited by: 12; All Open Access, Gold Open Access**)

Liao F, Ma L, Pei J, Tan L (2019) Combined self-attention mechanism for Chinese named entity recognition in military. Future Internet. https://doi.org/10.3390/fi11080180

Manikandan R, Madgula K, Saha S (2018) TeamDL at SemEval-2018 task 8: cybersecurity text analysis using convolutional neural network and conditional random fields. In: Proceedings of the 12th international workshop on semantic evaluation, pp 868–873. Association for computational linguistics, New Orleans, Louisiana. https://doi.org/10.18653/v1/S18-1140 . https://aclanthology.org/S18-1140

Mansouri A, Affendey L, Mamat A (2008) Named entity recognition using a new fuzzy support vector machine. Int J Comput Sci Netw Secur 8

Morwal S, Jahan N, Chopra D (2012) Named entity recognition using hidden Markov model (hmm). Int J Nat Lang Comput 1:15–23. https://doi.org/10.5121/ijnlc.2012.1402

Mulwad V, Li W, Joshi A, Finin T, Viswanathan K (2011) Extracting information about security vulnerabilities from web text. In: 2011 IEEE/WIC/ACM international conferences on web intelligence and intelligent agent technology, vol 3, pp 257–260. https://doi.org/10.1109/WI-IAT.2011.26

Oord A, Li Y, Vinyals O (2019) Representation learning with contrastive predictive coding

Qin Y, Shen G-W, Zhao W-B, Chen Y-P, Yu M, Jin X (2019) A network security entity recognition method based on feature template and cnn-bilstm-crf. Front Inf Technol Electronic Eng 20(6):872–884. https://doi.org/10.1631/FITEE.1800520

Sarhan I, Spruit M (2021) Open-cykg: An open cyber threat intelligence knowledge graph. Knowl-Based Syst 233:107524. https://doi.org/10.1016/j.knosys.2021.107524

Simran K, Sriram S, Vinayakumar R, Soman KP (2020) Deep learning approach for intelligent named entity recognition of cyber security

Wang X, Liu J (2023) A novel feature integration and entity boundary detection for named entity recognition in cybersecurity. Knowl-Based Syst 260:110114. https://doi.org/10.1016/j.knosys.2022.110114

Weerawardhana S, Mukherjee S, Ray I, Howe A (2015) Automated extraction of vulnerability information for home computer security. In: Cuppens F, Garcia-Alfaro J, Zincir Heywood N, Fong PWL (eds) Foundations and practice of security, pp 356–366. Springer, Cham

Wu G, Tang G, Wang Z, Zhang Z, Wang Z (2019) An attention-based BiLSTM-CRF model for Chinese clinic named entity recognition. IEEE Access 7:113942–113949. https://doi.org/10.1109/ACCESS.2019.2935223

Wu X, Zhang T, Yuan S, Yan Y (2022) One improved model of named entity recognition by combining bert and BiLSTM-CNN for domain of Chinese railway construction. In: 2022 7th international conference on intelligent computing and signal processing (ICSP), pp 728–732. https://doi.org/10.1109/ICSP54964.2022.9778794

Yang J-B, Liu J, Wang J, Liu G-P, Wang H-W (2004) An optimal learning method for constructing belief rule bases. In: 2004 IEEE international conference on systems, man and cybernetics (IEEE Cat. No.04CH37583) vol 1, pp 994–9991. https://doi.org/10.1109/ICSMC.2004.1398434

Yang J-B, Liu J, Wang J, Sii H-S, Wang H-W (2006) Belief rule-base inference methodology using the evidential reasoning approach-rimer. IEEE Trans Syst Man Cybern A Syst Humans 36(2):266–285. https://doi.org/10.1109/TSMCA.2005.851270

Yao X, Burke EK, Lozano JA, Smith J, Merelo-Guervós J, Bullinaria JA, Rowe JE, Tiňo P, Kabán A, Schwefel HP (2004) [lecture notes in computer science] parallel problem solving from nature—PPSN VIII volume 3242—evaluating the cma evolution strategy on multimodal test functions https://doi.org/10.1007/b100601(Chapter 29), pp 282–291

Zhang P, Wang X, Ya J, Zhao J, Liu T, Shi J (2022) Darknet public hazard entity recognition based on deep learning. In: Proceedings of the 2021 ACM international conference on intelligent computing and its emerging applications. ACM ICEA' 21, pp 94–100. Association for computing machinery, New York, NY. https://doi.org/10.1145/3491396.3506525

Zhou S, Liu J, Zhong X, Zhao W (2021) Named entity recognition using bert with whole world masking in cybersecurity domain. In: 2021 IEEE 6th international conference on big data analytics (ICBDA), pp 316–320. https://doi.org/10.1109/ICBDA51983.2021.9403180

Zhu X, Zhang Y, Zhu L, Hei X, Wang Y, Hu F, Yao Y (2021) Chinese named entity recognition method for the field of network security based on roberta. In: 2021 international conference on networking and network applications (NaNA), pp 420–425. https://doi.org/10.1109/NaNA53684.2021.00079

Publisher's Note