Cross-site Scripting Threat Intelligence Detection Based on Deep Learning

Zhonglin Liu, Yong Fang, and Yijia Xu(✉)

School of Cyber Science and Engineering, Sichuan University, Chengdu, Sichuan, China

jungleforsa@gmail.com, yfang@scu.edu.cn, xuyijia@stu.scu.edu.cn

Abstract. In an increasingly complex cyber environment, where the role of traditional protection tools is increasingly limited, intelligence is the key point in the battle. Through the information monitoring of Internet social platforms, potential cyberattack threats to enterprises, governments, and other institutions could be analyzed. Twitter, the world's largest social media platform, spreads news and shares tweets about cybersecurity-related events and technologies daily, with cross-site scripting attacks being one of them. In the status quo, this paper proposes a cross-site scripting threat intelligence detection model based on deep learning, which can detect tweets involving threats related to cross-site scripting attacks. We utilized a variety of word vector extraction tools blended with topic word extraction techniques to construct a word vector matrix with multi-dimensional features. Then, the threat event detection model is trained using a bidirectional recurrent convolutional neural network with a self-attentive mechanism. In the experiment, the accuracy rate of our proposed model exceeds 0.96, and through multiple sets of control experimental data results, it is proved that the structure designed in the model is conducive to improving the performance of the model and that the model is effective in detecting tweets that involve cross-site scripting threats.

Keywords: Neural networks · Cross-site scripting · Threats · Deep learning

1 Introduction

As Internet technology becomes more popular, cyber threats are happening all the time. According to the report of the World Economic Forum in 2021, the frequency of data fraud, theft, and cyberattacks has greatly increased economic and social risks [1]. Therefore, the ability to detect cyber threats in time can effectively reduce the losses of related departments and provide reliable evidence and clues for prosecutors [2].

The Twitter social platform is one of the most visited social media in the world. There are a large number of news and personal tweets posted every day, which contain a wide range of information, including security-related events and news [3]. There are also many cybersecurity practitioners sharing information

E. Ahene and F. Li (Eds.): FCS 2022, CCIS 1726, pp. 89–104, 2022.
https://doi.org/10.1007/978-981-19-8445-7_6

about new vulnerabilities, exploit codes, and related security incident details they understand or research on social media [4]. In this way, this cybersecurity data generator can be used to help researchers investigate the identification of short-text content threats.

Cross-site scripting attack is a very common script injection attack. It is the top three vulnerabilities in web security threats in the OWSAP report all year round [5]. It enables the victim's browser to load and execute malicious JavaScript scripts after visiting a website or page containing malicious code, thereby being controlled and attacked by the malicious script code. So we set cross-site scripting threat intelligence as the research object of this paper.

Deep learning is currently one of the core technologies leading the development of artificial intelligence and is widely used in the field of pattern recognition and text classification [6]. In this paper, we took advantage of natural language processing to analyze text and proposed a multi-dimensional text feature detection model based on deep learning to detect related threats that appeared on Twitter. Our main contributions are as follows:

- We proposed a multi-dimensional method to obtain word-level feature vectors. It is a combination of a variety of pre-training word vector tools so that our model can obtain more features and context-related information from tweets.
- We used the Latent Dirichlet Allocation (abbreviated as LDA) algorithm to extract relevant topic words in the tweets to obtain text-level features. These features can improve the performance of the model in discriminating tweets.
- We have designed a state-of-the-art detection model. This model is based on the Bi-LSTM neural network, which learns the context and various features, and then uses the self-attention mechanism to improve the learning efficiency, so that our detection model has a better classification effect. The final model has an accuracy rate of over 0.96 under actual data.

The rest of the paper is organized as follows. In Sect. 2, the research results are introduced on threat intelligence detection in recent years. In Sect. 3, We will introduce the specific details and key technologies of the detection model. The results of our experiment and evaluation will be shown in Sect. 4. Section 5 is the summary of the work.

2 Related Work

A large amount of real-time data on social platforms has attracted many researchers to study related security incidents. Cyber threat incidents are always a hot topic and it can be seen as a dichotomous problem. Researchers usually use machine learning methods to find the trigger words of the event and then classify the text. Willett et al. proposed a framework for network assessment, which extracts threat events and assesses the degree of damage, which is used to assist the security personnel in better understanding the global network situation [7]. Qiu et al. performed a detailed parsing of sentence features and used the obtained feature information to classify cyber-attack events, while also pointing out that the detection of event types is best suited to the trigger-matching method, and

that the performance of word embedding feature models trained on large corpora is also superior to that of other models [8]. Khandpur et al. proposed a query strategy based on convolution kernel and dependency analysis to detect various types of network attack events, such as data leakage, account hijacking, and code injection [9]. However, due to the high training cost and computational cost of the autoencoder, it is not conducive for other researchers to expand and adjust on this basis. Le et al. proposed a keyword seed-based learning framework for threat event detection [10], a scheme that is good at detecting known threats but does not discriminate well against unknown kinds of threats. The unsupervised learning scheme proposed by Bose et al. selects and extracts key terms in the text [11], calculates a threshold for the weight of the term in the threat event, and then ranks its severity. However, the processing of its text features is too narrow, and it is easy to make the model over-fitting. In some specific cases, it cannot be classified accurately. The supervised learning method proposed by Ji et al. [12] used a multi-task learning model to detect security threats in tweets. The feature processing method is too complicated and the final experimental effect is mediocre. Trong et al. used a manually labeled data set to use security event trigger words to detect events, which is not efficient [13]. Shin et al. considered the detection of threat events through repeated words and new words [14]. This scheme can theoretically effectively identify new threat events. Balakrishna et al. proposed a high-level text representation method to learn contextual features in samples [15], and detect threat events through high-dimensional and deep text features.

Given the problems and shortcomings of the above research, we start from the text feature vector representation method, through the fusion of different dimensional vector representations, and with the help of the deep neural network to extract high-dimensional features, we propose a multi-dimensional text based on deep learning feature detection model.

3 Methodology

3.1 Overview

Aiming at the identification of cross-site scripting threats, we designed a scientific and effective detection model based on the data obtained on Twitter. The overall architecture of the model is shown in Fig. 1 and is divided into three parts: data collection, Multi-dimensional feature vector fusion, and deep neural network.

i) **Data collection**. The purpose is to provide data for training the deep neural network model. We used a crawler to obtain a large amount of tweet data containing related threats and not containing threats from Twitter. It is hoped that the meaning of the entire tweet can be obtained from the terms of the tweet, contextual content, topic words, etc., to detect tweets involving threats. In addition, to facilitate feature extraction and deep neural network training, we clean and de-duplicate the collected data, delete punctuation marks, emoticons, and other interference information in tweets, and mark positive and negative samples.

ii) **Multi-dimensional feature vector fusion**. Traditional event detection needs to consider event trigger words in the text, which is inefficient. To achieve a better model detection effect, we have carried out better-vectorized expressions for short text content such as tweets. The existing pre-training models are Word2Vec, GloVe, and FastText, each of which has different characteristics. We used these three pre-training models to train and vectorize the representation of this paper respectively and then concatenate it so that the text is converted into a feature vector with multiple encoding features. In addition, we not only considered the vector features of the words themselves, but we used LDA (Latent Dirichlet Allocation), which implies Dirichlet distribution, to extract the topic words of tweets so that we can obtain text-level feature vectors. Finally, the feature vector obtained by the fusion of the text feature and the word feature can fully express the meaning of the tweets in multiple dimensions and also serve as the input of the deep neural network model.

iii) **Deep neural network**. Aiming at the acquired feature vectors of tweets, and considering that the text content has the characteristics of contextual relevance, we used Bi-directional Long Short-Term Memory neural network (abbreviated as Bi-LSTM) to train the content threat detection model. The definition of a threat event is usually determined by the trigger word and the elements that appear in certain specific events. In a tweet, different tweet fragments contribute differently to the definition of the entire event. Therefore, a self-attention mechanism is introduced into the detection model, it can weigh the importance of different lexical units, adjust the feature weight value, reduce a certain data dimension, make the detection model pay attention to the key information of the current content, and improve the accuracy of the model.

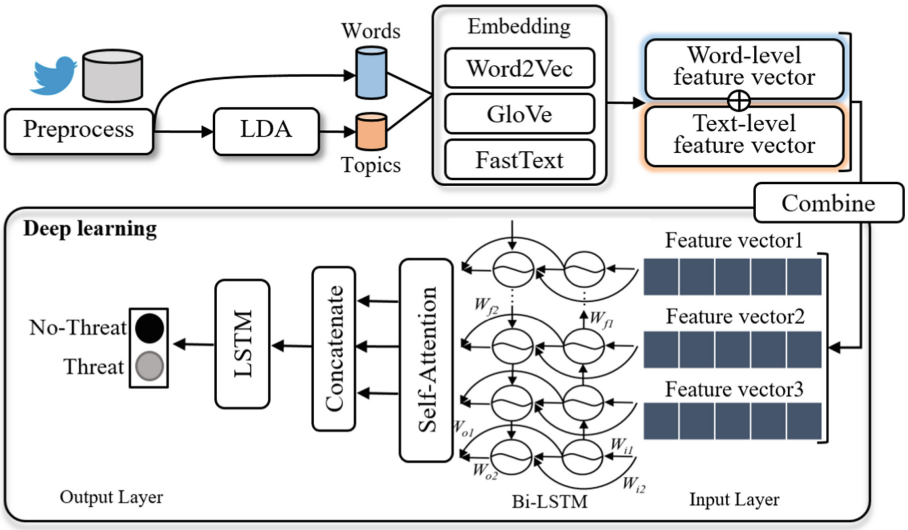

Fig. 1. The overall framework of the threat event detection model

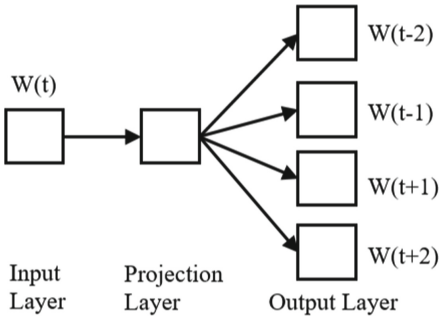

Fig. 2. Skip-Gram model structure

3.2 Feature Integration

Tweets are made up of words, but this cannot be directly applied to our model as training parameters, so we have to convert the words of the tweets into a vector representation via a word vector model. The three most effective word vector models are Word2Vec, GloVe, and FastText. In addition, to obtain text-level tweet features, we use the LDA model to extract the topic words of the tweet and combine them with the tweet vector, so that the deep neural network can learn more information.

Word-Level Features. Word2Vec. Although traditional One-hot vectors are easy to understand, the resulting vector matrix becomes very sparse when encountering a large collection of data with a large lexicon, which does not represent the information contained in the text well, nor does it represent contextual associations. In addition, the One-hot method generates a huge matrix when calculating, which will reduce the computational efficiency of the model [16]. Therefore, the deep learning research team under Google proposed the Word2Vec word vector model in 2013. This model can reduce the dimensionality of the data generated by the original One-hot model, and calculate the cosine similarity of the word vector space to establish the correlation between the upper and lower questions in this article [17]. The common training models of the Word2Vec tool are CBOW and Skip-Gram. The Skip-Gram model is used in this paper.

The structure of the Skip-Gram model is shown in Fig. 2. Set up a window with adjustable width, use the word in the center of the window as the output, slide from the beginning of the sentence to the end in turn, and then adjust the parameters through the neural network gradient descent method. After the training is complete, the words can be represented as vectors in the network using the weight matrix. Assuming that Wt represents the central word, then $W_{t-c}, W_{t-c+1}, ..., W_{t+c}$ represents the context of the central word, the objective function of Skip-Gram is shown in Eq. (1), and c represents the window size:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leqslant j\leqslant c, j\neq 0} logp(w_{t+j}|w_t) \tag{1}$$

Using the Softmax function, the conditional probability $P(w_t|w_{t+j})$ is obtained as shown in formula (2), where $v_{w_{t+j}}$ represents the vector of the word w_{t+j}.

$$P(w_t|w_{t+j}) = \frac{exp(v_{w_t}^t v_{c_{t+1}})}{\sum_{c=1}^{C}(v_{c_t}^T v_c)} \tag{2}$$

GloVe. GloVe is designed by Stanford University after improving the Word2Vec encoding model [18]. It is characterized by using the co-occurrence information characteristics of words and words, fitting the differences between co-occurring words of different word pairs so that the word vector model can obtain the global information of the text. In terms of formula description, the probability of word j appearing in the environment of the word i is shown in formula (3):

$$P_{ij} = P(j|i) = \frac{x_{ij}}{x_i} \tag{3}$$

At the same time, a constraint condition is needed to make the formula hold, that is, the distance between words, for example, the distance between words does not exceed 3 words. The main contribution of GloVe is to establish the association between the co-occurrence probability and the word vector. Then there is the co-occurrence probability Pij, the word i and the word j respectively represent the central word and the background word, and the corresponding vectors are represented by V and $\widetilde{v}$.

$$f(v_i, v_j, \widetilde{v_k}) = \frac{p_{ik}}{p_{jk}} \tag{4}$$

For the symmetry of any pair of words, that is, $v_i = \widetilde{v_j}$, and $X_{ij} = \widetilde{X_{ji}}$, then:

$$f((v_i - v_j)^T \widetilde{v_k}) = \frac{f(v_i^T \widetilde{v_k})}{f(v_j^T \widetilde{v_k})} \tag{5}$$

$$f(x) = exp(x) \tag{6}$$

available:

$$v_i^T \widetilde{v_k} = log(x_{ik}) - log(x_i) = log(x_i k) - b_i - b_k \tag{7}$$

The loss function is defined as:

$$Loss = \sum_{i,j=1}^{V} f(x_{ij})(v_i^T \widetilde{v_j} + b_i + b_j - log(x_{ij}))^2 \tag{8}$$

The vector of the final word is represented by the sum of the vector of the center word and the background word.

FastText. This is a fast text classifier and word vector tool proposed by Facebook, which provides efficient feature representation and text classification, and is mainly used in synonym mining and text classification systems in the company's systems [19]. Due to the simple structure of the model, it is very convenient and fast to represent word features. This model is also based on the improved Word2vec vector model, which proposes a subword embedding approach based on the original, taking into account additional features of the character-level N-gram. For short textual word vector training, the model can combine the internal structure information of words with short textual character features more quickly. The FastText structure is shown in Fig. 3.

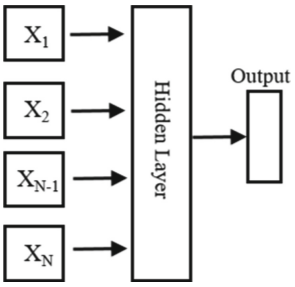

Fig. 3. FastText structure

X_1, X_2, X_N represent N-gram vectors corresponding to different texts. For a phrase w, combined with the sub-word information, the sub-word with a length of 3–6, that is, the N-gram character set is expressed as $G_w \subset \{1, ..., G\}$. Assuming that the sub-word vector in the dictionary is denoted as z_g and the word vector of the vocabulary w is denoted as v_w, then there is an objective function as shown in formula 9.

$$v_w = \sum_{g \in G_w} z_g \tag{9}$$

Text-level Features and Integration. The representation of word features can be enhanced by integrating the features obtained from the three different pre-trained models in 3.2.1. The focus of these word embedding models is still on the words themselves, which are word-level features. The description of an event must have the theme it wants to express. Most sentences of the same type of event have similar expressions. Therefore, we introduced the LDA topic classification model to enhance the expression of text features, obtain the topic keywords of a tweet, and get text-level features.

The LDA topic word generation model was originally a three-layer Bayesian production model proposed by Blei et al. [20], which created the association from

text to topic and then to words. The basic scheme is to treat the document as the probability distribution of the topic described by the keyword. The topic words can be used as the text clustering center, and the texts of the same topic can be aggregated. Topic words can also be used for feature generation, and the LDA model to obtain the topic distribution of the text and extract keywords. So that the feature space of the text can be dimensioned down and can be used for a variety of prediction tasks. The model design framework is shown in Fig. 4.

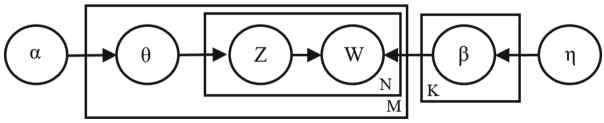

Fig. 4. LDA topic classification model

M is the number of texts and N is the number of words, which are associated with K topics. W is each word, and Z is the topic distribution of its word, α is the hyperparameter of θ, β is the probability distribution of word and topic, θ is the probability distribution of text and topic, and η is the hyperparameter expression of β.

To transform each tweet into a feature vector that has enough feature information for deep learning, we merge text-level features with word-level features. That is, the text-transformed word vector feature matrix is stitched with the topic word vector matrix, and the topic words obtained through the LDA model are used as global additional feature words. Finally, the tweets are passed through the word vector tool of the same dimension, combined with the text-level features, and the encoding process of the feature vector matrix is shown in Fig. 5.

3.3 Deep Learning Algorithm

Recurrent Neural Network. Recurrent neural network (referred to as RNN) in deep learning is an important branch of it, and its main feature is the ability to obtain contextual feature information [21]. Researchers have improved their long-term dependence and built a long and short-term memory neural network referred to as LSTM. It can solve the problem of gradient disappearance and gradient explosion in the process of model training, making it perform better in tasks that rely on contextual information. Compared with a simple RNN, the repetitive module of LSTM is much more complicated. This repetitive module is called a cell.

Inside the cell, the LSTM network will go through three stages [25]. The first stage is the forgetting stage. Whether to discard the output content of the previous node, is also called the forget gate; The second stage is to select the memory part, and the input content can be selectively memorized after calculation; The last stage is used to control the output, which can control

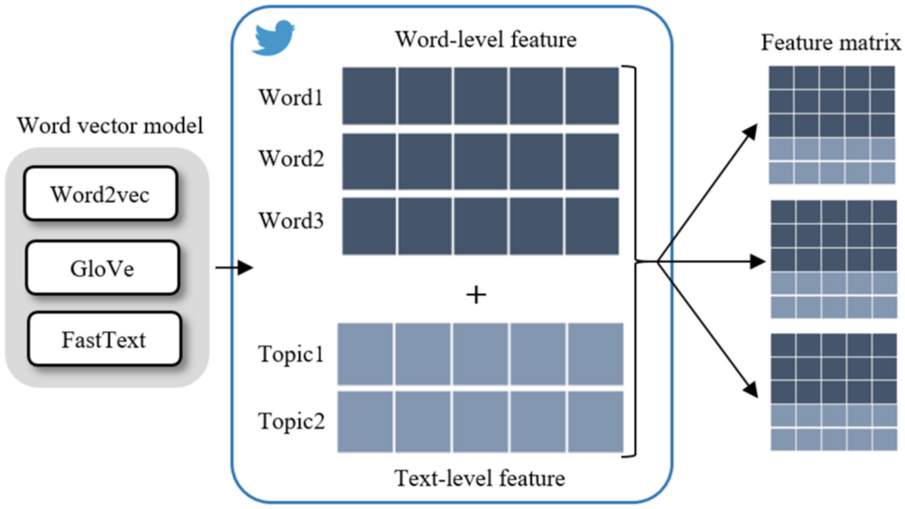

Fig. 5. Text feature vector combination process

whether the cell outputs the results of the current round of calculations. The calculation process is shown in the following Eqs. 10-15.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{10}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{11}$$

$$\widetilde{C_t} = tanh(W_C[h_{t-1}, x_t] + b_C) \tag{12}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \tag{13}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{14}$$

$$h_t = o_t * tanh(C_t) \tag{15}$$

In the formula, α represents the Sigmoid activation function, W is the weight of the layer, and b represents the deviation. f_t represents the forget gate, i_t is the input gate, h_t represents that the selective forgetting function at time t is implemented by the sigmoid layer, mainly referring to the output of the previous stage and the input of the current stage, and then implementing selective forgetting. In the second stage, the information stored in the LSTM network cell is determined by using the *tanh* function to obtain the candidate vector, multiplying the previous state value by the corresponding forgetting gate, and adding the product of the input gate and the candidate value to obtain the final cell state Value C.

Researchers had proposed bi-directional recurrent neural networks to improve LSTM into Bi-LSTM (Bi-directional Long Short-Term Memory) to better use forget gates and memory gates to capture contextual relationships and obtain features that are not limited to sequential inputs [22]. The output of Bi-LSTM

cell at time t is updated to $h_t = \overrightarrow{h_t f} + \overrightarrow{h_t b}$, where $\overrightarrow{h_t f}$ is the forward hidden vector, $\overrightarrow{h_t b}$ is the backward hidden vector. For the feature divergence of short text corpus in social media, the contextual feature relationship of the text is particularly important. Therefore, the use of a bidirectional recurrent neural network will make the classification performance of the entire model better.

Self-attention. The attention mechanism is similar to the feature that humans can quickly capture important information in a paragraph of text or a picture. By paying more attention to key information, the main content can be obtained more accurately. The definition of a threat event is usually determined by trigger words or some specific elements [23]. Different text fragments make different contributions to the detection task of the model in this event, so this paper introduces a self-attention mechanism to improve the performance of the model. The self-attention mechanism is a special case of the attention mechanism, that is, every word needs attention calculation with the words in the sentence, the purpose is to learn the relationship between words in the sentence [24].

The attention mechanism can weigh different grammatical units, which can reduce the data dimension and improve the accuracy of the model. Solve the long-distance dependence problem by calculating the mutual influence between words. The data source is generally represented by a series of key-value pairs, such as <k,v>, where k represents the key, and v is the corresponding value. When calculating the attention value, we need to calculate the similarity between the sequence and each key in the data, denoted by S. Common similarity calculations are dot product and multi-layer perception forms, as shown in equation (16).

$$S(k_i, q) = \begin{cases} k_i^T q & dot \\ V_a^T tanh(W k_i + U q) & preceptron \end{cases} \tag{16}$$

Then the obtained similarity S is normalized using the Softmax function, and the weight a of the sequence value is expressed by formula 17.

$$a_i = softmax(S(k_i, q)) \tag{17}$$

Finally, after the weights are obtained, they can be weighted and summed to obtain the corresponding attention value. The self-attention mechanism makes q, k, and v equal in the attention mechanism. The expression of the attention value is shown in Eq. 18.

$$Attention = \sum_{t=1}^{t} a_i * v_i \tag{18}$$

Network Structure. The neural network structure of the entire model is shown in Fig. 6. Three different word vector models are used for word embedding for a piece of the tweet, and then Bi-LSTM and self-attention mechanism are used for text feature extraction on the embedding results. Then the three output vectors

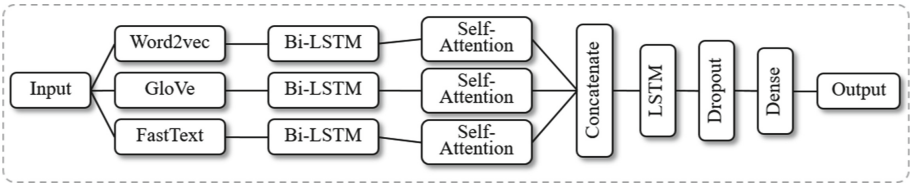

Fig. 6. Details of the detection model

are concatenated and passed into the LSTM for further feature extraction. The Dropout layer is used to prevent over-fitting, and finally the prediction result is output through the fully connected layer.

4 Experiment

To verify the reliability and effectiveness of the threat event detection model based on deep learning, we built a detector based on the overall framework of Fig. 1. First, use the collected samples to train the model and optimize the parameters, and then use the test sample set to evaluate the detection effect of the model.

4.1 Environment

The experimental environment designed in this paper is all running under the Windows 10 64-bit operating system, and all the functional modules in the content are developed and run under the Python 3.6 environment. All the involved Python libraries and their versions are shown in Table 1.

Table 1. Environment and configuration

Type	Content
Configuration	CPU: i7-10700F, RAM: 16G GPU: GeForce GTX1060 6G
Python Library	Genism 3.8.3, FastText 0.9.2, Keras 2.2.4, Keras-self-attention 0.50.0, Scikit-learn 0.24.2, Seaborn 0.9.0, Matplotlib 3.3.4, Numpy 1.19.5, Nltk 3.6.5, Pandas 1.1.5, Tensorflow-gpu 1.13.1, Tensorboard 1.13.1, Tensorflow-estimator 1.13.1

4.2 Dataset

In the paper, we used the keywords such as "cross site-scripting, XSS, alert(, CSRF" as seeds to collect tweets from the last 4 years (2018–2021) and obtained more than 24,000 valid tweets in total. After we cleaned and de-duplicated the data, we manually screened and obtained 1812 pieces of data containing cross-site scripting threats. These tweets are usually the authors sharing vulnerabilities discovered by them or reposting some exploit methods. At the same time, we also extracted 2000 tweets that did not contain threats from the data. These tweets are often advertisements containing relevant keywords or other abbreviations of the same name. Therefore, this type of control sample is highly confusing. We label the tweets that contain threats and the tweets that are not, which can be used for neural network model training.

4.3 Experiment and Analysis

The main characteristics of the threat event detection model are its multi-dimensional coding feature representation and attention mechanism. In the experiment, different parameters will affect the detection performance of the model. After comparison, we set the loss function loss of the neural network to Categorical_Crossentrop, Adam as the optimization function; the number of neurons in the Bi-LSTM hidden layer is 200, the number of neurons in the LSTM hidden layer is 128, the Batch-size is 32, and the epoch is 30; The dimension of the word vector pre-training model is 128, the window is set to 5, the number of iterations is 5, and 2 subject words are obtained through LDA. At the same time, to make the experimental results more objective, we conducted multiple sets of controlled experiments in the experiment to verify the effectiveness of the method.

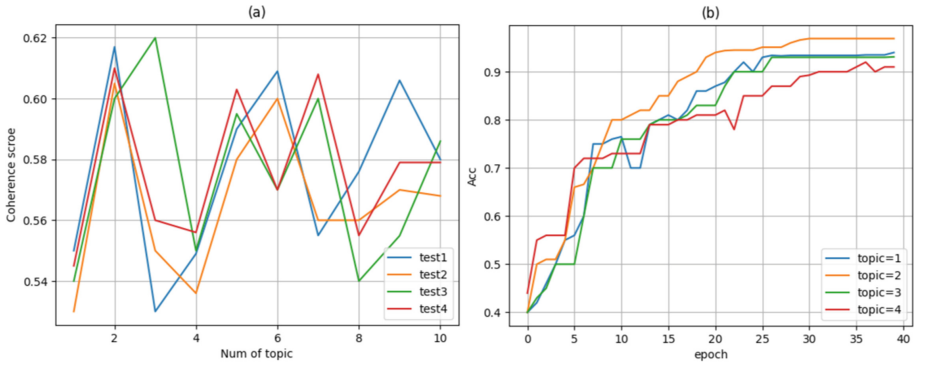

Fig. 7. (a) The number of topic words and the corresponding coherence score; (b) The influence of the number of topic words on the detection accuracy

Table 2. Results of threat event detection

	Accuracy	Recall	**F1 score**
Contains threat events	0.969	0.977	0.973
No threats included	0.970	0.975	0.972
Contains threat events (without keywords)	0.932	0.953	0.942
No threats included (without keywords)	0.940	0.952	0.946

a) **Threat event detection** The first experiment is mainly to evaluate the accuracy, recall, and F1 value of threat event detection in the dataset to prove the effectiveness of the model in detecting threat tweets. In the experiment, 1812 tweets containing threat information were used as positive samples, 2000 tweets that did not contain threat were used as negative samples, and a 10-fold crossover was used for verification experiments. In addition, because the experimental data was searched by some special keywords, we considered that the detection model of threatening tweets may be affected by some trigger words, so we deleted the keyword seeds in the 900 positive samples (such as xss, XSS, CSRF) as a control group to verify the robustness of the entire detection model. The experimental results are shown in Table 2.
It can be seen from Table 2 that the tweet detection model related to cross-site scripting threats has a good classification effect, and can accurately identify both positive and negative samples. Moreover, after we remove the keyword seeds with strong features, the detection accuracy, and recall of our model can exceed 93%, although the accuracy drops slightly. This indicates that these keywords are indeed more informative in terms of semantic representation and that the detection model we designed is robust.

Table 3. Results of control experiment

id	Model component	Accuracy	Recall	F1 score
1	Word2vec+BiLSTM	0.864	0.843	0.853
2	Word2vec+BiLSTM+Self-Att	0.898	0.890	0.894
3	Word2vec+LDA+BiLSTM+Self-Att	0.912	0.910	0.911
4	Mult-W2V+BiLSTM	0.940	0.934	0.935
5	Mult-W2V+BiLSTM+Self-Att	0.959	0.954	0.956
6	Mult-W2V+LDA+BiLSTM+Self-Att	**0.969**	**0.976**	**0.972**

b) **Number of topics** In the process of multi-dimensional feature fusion, we have introduced topic words. Only the appropriate number of topic words have better coherence and feature expression meaning. Too many or too few topics will affect the performance of the model. The Coherence Score is used to indicate the interpretability, meaning, and semantic coherence of the

selected topic. The higher the score, the better. The result of the experiment is shown in Fig. 7.

The influence of the number of topic words on the detection accuracy. As shown in Fig. 7(a), we have done several experiments on the coherence score, and selected part of the data for graphing. It can be seen that the score reaches the highest when the number of topic words is 2 to 3. Our sample data is only aimed at cross-site scripting types of threats, with fewer topics, so it is more appropriate to choose 2 topic words as the multi-dimensional feature fusion segment; In addition, from (b), it can be seen that the number of different subject words has a certain influence on the accuracy of model detection. Too few or too many subject words will affect the recognition of the model. When the model topic=2, the accuracy rate is relatively highest. And as the number of training iterations increases, the accuracy of the model increases the fastest.

c) **Model component** For the same data set (without removing the keyword seed), to verify the impact of different feature engineering and neural network structures on the event threat detection model. We used the controlled variable method in the experiment and conducted multiple sets of controlled experiments to prove that the combination of multi-dimensional feature fusion, recurrent neural network, attention mechanism and other methods we used is effective and can improve the accuracy of model detection. The experimental results are illustrated in Table 3.

In Table 3, Mult-W2V represents the combination of three different word embedding methods (Word2vec, GloVe, FastText), and Self-Att represents the self-attention mechanism. It can be seen from Table 3 that we used Word2vec as a control group. From the results of experiments 1, 2, and 3, the self-attention mechanism can effectively improve the detection efficiency of the model. Compared with experiments No. 1 to No. 3 and No. 4 to No. 6, the use of multi-dimensional coding can enable the model to obtain more features and significantly improve detection accuracy. Compared with experiments No. 2, 3, and No. 5, 6, the addition of topic words can be more conducive to obtaining the information characteristics that the author intended to express from the tweets, thereby improving the performance of the model. Therefore, the third experiment uses multiple control groups to prove that the model components used in our proposed model are meaningful and can effectively help the model detect tweets involving threats.

5 Conclusion

In this paper, aiming at the growing threat information on the Internet, we propose a threat event detection model based on deep learning. By using real data on Twitter to train the model, it proves that threats such as cross-site scripting attacks can be effectively detected. To extract feature information in short texts, the model adopts the fusion of multi-type word vector features and tweet topic words to obtain the hidden information as comprehensively as possible. Then

use the bidirectional recurrent neural network and self-attention mechanism to train the data. Ultimately, experiments are designed according to the characteristics of threat event detection. In the results of multiple sets of comparative experiments, it is proved that the detection model proposed in this paper has good performance in terms of parameter selection, model robustness, and detection accuracy. In future work, the threat scope of model detection can also be extended to other aspects, not just limited to cross-site scripting threats, we believe that they are essentially the same.

Acknowledgements. This work was supported in part by National Natural Science Foundation of China (U20B2045).

References

1. Marsh, M,. et al.: The Global Risks Report 2021, 21 December 2021. https://www.oliverwyman.com/content/dam/mmc-web/insights/publications/2021/january/global-risks-report/The-Global-Risks-Report-2021-small-FINAL.pdf
2. Noor, U., Anwar, Z., Amjad, T., Choo, K.K.R.: A machine learning-based FinTech cyber threat attribution framework using high-level indicators of compromise. Future Gener. Comput. Syst. **96**, 227–242 (2019)
3. Yagcioglu, S., et al.: Detecting Cybersecurity Events from Noisy Short Text. arXiv (2019). arXiv:1904.05054
4. Internet Security Center. 2020 Global Advanced Persistent Threat APT Research Report, February 2021. https://zt.360.cn/1101061855.php?dtid=1101062360&did=211138962
5. OWASP, OWASP Top 10–2021. https://owasp.org/Top10/. 2021-12-21
6. Ruder, S.: An overview of multi-task learning in deep neural networks. arXiv (2017). arXiv:1706.05098
7. Willett, P.K., Kirubarajan, T.: System diagnosis and prognosis: security and condition monitoring issues III. In: System Diagnosis and Prognosis: Security and Condition Monitoring Issues, vol. III, p. 5107 (2003)
8. Qiu, X., Lin, X., Qiu, L.: Feature representation models for cyber attack event extraction. In: 2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops (WIW), pp. 29–32. IEEE (2016)
9. Khandpur, R.P., Ji, T., Jan, S., et al.: Crowdsourcing cybersecurity: cyber attack detection using social media. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1049–1057 (2017)
10. Le Sceller, Q., Karbab, E.M.B., Debbabi, M., et al.: Sonar: automatic detection of cyber security events over the twitter stream. In: Proceedings of the 12th International Conference on Availability, Reliability and Security, pp. 1–11 (2017)
11. Bose, A., Behzadan, V., Aguirre, C., et al.: A novel approach for detection and ranking of trendy and emerging cyber threat events in twitter streams. In: 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 871–878. IEEE (2019)
12. Ji, T., Zhang, X., Self, N., et al.: Feature driven learning framework for cybersecurity event detection. In: Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 196–203 (2019)

13. Trong, H.M.D., Le, D.T., Veyseh, A.P.B., et al.: Introducing a new dataset for event detection in cybersecurity texts. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 5381–5390 2020)
14. Shin H, Shim W C, Moon J, et al. Cybersecurity Event Detection with New and Re-emerging Words. In: Proceedings of the 15th ACM Asia Conference on Computer and Communications Security, pp. 665–678 (2020)
15. Simran, K., Balakrishna, P., Vinayakumar, R., Soman, K.P.: Deep learning approach for enhanced cyber threat indicators in twitter stream. In: Thampi, S.M., Martinez Perez, G., Ko, R., Rawat, D.B. (eds.) SSCC 2019. CCIS, vol. 1208, pp. 135–145. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-4825-3_11
16. Cassel, M., Lima, F.: Evaluating one-hot encoding finite state machines for SEU reliability in SRAM-based FPGAs. In: 12th IEEE International On-Line Testing Symposium (IOLTS 2006). IEEE (2006). 6 pp
17. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. Adv. Neural. Inf. Process. Syst. **26**, 3111–3119 (2013)
18. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
19. Joulin, A., Grave, E., Bojanowski, P., et al.: Bag of tricks for efficient text classification. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, vol. 2, pp. 427–431
20. Sun, F., Chen, H.: Feature extension for Chinese short text classification based on LDA and word2vec. In: 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA), pp. 1189–1194. IEEE 2018
21. Sutskever, I., Vinyals, O., Le Quoc, V.: Sequence to sequence learning with neural networks. In: Proceedings of the 28th Annual Conference on Neural Information Processing Systems-NIPS, Montreal, QC, Canada, 8–13 December 2014, pp. 3104–3112 (2014)
22. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw. **18**(5–6), 602–610 (2005)
23. Li, X., Zhang, W., Ding, Q.: Understanding and improving deep learning-based rolling bearing fault diagnosis with attention mechanism. Signal Process. **161**, 136–154 (2019)
24. Zhao, H., Jia, J., Koltun, V.: Exploring self-attention for image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10076–10085 (2020)
25. Yin, X., Zhao, H., Zhao, J.B.: Military named entity recognition based on multi-neural network cooperation. Tsinghua Univ. J. (Nat. Sci. Ed.) **60**(8), 648–655 (2020)