Analyzing the Effectiveness of Machine Learning Techniques in Detecting Attacks in a Big Data Environment

Omar Dhafer Madeeh[1,*], Osamah M. Abduljabbar[1] , Huda Mohammed Lateef[1]
[1]Electronic Computer Center, University of Fallujah, Anbar, Iraq
Emails: omar.dhafer@uofallujah.edu.iq; almihamdi@uofallujah.edu.iq; Hudamohammedlateef@uofallujah.edu.iq

Abstract

Protecting big data has become an extremely vital necessity in the context of cybersecurity, given the significant impact that this data has on institutions and clients. The importance of this type of data is highlighted as a basis for decision-making processes and policy guidance. Therefore, attacks on this data can lead to serious losses through illicit access, resulting in a loss of integrity, reliability, confidentiality, and availability of this data. The second problem in this context arises from the necessity of reducing the attack detection period and its vital importance in classifying malicious and non-harmful patterns. Structured Query Language Injection Attack (SQLIA) is among the common attacks targeting data, which is the focus of interest in the proposed model. The aim of this research revolves around developing an approach aimed at detecting and distinguishing patterns of loads sent by the user. The proposed method is based on training a model using random forest technology, which is considered one of the machine learning (ML) techniques while taking advantage of the Spark ML library that interacts effectively with big data frameworks. This is accompanied by a comprehensive analysis of the effectiveness of ML techniques in monitoring and detecting SQLIA. The study was conducted using the SQL dataset available on the Kaggle platform and showed promising results as the proposed method achieved an accuracy of 98.12%. While the proposed approach takes 0.046 seconds to determine the SQL type. It is concluded from these results that using the Spark ML library based on ML techniques contributes to achieving higher accuracy and requires less time to identify the class of request sent due to its ability to be distributed in memory.

Keywords: Big data; Spark ML; SQL Injection; Random Forest

1. Introduction

With the significant increase in the volume of data exchanged by companies and users in various domains, safety has become a fundamental factor in evolving web applications. Hence, enterprise big data requires web application architectures capable of detecting and stopping potential application defects. According to the Open Application Security Project (OWASP), SQLI is considered one of the most serious challenges in data targeting [1] [2].

The Big Data major relies on an interdisciplinary approach to data analysis and forecasting, combining the branches of Information technology (IT), mathematical representation, and Data analysis. Accessibility to data and developing strategies for doing with it have become crucial factors in this context. Companies can improve and handle big data more reliably by adopting and applying artificial intelligence (AI) technologies [3].

DOI: https://doi.org/10.54216/JCIM.150221
Received: May 20, 2024 Revised: July 21, 2024 Accepted: November 09, 2024

The continuing increase in security risks is increasingly linked to the expanding use of online data storage, as these risks stem from the increasing prevalence of attacks aimed at unauthorized access to the personal information of individuals and security structures[4],[5]

In the context of database servers, SQLIA stand out as one of the most dangerous attacks. Exploiting weak points allows unauthorized people to access and attack user data, leading to it being stolen, modified, read, or preventing the user from accessing his data [6],[7]
These sections chart the structure of the research paper, with a prominent focus in the second section on the proposed approach to identifying SQL queries within the context of a big data environment. The third section summarizes the findings drawn from the study, while the research concludes with the final section containing the conclusions and recommendations emerging from this research work.

2. Methodology

In the context of developing ML methodologies, the work involves a sequence of stages. This process begins with collecting basic data appropriate to deal with the challenge of a study, and this stage is followed by the data processing stage. This stage is concerned with preparing the data to make it understandable and ready for use later using ML methodologies. Later, the data is divided into two parts: samples for training and samples for testing, in order to They are used to build and test the model. This phase is followed by the phase of executing the ML algorithms on the training samples, and the process is completed by a phase of testing and evaluating the efficiency of the approach using the second part of the dataset.

A. Description of the SQL-i Dataset

When developing ML methodologies, the stage of obtaining a dataset related to the focus of the study is an essential step in developing ML methodologies. In this study, the dataset used included 109,518 samples, containing loads with attack intent and normal loads. However, this data appears to be inaccurate and unable to be used within ML methodologies. Therefore, inaccurate data were cleaned and removed, and this process reduced the size of the dataset to 85,974 samples. Table (1) describes the dataset used in this research paper:

Table 1: Comprehensive Overview of the Dataset

Total samples	Total training samples	Total test samples	Benign types	Harmful types
85974	68604	17370	45051	40923

B. Data pre-processing

At this stage, the dataset is processed for use in ML methodologies. This stage involves the process of selecting the best samples, establishing links between the features of the dataset, standardizing, and removing duplicates and missing values from the dataset [8].

In this proposal, CountVectorizer was employed in the process of transforming textual into digital features. The words in these data represent certain attributes and enable us to create a matrix containing a set of words [9]. This approach takes into account the frequency of vocabulary in the text, as CountVectorizer turns the text into a matrix that shows the frequency of vocabulary, giving users the ability to count the frequency of each word [10].

C. Dataset Segmentation Methodology

The third stage in building ML methodologies involves dividing the data into subset categories using the hold-out method. The first category represents 80% of the dataset to build and train the proposed approach, while the second category, which includes 20%, is used to test and evaluate the proposed approach [11].

DOI: https://doi.org/10.54216/JCIM.150221
Received: May 20, 2024 Revised: July 21, 2024 Accepted: November 09, 2024

D. Predictive Strategy

When building a predictive strategy using ML, the focus at this stage should be on choosing the appropriate classification technique. In this paper, we used RF technology, one of the supervised ML techniques, to classify these requests into two categories: zero and one, where zero represents normal payloads and one represents requests containing an attack. In a massive classification of large amounts of data, using a single classifier may not be effective and the level of classification accuracy may decrease. As a result, classification algorithms such as Decision Trees are preferred for those applications that require the classification of large amounts of data. Non-statisticians as well as IT professionals easily understand RF. The RF method is not complicated and often does not require extensive verification and excessive cost. The RF algorithm makes use of Adaboost and Bootstrapping techniques to create an ensemble of classifiers.
The RF algorithm develops decision trees through several stages:

- N is taken to be the total number of training data occurrences in the samples, and M is taken to be the number of features present in the input dataset.
- In the context of tree construction, m stands for the number of parameters used at each node to determine the next attribute, where m is less than M, which represents the total number of attributes.
- The stage consists of collecting training samples and building a subtree assigned to each case, where each tree is created based on the data available to it.
- M properties are specified for each node in the decision tree.
- Based on the characteristics of the data samples, the optimal partitioning is chosen.
- None of the trees are pruned during their growth stages [12] ,[13].

E. Proposed Model

The proposed model for specifying SQL requests in a big data environment is based on several stages. The primary goal of this framework is to use the RF algorithm to detect suspicious or abnormal patterns in the behavior of database queries, in order to stand up to and prevent potential malicious attacks. This approach consists of a set of stages, and the steps are defined below :

1) In its first stage, the approach begins by acquiring data containing attack and harmless payloads to train the proposed model
2) The second stage includes applying pre-treatment.
3) In the third stage, the program divides the data into two groups designated for training, testing, and evaluation.
4) In the fourth stage, the model uses the first part of the dataset to train the proposed model.
5) For the fifth stage, the second group is used to test and evaluate the model.
6) The sixth stage evaluates the approach using a set of metrics to determine performance efficiency.

DOI: https://doi.org/10.54216/JCIM.150221
Received: May 20, 2024 Revised: July 21, 2024 Accepted: November 09, 2024

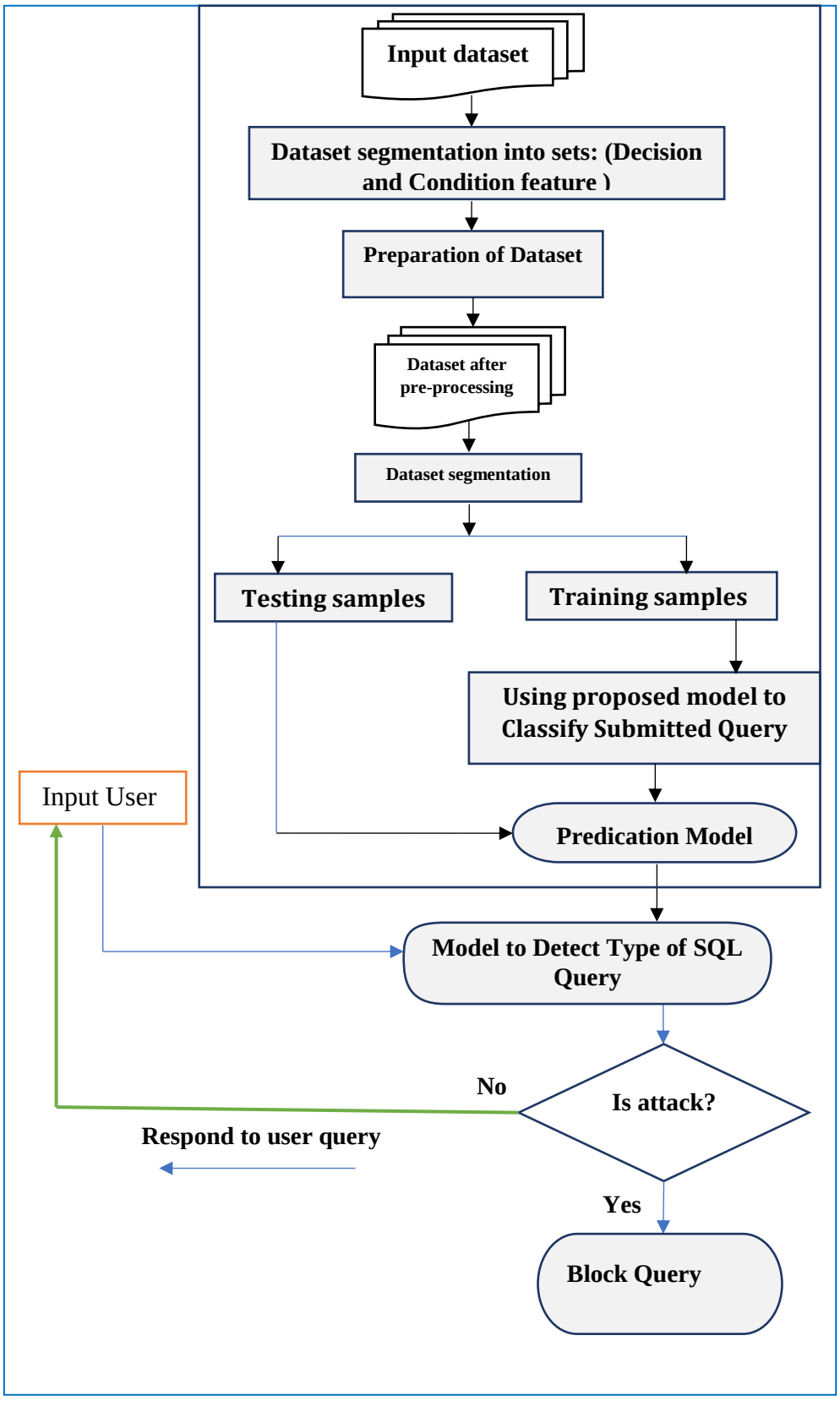

Figure 1. Shows the steps for implementing the proposed model.

DOI: https://doi.org/10.54216/JCIM.150221
Received: May 20, 2024 Revised: July 21, 2024 Accepted: November 09, 2024

F. Performance Evaluation Criteria

In the last stage of evolving the predictive approach, the performance of the proposed model is estimated using multiple criteria, such as accuracy, response time, precision, recall, etc., to evaluate the results and understand the effectiveness of the adopted approach.
The confusion matrix is used as a common evaluation tool, with variations in the values used to calculate these metrics. Table II presents a set of values namely false positives (FP), false negatives (FN), true negatives (TN), and true positives (TP) which represent the contents of the confusion matrix [14].

Table 2: Confusion Matrix

		Predict class	
		Class X	Class Y
True class	Class X	TN	FP
	Class Y	FN	TP

Measuring and evaluating the effectiveness of ML models depends on a set of different metrics that will be described below:

$$\textbf{Accuracy} = \frac{(TP+TN)}{(TP+TN+FP+FN)} * 100 \quad (1)$$

$$\textbf{Precision} = \frac{(TP)}{(TP+FP)} * 100 \quad (2)$$

$$\textbf{Recall} = \frac{(TP)}{(TP+FN)} * 100 \quad (3)$$

$$\textbf{F1 - score} = 2 * \frac{Precision*Recall}{Precision+Recall} * 100 \quad (4) \text{ [15][16]}$$

3. Results And Discussions

This part reviews the results of predicting SQL requests from client to server, using the RF classifier within the Big Data framework, which enables to determine the nature of the request, whether it is malicious or benign. The evaluation results are presented in Table (3).
The experiment included a dataset of 85,974 payloads, which were divided into benign payloads (45,051 payloads) and malicious payloads (40,923 payloads). A holdout method was used to distribute the data, with 80% selected for training and the remainder for experimentation and evaluation

Table 3: Result of Experiment

Seq	**Name of Metrics**	**Value**
	Time Duration for Test	0.046 seconds
	Accuracy criterion	98.12%

DOI: https://doi.org/10.54216/JCIM.150221
Received: May 20, 2024 Revised: July 21, 2024 Accepted: November 09, 2024

	Precision criterion	98.16%,
	Recall criterion	98.12%
	F1 Score criterion	98.12%.

The RF method classified SQL queries sent to databases used in web result of experiment applications with 98.12% accuracy, demonstrating its high ability to distinguish between malicious and benign payloads. The process of discovering the query type also took a short time, not exceeding 00.046 seconds.
The table (4) below shows the differences and comparison results between previous research and the new study.

Table 4: Analysis Resulting From Comparing Previous Research with the Current Study

Ref	Model	Accuracy	Time Complexity	Size of Dataset
[18]	Neural Network of Direct Signal Propagation	95	No time is indicated	30,233
[19]	LSTM	95.2	37.1494 sec	42,212
[20]	Support Vector Machine Gradient boosting Naive Bayes classifier REGEX classifier	94.92 94.27 70.79 97.48	3.98 sec	20474
[21]	Naive Bayes Random forest CNN SVM Passive Aggressive	95 92 97 79 79	No time is indicated	
[22]	CNN-BiLSTM	98	45 sec	4,200
	Proposed Model	**98.12%**	**0.046**	**85974**

After completing building the model, a significant increase in accuracy was recorded with a significant decrease in the time taken compared to previous models, as the previous table shows. These impressive results were achieved using the Spark ML library, which operates in a distributed in-memory manner, significantly reducing time and improving accuracy compared to previous research on dealing with big data.

4. Conclusion And Recommendations

Protecting big data is very essential in our digital society, as we must address data security threats, especially SQLIA, to ensure the safety and privacy of this vital data for organizations and individuals.

A. Research Contributions

- The main contribution of this study is to present an approach based on analyzing and classifying SQL requests sent by users, which identifies whether the requests are malicious or benign. This approach is based on random

DOI: https://doi.org/10.54216/JCIM.150221
Received: May 20, 2024 Revised: July 21, 2024 Accepted: November 09, 2024

forest technology with the help of the Spark ML library specialized in the field of big data. This approach aims to prevent unauthorized access to data by validating access to data before allowing access to it.

- The second contribution is to achieve a significant reduction in the time it takes to classify the request type, where the implementation of the Spark ML library is considered a paradigm shift. This contribution relies on the superior ability of the Spark ML library to operate in a distributed, in-memory manner, which significantly reduces the time spent in payload type classification. This enhances the speed of the process and gives an accurate estimate of the type of request without a negative impact on performance.

B. Recommendations

Traditional methods for detecting and preventing attacks show limited efficiency when dealing with small data, but lose their effectiveness when dealing with big data. The current study presents a new approach based on RF technologies and the distributed Spark ML platform to detect SQLIA in real-time and this approach is considered a basic protection layer between the user and the database. This approach improves classification accuracy and reduces the time it takes to anticipate attacks, enhancing real-time data security.
Evaluating and analyzing the effectiveness of ML models in SQLIA discovery or any other aspect of scientific fields depends on points including:

- The number of false positives and false negatives. These values are among the basic values whose percentage must be reduced, as when the percentage of these values is very high, and then the model works less efficiently in prediction.
- The purpose of reducing the values of false positives and false negatives is to increase the effectiveness of the model and allow the user to access the data while maintaining the privacy of the data and its availability at the required time for institutions and individuals.
- The time factor: The time factor is considered one of the important elements when dealing with ML models, as the speed of time in determining the type of request is an important factor in the availability of data.

Therefore, when dealing with ML or deep learning models, the number of false positives and false negatives, as well as time and classification accuracy, must be taken into account.

References

[1] A. H. Farhan and R. F. Hasan, "Detection SQL Injection Attacks Against Web Application by Using K-Nearest Neighbors with Principal Component Analysis," in Proceedings of Data Analytics and Management: ICDAM 2022, Springer, pp. 631–642,2023.

[2] K. N. Durai, R. Subha, and A. Haldorai, "A Novel Method to Detect and Prevent SQLIA Using to Cloud Web Security," Wirel. Pers. Commun., vol. Ontology 117, no. 4, pp. 2995–3014, 2021, doi: 10.1007/s11277-020-07243-z.

[3] A. Haldorai, S. Devi, R. Joan, and L. Arulmurugan, "Big Data in Intelligent Information Systems," Mob. Networks Appl., no. October 2021, pp. 997–999, 2022, doi: 10.1007/s11036-021-01863-w.

[4] M. J. Awan et al., "Real-time ddos attack detection system using big data approach," Sustain., vol. 13, no. 19, pp. 1–19, 2021, doi: 10.3390/su131910743.

[5] A. H. Farhan and R. F. Hasan, "Using random forest with principal component analysis to detect SQLIA," in AIP Conference Proceedings, 2023.

[6] M. Alghawazi, D. Alghazzawi, and S. Alarifi, "Detection of SQL Injection Attack Using Machine Learning Techniques: A Systematic Literature Review," J. Cybersecurity Priv., vol. 2, no. 4, pp. 764–777, 2022, doi: 10.3390/jcp2040039.

[7] O. S. F. Shareef, R. F. Hasan, and A. H. Farhan, "Analyzing SQL payloads using logistic regression in a big data environment," J. Intell. Syst., 2023, [Online]. Available: https://doi.org/10.1515/jisys-2023-0063

[8] S. A. Alasadi and W. S. Bhaya, "Review of data preprocessing techniques in data mining," J. Eng. Appl. Sci., vol. 12, no. 16, pp. 4102–4107, 2017.

[9] H. El Rifai, L. Al Qadi, and A. Elnagar, "Arabic text classification: the need for multi-labeling systems," Neural Comput. Appl., vol. 34, no. 2, pp. 1135–1159, 2022, doi: 10.1007/s00521-021-06390-z.

[10] J. S. Yang, C. Y. Zhao, H. T. Yu, and H. Y. Chen, "Use GBDT to Predict the Stock Market," Procedia Comput. Sci., vol. 174, no. 2019, pp. 161–171, 2020, doi: 10.1016/j.procs.2020.06.071.

[11] M. Rafało, "Cross validation methods: Analysis based on diagnostics of thyroid cancer metastasis," ICT Express, vol. 8, no. 2, pp. 183–188, 2022, doi: 10.1016/j.icte.2021.05.001.

[12] A. B. Shaik and S. Srinivasan, A brief survey on random forest ensembles in classification model, vol. 56. Springer Singapore, 2019. Doi: 10.1007/978-981-13-2354-6_27.

DOI: https://doi.org/10.54216/JCIM.150221
Received: May 20, 2024 Revised: July 21, 2024 Accepted: November 09, 2024

[13] O. D. Madeeh and H. S. Abdullah, "An Efficient Prediction Model based on Machine Learning Techniques for Prediction of the Stock Market," J. Phys. Conf. Ser., vol. 1804, no. 1, 2021, doi: 10.1088/1742-6596/1804/1/012008.
[14] I. S. I. Abuhaiba and H. M. Dawoud, "Combining different approaches to improve Arabic text documents classification," Int. J. Intell. Syst. Appl., vol. 9, no. 4, pp. 39–52, 2017, doi: 10.5815/ijisa.2017.04.05.
[15] F. K. Alarfaj, "applied sciences Enhancing the Performance of SQL Injection Attack Detection through Probabilistic Neural Networks," 2023.
[16] R. F. Hasan, O. S. F. Shareef, and A. H. Farhan, "Analysis of the False Prediction of the Logistic Regression Algorithm in SQL Payload Classification and its Impact on the Principles of Information Security (CIA)," Iraqi J. Comput. Sci. Math., vol. 4, no. 4, pp. 191–203, 2023, doi: 10.52866/ijcsm.2023.04.04.015.
[17] S. O. Uwagbole, W. J. Buchanan, and L. Fan, "Applied Machine Learning predictive analytics to SQL Injection Attack detection and prevention," Proc. IM 2017 - 2017 IFIP/IEEE Int. Symp. Integr. Netw. Serv. Manag., pp. 1087–1090, 2017, doi: 10.23919/INM.2017.7987433.
[18] O. Hubskyi, T. Babenko, L. Myrutenko, and O. Oksiiuk, "Detection of sql injection attack using neural networks," Adv. Intell. Syst. Comput., vol. 1265 AISC, pp. 277–286, 2021, doi: 10.1007/978-3-030-58124-4_27.
[19] P. Tang, W. Qiu, Z. Huang, H. Lian, and G. Liu, "Detection of SQL injection based on artificial neural network," Knowledge-Based Syst., vol. 190, p. 105528, 2020, doi: 10.1016/j.knosys.2020.105528.
[20] B. Kranthikumar and R. L. Velusamy, "SQL injection detection using REGEX classifier," J. Xi'an Univ. Archit. Technol., vol. Volume XII, no. Issue VI, pp. 800–809, 2020.
[21] A. Joshi and V. Geetha, "SQL Injection detection using machine learning," in 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT 2014, pp. 1111–1115, 2014, doi: 10.1109/ICCICCT.2014.6993127.
[22] P. Aggarwal, A. Kumar, K. Michael, J. Nemade, S. Sharma, and others, "Random Decision Forest approach for Mitigating SQL Injection Attacks," in 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), pp. 1–5,2021.

DOI: https://doi.org/10.54216/JCIM.150221
Received: May 20, 2024 Revised: July 21, 2024 Accepted: November 09, 2024