SQL Injection Attack Detection and Prevention Techniques Using Deep Learning

Ding Chen*, Qiseng Yan, Chunwang Wu, Jun Zhao
School of Cyberspace Security, Chengdu University of Information Technology, Chengdu 610225, China

*chending@cuit.edu.cn

Abstract: Web application brings us convenience but also has some potential security problems. SQL injection attacks topped the list of Top 10 Network Security Problems released by OWASP, and the detection technology of SQL injection attacks has been one of the hotspots of network security research. In this paper, we propose a SQL injection detection method that does not rely on background rule base by using a natural language processing model and deep learning framework on the basis of comprehensive domestic and international research. The method can improve the accuracy and reduce the false alarm rate while allowing the machine to automatically learn the language model features of SQL injection attacks, greatly reducing human intervention and providing some defense against 0day attacks that never occur.

Keywords: SQL Injection Attack, Deep Learning, Word2Vector, CNN, MLP.

1. Introduction
With the rapid development of Internet technology, network information has exploded. Web applications bring us convenience but also face major network security challenges. At the end of 2016, Qihoo 360 conducted security tests on 1.979 million websites in China and found that 46.3% of web applications had security vulnerabilities, with SQL injection attack (SQLIA) and cross-site scripting attack (XSS) vulnerabilities accounting for the highest percentage.

As one of the most common network security vulnerabilities, SQL injection attacks cannot be ignored. In April 2011, Sony's Play Station Network was attacked by SQL injection. More than 77 million accounts were affected, of which 12 million credit cards were stolen. Information such as user accounts, passwords, addresses, credit card spending records was leaked, which indirectly caused Sony to lose up to 170 million US dollars. In February 2017, Russian hacker "Rasputin" used SQL injection vulnerabilities to gain super access to the database server and successfully invaded the system. A large amount of sensitive information was stolen in more than 20 universities and government agencies in the United Kingdom and the United States.

Theoretically, any database-driven Web application system may be at risk from SQL injection attacks. Because the SQL injection attack is no different from a user's normal access to the system, it can be achieved simply by submitting Web forms, query strings, or page requests, and is more covert, while the current Web application firewall (WAF) based on feature matching algorithms (rule base) is difficult to cover all variants of the SQL injection attack. Therefore, there are no solutions that can
prevent or detect all the different types of SQLIA. Recently, researchers try to benefit from AI techniques to propose more sophisticated solutions.

In this paper, we have proposed a lightweight approach to prevent SQLIA by employing word embedding and CNN, MLP. We denoise and decode HTTP requests, then use Word2vec to generate word embedding of these decoded characters, train a CNN, MLP classifier, finally use the classifier to identify the malicious request.

The paper is structured as follows: Section 2 reviews the related works about research in this area. We detail our approach in Section 3 and present the experimental results in Section 4. Finally, we conclude the paper with a note on future directions of research in Section 5.

2. Relation Works

SQL (Structured Query Language) is a database language that is used to add, delete, modify, and query data in a relational database. As long as the system uses the database, most of it interacts with the database through SQL statements.

The main reason for SQL injection vulnerabilities is that when developers develop code, they use the method of concatenating strings to construct SQL statements that are passed to the database. As a result, attackers can change the SQL statement by entering SQL keywords or special symbols. As a result of the execution, the system is attacked. The fundamental reason is to trust the data submitted by users too much, without filtering user input, and failing to perform reasonable verification on the server-side, so as to achieve the attacker's intended purpose, such as stealing sensitive system information and obtaining server control authority. Figure 1 shows the basic principles and processes of a SQL injection attack.

![Fig. 1 SQL Injection Attack Process](image)

From the emergence of SQL injection to the present, a large number of research scholars have conducted research and analysis on SQL injection detection problems from different angles and methods. At present, the methods of preventing SQL injection are mainly divided into two categories. One is to consider security factors when coding, and the other is to detect SQL injection attacks through additional measures during program operation to protect operational security. The second method includes black box and white box testing, pattern matching method, abstract syntax tree, sequence comparison, static analysis, and dynamic analysis. There are also scholars who have applied research techniques from other fields, such as research text analysis, data mining, machine learning, and other methods of SQL injection detection, and have achieved some results. Ouarda et al. propose a web page similarity analysis method for detecting and defending against SQL injection attacks [1]; Anamika Joshi and Karnataka Surathkal et al. proposed a method for detecting SQL injection attacks using a Bayesian algorithm [2]; Romil Rawat et al. used SVM in machine learning as a classification criterion to predict SQL injections, and the algorithm achieved 96.4% accuracy for detecting SQL
injections [3]; Zhichao Zhang's team proposed a neural network-based SQL injection vulnerability detection model [4]; a second-order injection detection method based on machine learning was proposed by Le Deguang et al [5].

3. The Proposed Approach
With the continuous escalation of attack methods, traditional filtering systems and WAF face many problems. The benefit to the improvement of computing power and the rapid development of deep learning technology, the goal of this article is to select a suitable deep learning framework through experimental analysis to detect whether HTTP requests contain malicious code for SQL injection attacks, so as to improve the accuracy of detection and reduce the false alarm rate.

Typically an attacker will bypass the WAF and filtering system by wrapping the injected attack code in various ways. Therefore, before model training, the training data first needs to be cleaned using recursive decoding, then the lexical analysis is used to generate word vector models, and finally input into the neural network model for training. In this paper, we chose a convolutional neural network (CNN) and a multilayer perceptron (MLP) to compare the experiments. Once the model is trained, the model accuracy is verified by testing the dataset, and the training process is visualized using the Tensorboard feature of the Tensorflow framework to evaluate the speed and ease of use of the model. The overall structure of the testing system is shown in Figure 2.

![System Framework Diagram](image)

**Fig. 2 Principle diagram of the system framework**

Visual teaching supervision: through the online system, the supervising teacher can check the actual situation of the class in each classroom and evaluate the class online, which can effectively reduce the impact on the teaching teachers and improve the efficiency of teaching supervision.

Intelligent question-answering system: based on big data processing, knowledge graph, intelligent decision-making, man-machine integration, and so on, it provides accurate answers to various questions raised by students in the teaching of professional courses, effectively assists teachers in classroom teaching, and enhances students' learning effect.

3.1. Data Cleaning
In the user's HTTP request data traffic, it may contain various different encodings such as urllencode, querystring, JSON, PHP serialize, base64, etc. Before training, we need to do all the decoding until the input is accepted by the final application, called payload. When an input is decoded once, you can't exit the loop decoding directly, because this input can also decode other payloads through other decoding methods, and these payloads maybe SQL injection statements, so you should treat these payloads as a new input and recursively decode them again, which is the important reason why SQL injection is difficult to prevent.

After decoding, to make the features of the sample clearer, generalization is also needed to reduce the influence of numbers and some extraneous factors on the sample. The rule is to replace the number with "0" and the url with "http://u", and then split the word.
Undecoded sample data:
/myhome/do.php?ac=71ee30ae117cddace55bd01714904227%20%2C%20%28SELECT%20%2C%20
ASE%20WHEN%20%288281%3D8281%29%20THEN%20%20ELSE%20%28SELECT%20%20FROM%20%28SELECT%20%20UNION%20%20SELECT%20%20%20%29%20%20END%29%29

List generated after decoding and generalization process:
['myhome', 'do.php', 'ac=', '0ee0ae0cddace0bd0', 'select', 'case', 'when', '0=', '0', '}', 'then', '0', 'else', 'select', '0', 'from', 'select', '0', 'union', 'select', '0', '}', 'x', '}', 'end', '}', '}

3.2. Training word vector model
Word vectors are very important tools in natural language processing, word vectors represent words as a fixed-length continuous dense vector. Word vectors have the following advantages: 1. can represent the similarity between words, because there is a distance between the vectors, which directly represents the degree of similarity between two words. 2. contains more information, each dimension has a meaning, equivalent to the composition of the word each component[6]. Word2Vector is Google's open source word vector generation tool, which has two learning modes, CBOW and Skip-gram. In our method we use CBOW model to generate word embedding. The vector dimension is 16. The differences between the two models is that CBOW predicts the middle word by the semantics before and after, while Skip-gram does the opposite, predicting the words on either side by the middle word. Figure 3 is the structure of CBOW model.

3.3. Convolutional Neural Network
CNN is a feedforward neural network model. In this paper, the CNN model is composed of three convolutional layers and three pooling layers and finally connected to the fully connected layer. The first convolutional layer, 16 convolutional kernels, size 3*3, the second convolutional layer, 32 convolutional kernels, size 4*4, the third convolutional layer, 64 convolutional kernels, size 5*5, and the convolutional mode are all same. Two activation functions are used in the CNN, the convolutional and fully connected layers use the ReLu function as the activation function, while the output layer uses the Softmax function as an activation function[7]. Figure 4 is the Visualization of the CNN model training process.
Fig. 4 Accuracy and loss value of the CNN training process

3.4. Multi-Layer Perceptron
Multilayer Perceptron (MLP) is an artificial neural network with a forward structure that maps a set of input vectors to a set of output vectors. MLP can be regarded as a directed graph consisting of multiple node layers, each layer is fully connected to the next layer. Except for input nodes, each node is a neuron (or processing unit) with a nonlinear activation function. The multilayer perceptron used in this paper is designed with only two hidden layers, and the activation functions are also selected as ReLu and Softmax functions. Figure 5 is the Visualization of the MLP model training process.

Fig. 5 Accuracy and loss value of the MLP training process

4. Experiments and Comparison
The SQL injection detection model designed in this paper does not require high machine performance. The experimental setup consisted of a notebook computer with Intel Core-i7 CPU and 16GB RAM, running with Ubuntu16.04. The programming language is Python3, using the Keras framework based on TensorFlow 2.0.

4.1. dataset
The dataset is divided into training and test data, and each dataset consists of two parts: white samples and black samples. The training data includes 25487 samples of SQL injection from the internet as negative examples and 24500 samples of the normal HTTP request as positive examples. We specifically selected a data set for testing during data processing that contained 4000 SQL injection data and 4000 normal data.

4.2. Experiments Result
As shown in Table 1, this is the confusion matrix for the CNN model.

| Actual Class | Predicted Class |
|--------------|-----------------|
|              | Positive | Negative |
| Positive     | TP = 3963 | FN = 37 |
| Negative     | FP = 103  | TN = 3897 |

Tab. 1 Confusion matrix of CNN

As shown in Table 2, this is the value of accuracy, precision, recall for the CNN model.

| Performance                  | Value   |
|------------------------------|---------|
| Accuracy = (TP + TN) / (TP + FP + FN + TN) | 0.982500 |
| Precision = TP / (TP + FP)     | 0.974668 |
| Recall = TP / (TP + FN)        | 0.990750 |
| F1 = 2*(Recall * Precision) / (Recall + Precision) | 0.982643 |
| Normal Test Cost Time(s)       | 26.09317 |
Negative Test Cost Time(s) 26.70283

**Tab. 2** Performance of CNN
As shown in Table 3, this is the confusion matrix for the MLP model.

| Actual Class | Predicted Class |
|--------------|-----------------|
|              | Positive | Negative |
| Positive     | TP = 3969  | FN = 31   |
| Negative     | FP = 83    | TN = 3917 |

**Tab. 3** Confusion matrix of MLP
As shown in Table 4, this is the value of accuracy, precision, recall for the MLP model.

| Performance                           | Value |
|---------------------------------------|-------|
| Accuracy = (TP + TN) / (TP + FP + FN + TN) | 0.985750 |
| Precision = TP / (TP + FP)             | 0.979516 |
| Recall = TP / (TP + FN)                | 0.992250 |
| F1 = 2*(Recall * Precision) / (Recall + Precision) | 0.985842 |
| Normal Test Cost Time(s)               | 12.56789 |
| Negative Test Cost Time(s)             | 13.28720 |

**Tab. 4** Performance of MLP

4.3. Model Comparison
After comparing the tests of the two models, it can be found that different neural networks will have different applicability scenarios and the performance will be very different.

The CNN and MLP models chosen in this paper both perform well for SQL injection attack detection, as shown in Figure 6, which is a comparison of the training process of the two network structures.

![Fig. 6](image)

**Fig. 6** Comparison of CNN and MLP models in terms of accuracy and loss values

From the test results, MLP uses about 12s more to detect 4000 data, however, CNN uses about 26s, which is nearly twice as much time. From the analysis of the model summary, we can see that in MLP we have only two hidden layers and the total number of model parameters is only 695,875, while the CNN model has three sets of convolutional and pooling layers, and the full connection between the layers is used.
5. Conclusions
This paper implements a SQL injection detection system based on a deep learning framework and combining data preprocessing and lexical analysis techniques. The experiments show that the system can detect first-order SQL injection attacks more accurately and efficiently. Subsequent research will focus on advanced SQL injection attack methods, such as second-order injection and hybrid attacks.

Acknowledgments
This work was supported by The Open Project of Sichuan Provincial Department of Education (No.18ZB0120), The Open Project of Network and Data security Key Laboratory of Sichuan Province (No. NDS2018-1).

References
[1] O. Lounis, S. E. B Guermeche, L. Saoudi and S. E. Benaicha 2014 Science and Information Conf.(2014 London) 1 pp 589-594
[2] Anamika Joshi and V Geetha 2014 Int. Conf. on Control, Instru., Comm. and Comput. Tech. (2014 Kumaracoil) D pp 1111-1115
[3] Romi R and Kumar S 2012 Int. J. Comput. Appl. vol 42(13) pp 0975–8887
[4] Zhichao Z 2016 Computers and Modernization. C 10 pp 67-71
[5] Deguang L 2017 Journal of Communication C 3 pp 201528501-201528505
[6] Lu Y, Senlin L and Limin P 2019 Int. Conf. on Computer Engineering, Information Science & Application Technology, 3rd. (2019 Chongqing) D 10 p2991
[7] Xie C, Ren Y, Fu J Xu and J Guo 2019 IEEE Access C 7 pp 151475-151481