Published by
NIGERIAN SOCIETY OF PHYSICAL SCIENCES
Available online @ https://journal.nsps.org.ng/index.php/jnsps

J. Nig. Soc. Phys. Sci. 4 (2022) 832

Journal of the Nigerian Society of Physical Sciences

Countermeasure to Structured Query Language Injection Attack for Web Applications using Hybrid Logistic Regression Technique

Shehu Magawata Shagari[a,*], Danlami Gabi[a], Nasiru Muhammad Dankolo[a], Noah Ndakotsu Gana[b]

[a]*Department of Computer Science, Kebbi State University of Science and Technology, Aliero, Nigeria*
[b]*Department of Cyber Security Science, Federal University of Technology, Minna, Nigeria*

Abstract

The new generation of security threats has been promoted by real-time applications, where several users develop new ways to communicate on the internet via web applications. Structured Query Language injection Attacks (SQLiAs) is one of the major threats to web application security. Here, unauthorised users usually gain access to the database via web applications. Despite the giant strides made in the detection and prevention of SQLiAs by several researchers, an ideal approach is still far from over as most existing techniques still require improvement, especially in the area of addressing the weak characterisation of input vectors which often leads to low prediction accuracy. To deal with this concern, this paper put forward a hybrid optimised Logistic Regression (LR) model with Improved Term Frequency Inverse Document-Frequency (ITFIDF-LR). To show the effectiveness of the proposed approach, attack datasets is used and evaluated using selected performance metrics, i.e., accuracy, recall, specificity and False Positive Rate. The experimental results via simulation when compared with the benchmarked techniques, achieved performance record of 0.99781 for accuracy, recall and F1-score as well as 0.99782, 0.99409 and 0.00591 for precision, specificity and False Positive Rate (FPR) respectively. This is an indication that the proposed approach is efficient and when deployed is capable of detecting SQLiA on web applications.

DOI:10.46481/jnsps.2022.832

Keywords: Database management system, Logistic regression, SQL injection attack

Article History :
Received: 25 May 2022
Received in revised form: 07 August 2022
Accepted for publication: 08 August 2022
Published: 01 October 2022

Communicated by: J. Ndam

1. Introduction

The Internet, has witnessed tremendous growth over the past decade, making the web and various internet products of great research interest [1]. With this growth, the Netcraft Web Server overviews as at January 2019 estimated that there are over 1.8 billion sites and over 4.39 billion internet users. With these increased number of websites comes with increased risks of security challenges confronting websites and other web applications. Top on the list of security risks on web applications and sites is the Structured Query Language Injection

Attacks (SQLIAs) and are lethal and account for 51% of attacks [2]. The SQLIAs are attacks in which an attacker finds out the weakness of a web application and exploits the weakness by executing malicious statement through the Internet to

*Corresponding author tel. no: +234 8036256995
Email address: shagari1978@gmail.com (Shehu Magawata Shagari)

access private data from the database [3].

A database is a collection of interrelated and organised data [4]. The management of database is most significant to ensuring information are kept in secrecy while authorising only those that have permission to access resources within the database. To interact with the database, a query language such as Structured Query Language (SQL) is required.

Thus, Database Management System (DBMS) is therefore software that provides a functionality of creating database, maintaining or updating the data in the database [5]. The common types of DBMS are hierarchical, network, object-oriented and relational database models [6]. The unauthorised access of the database greatly endangers web applications as the database is a key component used by all web applications to store all the data required by the application [2]. SQLIAs occur when there is no validation of user inputs, cookies and input parameters, before they are passed to SQL queries that will be executed on the database.

There are three basic types of SQLIAs: Union Based SQL injection attack, Error Based SQL injection attack and Blind SQL injection attack [7]. Union Based SQLIAs are attacks where Union statement is used. That is, two statements are joined by the attackers to get information from the database, on the other hand, Error Based SQLIAs are perpetrated by attackers through querying the server with a view to causing error and using the error message to determine vulnerabilities that can be exploited to attack the system. The Blind SQLIAs are dubbed the hardest forms of SQLIAs as the attackers only extract data by querying the server. Blind SQLIAs can either be Boolean or time based [8-11]. When successful, these attacks enable attackers bypass the system authentication and gain control of the database and consequently private information [12, 13]. Thus, enabling attacker to change user's password, retrieve users, make illegal transaction, delete table or can damage the database as well as perpetrate several other illegalities.

The widespread use of SQLIAs has led to development of several methods for the detection and prevention of such attacks. Some of these methods can widely be seen e.g., in [14-22]. Although, these solutions have contributed immensely towards providing an understanding on how SQLIAs attacks do occur and addressed, however despites these giant leaps, an ideal solution is far from been achieved. With the negative impact of SQLIAs, several researchers in [23-29] have tried to address concerns arising from such attacks through provision of techniques to serve as potential solutions. Despites the giant strides made in the detection and prevention of SQLIAs, the following limitations still exist:

1. Weak characterisation of the input vector as witnessed by most machine learning based methods, leading to low accuracy and unsatisfactory recall rate.

2. Weak representation of attributes as witnessed by text vectorisation-based algorithms which leads to inaccurate description of keyword weight and consequently low prediction accuracy.

3. The use of SVM by some of the existing schemes reduces accuracy as they inherit the weaknesses of SVM which lacks the ability to perform well when the data set has more noise and the number of features for each data point exceeds the number of training data sample

Therefore, in this paper, we proposed an optimised logistic regression model based based-Improved Term-Frequency-Inverse-Document-Frequency (ITFIDF-LR) to serve as a potential solution. Simulation results show the proposed approach achieved performance record of 0.99781 for accuracy, recall and F1-score as well as 0.99782, 0.99409 and 0.00591 for precision, specificity and False Positive Rate (FPR) respectively. This is an indication that the proposed approach is efficient and when deployed is capable of detecting and preventing SQLiA in DBMS.

The contribution of this paper is as follows:

1. Determination of existing gaps through exploring existing techniques for countering SQL injection attacks from literatures,

2. An optimised logistic regression model for prevention of SQL injection attacks in DBMS,

3. An improved ITFIDF-LR approach for the detection of SQLiA in DBMS

The rest of this article is organised as follows. Section 2 provides discussion on related work. Discussion on IDF, TF-IDF and Logistic Regression are provided in Section 3. Section 4 discusses the Experimental Model with Proposed Testbed and Simulation. Performance Evaluation Metric were presented in Section 5. Section 6 is the results and discussion section while Section 7 concludes the paper.

2. Related Work

With the negative impact of SQLiAs, several researchers have tried to address concerns arising from such attacks through provision of techniques to serve as an ideal. Some of these researcher(s) and their techniques are discussed as follows:

Hassan et al. [23] proposed a deep neural network-based technique for the detection of SQL injection vulnerability. The proposed method which is targeted at addressing the challenging effects associated with financial loss in business, application compromise and administrative exploit claimed to have outperformed existing methods in detections of SQL injection attack with the accuracy of 98.04% over 1850 dataset records, however, improve performance can be recorded if a significantly available dataset is used for training the machine learning algorithms. Yu et al. [24] proposed a novel technique for the detection of SQL injection attacks, the technique encompasses tokenization, skin-gram model in word2vec and SVM algorithm with eigenvectors, the proposed model was said to have obtained an effective means of detection of SQL injection attack with an unspecified claimed good accuracy, 0.35%, and 1.9% was also achieved for FPR and FNR respectively, however, though the accuracy score was undisclosed, better performance can be attained, with regards to aforementioned scores

in FPR and FNR it is of no doubt that performance can be improved upon.

Pan et al. [25] proposed a robust software modelling tool with the capability of detecting threats from SQL injection attacks and cross-site scripting, the following machine learning models were used in the experimental analysis for SQL injection attacks, Naïve Bayes, Random Forest, and SVM, the study claimed to have achieved an efficient and accurate method of detecting and checkmating threats from SQL injection and cross-site scripting attacks with the following performance scores; 0.941, 1.00 and 0.933, for precision score and 0.800 each for recall, while F-score of 0.865 and 0.889 respectively for Naïve Bayes, Random Forest and SVM, nevertheless, a considerable significant sample dataset can aid in the effective detection of SQL attack injection as the sample dataset used for this study is slightly above 200 sample point. Krishnan et al. [26] performed research on SQL injection detection based machine learning, the effectiveness of some selected machine learning algorithms was evaluated in the quest to establish the most robust learning model, the selected models that were explored are Naïve Bayes, Logistic Regression, CNN, SVM and Passive-Aggressive algorithms, with CNN proving the most optimal model for SQLi attack detection after outperforming other models compared against with 97%, 0.92 and 0.96 for the performance matric accuracy, precision and recall respectively, however, the study further proposed research of other machine learning model and its optimisation for enhancing performance.

Farooq [27], an ensemble machine learning model was employed to detect SQLi attack, Gradient Boosting Machine, Adaptive Boosting, Extended Gradient Boosting Machine and Light Gradient Boosting Machine learning algorithms were analysed, the optimal result was obtained from Light Gradient Boosting Machine with the following performance scores; 0.9934 each for accuracy, precision, recall, and F1 score respectively, and 0.0093, 0.0146, 0.1208 and 0.007 for MAE, MSE, RMSE and FPR respectively, however, with the tuning of model the performance rate for SQLi attack can be improved. In order to address the challenge associated with machine learning-based detection for SQLi attack, a novel system termed semantic query-featured ensemble learning model for SQLi attack was proposed in [28]. It was claimed that the proposed model achieved the following optimal performance for accuracy, F-score, and AUC of 98%, 0.989, 0.999 respectively, however, with an optimised model a more enhanced performance can be achieved when compared to contemporary techniques of model optimisation and dataset enhancement. A Random Forest-based SQLi attack detection approach was introduced in [29] to address the challenge of detection efficiency for SQLi attack detection, the performance score achieved in the experiment was 97% and 95% for precision and accuracy respectively, however, contemporary research has it that more effective means of detection of malware attack exist that can offer a robust performance result.

Despites the giant strides made in the detection and prevention of SQLiAs, the following limitations still exist. The existing schemes;

1. Weak characterisation of the input vector as witnessed by most machine learning based methods, leading to low accuracy and unsatisfactory recall rate.
2. Weak representation of attributes as witnessed by text vectorisation-based algorithms which leads to inaccurate description of keyword weight and consequently low prediction accuracy.
3. The use of SVM by some of the existing schemes reduces accuracy as they inherit the weaknesses of SVM which lacks the ability to perform well when the data set has more noise and the number of features for each data point exceeds the number of training data sample.

3. Inverse Document Frequency (IDF), Term Frequency-Inverse Document Frequency (TF-IDF) and Logic Regression Models

This section discusses three existing approaches in the proposed technique. These approaches were later improved as shown in section three (3) to achieve better results when compared to the benchmarked scheme.

3.1. Inverse Document Frequency (IDF)

The Inverse Document Frequency (IDF), is the logarithm of the number of the queries in the corpus divided by the number of queries where a specific term appears [30]. Inverse Document Frequency (IDF), measures how important a term is. The IDF attempts to weigh down the frequent terms and scale up the rare ones. It is given as:

$$IDF(t) = log\,(\frac{total\ number\ of\ documents}{number\ of\ \ documents\ with\ term\ t\ in\ it}), \quad (1)$$

3.2. Term Frequency-Inverse Document Frequency (TF-IDF)

The TF-IDF is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. It is often used as a central tool in scoring and ranking a document's relevance given a user query. TF-IDF can be successfully used for stop-words filtering in-text summarisation and classification [31]. The TF-IDF weight comprises of two elements: the first element is the normalised Term Frequency (TF), also referred to as the number of times a word appears in a query, divided by the total number of words in that query; the second term is the Inverse Document Frequency (IDF), which is the logarithm of the number of the queries in the corpus divided by the number of queries where the specific term appears [30]. Term Frequency, measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length also known as the total number of terms in the document as a way of normalisation:

$$TF(t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document}, \quad (2)$$

While computing TF, all terms are considered equally important but since certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus, the commonly used formular for the TF-IDF is shown in Equation 3 [32]:

$$W(t,d) \; = \; TF(t,d) \; log\left(\frac{N}{N_t}\right), \tag{3}$$

where, $TF(t,d)$ is the term frequency of the keyword t appear in the text d,N represents the number of full textsN_t represents the number of texts which have the word t.

3.3. Logistic Regression Model

Logistic Regression (LR) model is used to solve the classification problems. It is an extension of Linear Regression [33-35], where the dependent variable is categorical based on the concept of probability. In logistic regression, dependent variable is a binary variable that contains data represent as 1 (yes, success, spam) or 0 (no, failure, not-spam). The binary logistic regression maps the regression lines onto the interval [0,1], which is compatible with the logical range of probabilities. The core function of the model is logistic function or sigmoid function that can hold any real value and map it between 0 and 1. A simple logit model is shown in Equation 4:

$$In\left(\frac{\pi}{1-\pi}\right) = log(odds) = logit = \alpha + \beta x \tag{4}$$

$$\pi = Probability(Y = outcome\ of\ interest|X = x) = \frac{e^{\alpha+\beta x}}{1+e^{\alpha+\beta x}}, \tag{5}$$

where, π is the probability of the outcomes of interest, or the event, under variable Y, α is the Y intercept, and β is the slope parameter and within the inferential framework, the null hypothesis states that β is equal to zero in the population. As a result, by rejective such a null hypothesis signifies that a relation exists between X and Y. X can be categorical or continnous whereas Y is always categorical. By taking the antilog of Equation 4 on both sides, an equation to predict the probability of the occurrence of the outcome of interest can be derived as shown in Equation 5.

4. Experimental Model with Proposed Testbed and Simulation

This section discusses the experimental model with the proposed testbeds as well as the simulation environment used in the implementation.

4.1. Proposed ITFIDF-LR Model

The proposed Improved Term Frequency-Inverse Document Frequency with Optimised Logistic Regression (ITFIDF-LR employ both ITFIDF and the optimised LR model for vectorisation. In the default TFIDF equation as shown in Equation 3, $\sqrt[n]{TF}$ is substituted for TF, to eliminate excessive influence of term frequency and reflect the balance of weight [36]. Similarly, the word class weight coefficient P_t and that of the position weight coefficient b_t are incorporated into the traditional TF-IDF formula, leading to the formation of an ITF-IDF shown in Equation 6:

Table 1. Sample dataset

Sample (Sentence)	Percentage (%)
SQLi attack query	62
Normal SQL query	38

$$W(t,d) \; = \; \sqrt[n]{TF(t,d)} \times \; log\left(\frac{N}{N_t}\right) \; \times \; P_t \; \times \; b_t \tag{6}$$

An optimised logistic regression model is then put forward to maximise several predictors as well the likelihood of obtaining data given its parameter estimates. The optimised LR model is shown in Equation 7:

$$In\left(\frac{\pi}{1-\pi}\right) = \alpha + \beta x1 + \beta 2x2 +, \ldots, +\beta kxk \tag{7}$$

$$\pi = Probability\,(Y = outcomeofinterest \mid X = x1, X) = \frac{e^{\alpha+\beta 1x1+\beta 2x2,+...,+\ \beta kxk}}{1+e^{\alpha+\beta 1x1+\beta 2x2+,...,+\ \beta kxk}} \tag{8}$$

4.2. Datasets

This research make use of SQLi attack dataset [37] from a popular dataset repository known as Kaggle. The instances of the SQLi attack dataset contain 34,084 instances. The sample dataset is shown in Table 1.

Two features from the historic dataset was chosen; namely; sentence and class. The sentence needs to be detected as either normal or SQLi attack query while the second feature which is the class stands for a numeric value to determine whether it is a normal sentence or SQLi query. In our experiment, the value 1 has been used to represent the sentence as a SQLi query and 0 has been used to represent the sentence as normal statement.

4.3. Trained Model

In the detection method, the proposed SQLIAs is based on ITFIDF-LR, comprising of data pre-processing phase and optimised logistic regression model training and detecting phase. The pre-processing phase involves first getting the dataset. Then each data in the dataset is marked. The statement is marked as -1 when it is an attack sample and is marked as +1 when it is not. Then, a word segmentation tool (happierfuntokenising) is used to segment SQL statements in the dataset. In addition, in order to reduce the difference of different features during the experiment, normalisation is carried out to bring all values to a limited range. That is, the Min-Max method is used to normalise the data using Equation 9:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{9}$$

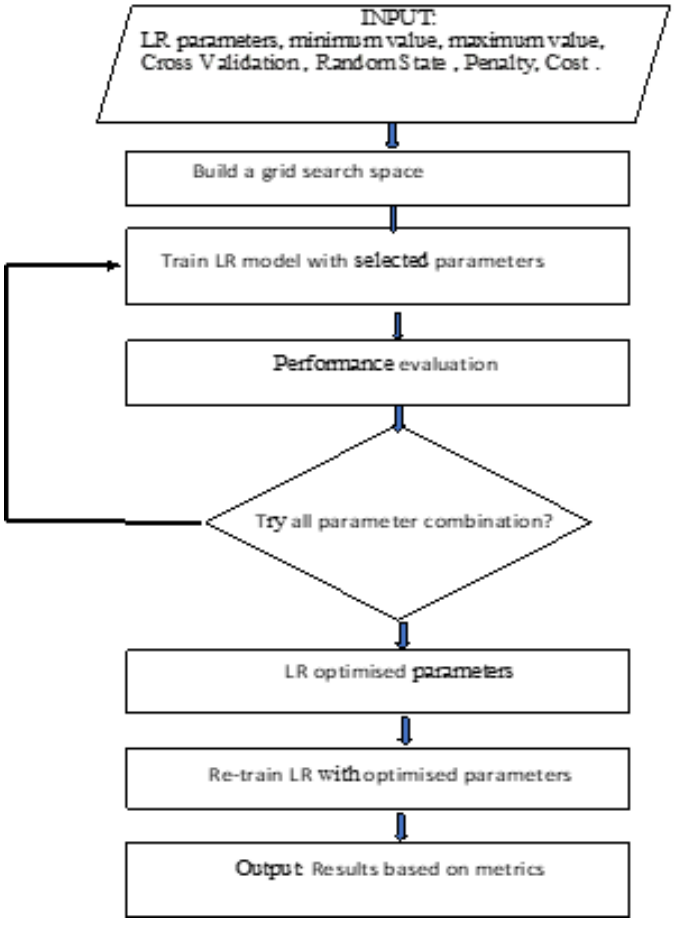

Figure 1. Flowchart of the proposed algorithm

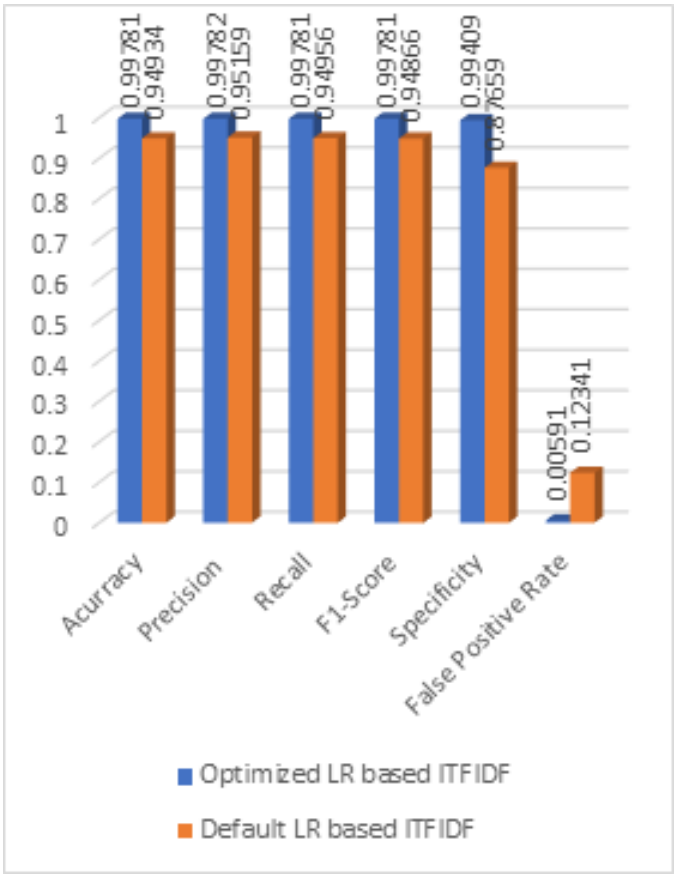

Figure 2. Comparative Analysis of Optimised and Default LR based ITFIDF

where, X denotes the current sample data value, X_{min} denotes the minimum of the sample data, X_{max} denotes the maximum of the sample data, and X norm denotes the normalised value. The feature dataset is then grouped and divided into training set and testing set in the model training and detecting phase with a composition of 70:30, that is 70% for training and 30% for testing the model. Grid Search optimisation algorithm was deployed to enhance the parameters of Logistic Regression model. The training set is used to train the optimised logic regression classifier model after which the testing set is used to verify the generated model and results of the classification generated are then evaluated. The flowchart for the proposed algorithm is shown in Figure 1.

4.4. Simulation Environment

The detail experimental hardware environments used are shown in Table 2. Likewise, Grid Search optimisation algorithm was deployed to enhance the parameters of Logistic Regression model after a complex computational operation was performed on the following defined ranges; the Cost (C), Random_State, Penalty and Solver parameters, Optimised Logistic Regression Parameter, and default Logistic Regression Parameter as shown in Table 3.

5. Performance Evaluation Metrics

After the data pre-processing and the implementing stages, the proposed approach is evaluated against the benchmarked scheme based on the metrics shown in Table 4. That is, Accuracy, Precision, Sensitivity, Specificity, F1 and FPR as used in [38, 39].

5.1. Results and Discussion

This section presents an analysis of optimised LR based ITFIDF and the default LR based ITFIDF, Table 5 while Figure 2 show graphical presentation as achieved based on the analysis performed in this research, a clear cut view gives insight of the enhanced performance and the capability strength of an optimised model compared to using a default parameter of a model, in the developed model for detection and prevention of SQLiA, there is a distinct outperformance record achieved across the board for performance metric evaluation analysis, such as outlined in succeeding subsections.

The accuracy score variation between the optimised and default LR based ITFIDF is 0.04847 which is quite significant in this analysis, 0.99781 was achieved through optimised LR based ITFIDF against 0.94934 achieved through the default LR based ITFIDF, this indicates the superiority of the optimised LR based ITFIDF model against the training of the model with default LR based ITFIDF model. The developed research

Table 2. Properties of the experimental environment

Items	Properties
Window	Windows 10 operating system
Processor	Intel (R) Core (TM) i7-7300HQ CPU@2.5gHz
Software	Visual Studio 2016,
Programming Language	Python

Table 3. Defined LR Parameter for Optimisation

	Optimised LR Parameter	Default LR Parameter
Cross Validation (CV)	5	5
Cost(C)	100.0	100.0
Random_State	7	7
Penalty	'L1'	'L2'
Solver	'liblinear'	'libfgs'

Table 4. Performance evaluation metrics

Metric	Formula
Accuracy	$=\frac{\text{True positves +true negs}}{\text{total number of examples}}$
Precision	$=\frac{\text{True positves}}{\text{True positves +true negatives}}$
Sensitivity / Recall	$=\frac{\text{True positves}}{\text{True positves + false negatives}}$
Specificity	$=\frac{\text{True negatives}}{\text{True negatives+true positves}}$
False Positive Rate (FPR)	$=\frac{\text{FP}}{\text{(TN+FP)}}$
F1 Score	$=\frac{2 * \text{TP}}{2 * \text{TP + FN + FP}}$

model for detection and prevention of SQLiA analysis evaluation of precision score reveals a definitive outperformance score with an excess score of 0.04623, the following precision score was achieved; 0.99782 and 0.951596 respectively for optimised and default LR based ITFIDF model. Recall rate of 0.99781 and 0.94956 was recorded in this research for optimised and default LR based ITFIDF developed model, the result proves the enhancement performance rating of the optimised model with 0.04825 superiority. The following scores was achieved from the result analysis of the developed model for SQLiA, 0.99781 and 0.94866 for the optimised and default LR based ITFIDF F-score performance, this is an indication outperformance of 0.04915 greater than and against default LR based ITFIDF model when compared to the optimised LR based ITFIDF model. A record of 0.99409 and 0.87659 was achieved for sensitivity score, this is an outperformance experienced with about 0.100 depicting the optimality of the tuned parameters of LR based ITFIDF in this research. The result obtained from this research based on the optimised LR based ITFIDF proves the promising feature expectant of a tuned model, the optimised LR based ITFIDF FPR achieved 0.00591 against the default LR based ITFIDF with a score of 0.12341, this presents an obvious width margin that depicts the efficiency of an enhanced LR based ITFIDF based on the excellent result achieved.

The analysis of the developed detection and prevention model for SQLiA against few other approaches in the literatures is presented to establish strength of the developed model in the detection and prevention of SQLiAs in terms of optimal performance, to this end the performance of the developed model was compared against the existing SQLiA detection techniques based on relevant performance evaluation metrics found in literatures, metrics such as accuracy, precision, recall, F1-score, specificity and FPR. Table 6 present scores of the performance metrics achieved in this research alongside that of baseline literatures, the performance record from this research outperformed other techniques in respect to accuracy, precision, recall, F1-score, specificity and FPR.

In addition, deducing from Table 5, accuracy record score achieved is the must employed performance evaluation metric in the field of SQLiA detection involving machine learning based models, this is based on the fact that it is most common in the baseline reviewed literatures. The developed model for detection and prevention of SQLiA achieved an optimal accuracy of 0,99781, which is followed by the technique employed by [27], [28] and [29] with a distinct wide margin in accuracy of 0.9934, 0.98 and 0.95 respectively while [29] have the least accuracy record of 0.95. The significant performance recorded by this research reflects how correctly the developed model can detect and prevent SQLiAs in DBMS environment, a low FPR of 0.00591 was recorded in the developed model, although baseline literature reviewed in this research did not capture FPR, FPR is an important evaluation metric in detection of attacks in computing environment as it is recorded for literatures earlier than 2021 articles reviewed.

The precision score of 0.99 was achieved for the developed model in this research and the research by [27], however, the precision that determine the exactness of the model developed in this research for the detection and prevention of SQLiA in DBMS environment outperformed that of [27] as well as [28] with a score of 0.99782 superseding 0.993 and 0.97 for [27] and [29] respectively. The recall performance score recorded in this research which entails the measure of the completeness of the performance of detection of SQLiA achieved 0.99781, against that of [27], F1-score of 0.99781 with a significant difference against [23] as well as [27], though, [27] score 0.9934 that is slightly above 0.99, [29] had the worst performance score of 0.989, the robustness of the developed detection and prevent model for SQLiA in DBMS environment have established its

Table 5. Results of the optimised LR based ITFIDF and the default LR based ITFIDF

Metrics / Approach	Accuracy	Precision	Recall	F-Score	Specificity	False Positive Rate
Default LR Based ITFIDF	0.94934	0.95159	0.94956	0.94866	0.9487	0.1234
Optimised LR Based ITFIDF	0.99781	0.99782	0.99781	0.99409	099409	0.00591

Table 6. Comparative Analysis of Optimised LR based ITFIDF with Baseline Literatures

Reference	Approach	Accuracy	Precision	Recall	F1-score	Specificity	FPR
Developed Detection and Prevention Model	Optimised LR based ITFIDF	0.99781	0.99782	0.99781	0.99781	0.99409	0.00591
[28]	Ensemble Learning Model	0.98	-	-	0.989	-	-
[29]	Random Forest Model	0.95	0.97	-	-	-	-
[27]	Ensemble Machine Learning Model	0.9934	0.9934	0.9934	0.9934	-	-

N.B: (-) means the metric value is not reported in the existing approach.

optimality capability across all performance metrics relevant in the field of this research area.

Though specificity and FPR was not recorded for the baseline journal model, this research used the performance metrics based on the fact that they are being employed for analysis engaging detection-based machine learning models, the developed detection and prevention model achieved the scores of 0.99409 and 0.00591 for specificity and FPR respectively, showing the efficient capability in detection of SQLiA in DBMS environment

6. Conclusion

The new generation of security threats have been promoted by real-time applications, where several users develop new ways to communicate on the internet via web applications. The internet is witnessing growth with several threats on a daily basis to the web application. SQLiA is one of a kind of these threat, since it serves as a medium for other several severe attacks. SQLiA, which is currently tagged as one of the most notorious means of attacking database of a system continue to haunt the securities of websites from multiple angles. This can only be checkmate through consistently deploying an evolving defense technique. Hence, in this paper, we proposed an ITFIDF-LR model to serve as a potential solution. The developed model for detection and prevention of SQLiA employed ITFIDF with an optimised Logistic Regression model to addressed threats that springs up from SQLiA in web application. The optimal performance as revealed from the analysis performed in this paper shows performance record of 0.99781 for accuracy, recall and F1-score respectively as well as 0.99782, 0.99409 and 0.00591 for precision, specificity and FPR respectively as compared to the benchmarked approaches. Future research will focus on hybridisation of learning models for further improved performance alongside other vectorising techniques.

Acknowledgment

The authors will like to appreciate the handling editor and the reviewers for their valuable comments that improved the quality of this paper.

References

[1] Z. Chen & M. Guo, "Research on SQL injection detection technology based on SVM", International Conference on Smart Materials, Intelligent Manufacturing and Automation (2018) 1.

[2] S. O. Uwagbole, W. J. Buchanan & L. Fan, "Applied machine learning predictive analytics to SQL injection attack detection and prevention", IFIP/IEEE Symposium on Integrated Network and Service Management (IM*)* (2017) 1087.

[3] R. Chandrashekhar, M. Mardithaya, S. Thilagam & D. Saha, "SQL injection attack mechanisms and prevention techniques", International Conference on Advanced Computing, Networking and Security (2011) 524.

[4] A. Dasgupta, V. Narasayya & M. Syamala, "A static analysis framework for database applications", IEEE 25th International Conference on Data Engineering (2009) 1403.

[5] C. S. Kumar, J. Seetha, S. R. Vinotha, "Security implications of distributed database management system models", International Journal of Soft Computing and Software Engineering **2** (2012) 20.

[6] S. O. Uwagbole, W. J. Buchanan & L. Fan, "Applied machine learning predictive analytics to SQL injection attack detection and prevention", IFIP/IEEE Symposium on Integrated Network and Service Management (IM) (2017) 1087.

[7] C. Anley. "Advanced SQL injection in SQL server applications,"https://crypto.stanford.edu/cs155old/cs155-spring09/papers/sql_injection.pdf. Accessed 14 December, 2021.

[8] J. Abirami, R. Devakunchari & C. Valliyammai, "A top web security vulnerability SQL injection attack—survey", Seventh International Conference on Advanced Computing. (2015) 1.

[9] D. Gabi, N. M. Dankolo & D. Muhammed, "Towards the use of new forensic approach as a panacea in investigation of cybercrime", International Journal of Scientific & Engineering Research **5** (2014) 942.

[10] B. Yusuf, R. M. Dima & S. K. Aina, "Optimized breast cancer classification using feature selection and outliers detection", J. Nig. Soc. Phys. Sci **3** (2021) 298.

[11] R. O. Oveh, O. Efevberha-Ogodo & F. A. Egbokhare, "Software process ontology: a case study of software organisations software process sub domains", J. Nig. Soc. Phys.Sci. **1** (2019) 122.

[12] O. E. Ojo, M. K. Kareem, O. Samuel & C. O. Ugwunna, "An internet-of-things based real-time monitoring system for smart classroom", J. Nig. Soc. Phys. Sci **4** (2022) 297.

[13] D. GABI, "Surveillance on security issues in cloud computing: a view on forensic perspective", International Journal of Scientific & Engineering Research **5** (2014) 1246.

[14] K. C. Rajeswari, " SQL injection attack prevention using 448 blowfish encryption standard", International Journal of Computer Science Trends and Technology (IJCST) **4** (2016) 325.

[15] M. Qbea'h, M. Alshraideh & K.E Sabri. " Detecting and preventing SQL injection attacks: a formal approach", Cybersecurity and Cyberforensics Conference (CCC) (2016) 123.

[16] L. Xiao, S. Matsumoto, T. Ishikawa & K. Sakurai, "SQL injection attack detection method using expectation criterion", 2016 Fourth International Symposium on Computing and Networking (CANDAR) (2016) 649.

[17] B. Aziz, M. Bader & C. Hippolyte, "Search-based sql injection attacks testing using genetic programming", European Conference on Genetic Programming (2016) 183.

[18] Q. Temeiza, M. Temeiza & J. Itmazi, "A novel method for preventing SQL injection using SHA-1 algorithm and syntax-awareness", Joint International Conference on Information and Communication Technologies for Education and Training and International Conference on Computing in Arabic (2017) 1.

[19] M. Sood, & S. Singh, "SQL injection prevention technique using encryption", International Journal of Advanced Computational Engineering and Networking **5** (2017) 4.

[20] L. Bossi, E. Bertino & S. R. Hussain, "A system for profiling and monitoring database access patterns by application programs for anomaly detection", IEEE Transactions on software engineering (2017) 415.

[21] S. N. Raj & E. Sherly, "SQL injection attack prevention by direct reverse resemblance technique", International Journal of Pure and Applied Mathematics **118** (2018) 599.

[22] Y. Li & B. Zhang, "Detection of SQL injection attacks based on improved TFIDF algorithm", Journal of Physics: Conference Series **1395** (2019) 012013.

[23] M. M. Hassan, R. B. Ahmad & T. Ghosh. "SQL injection vulnerability detection using deep learning: a feature-based approach", Indonesian Journal of Electrical Engineering and Informatics (IJEEI) **9** (2021) 702.

[24] L. Yu, S. Luo & L. Pan, "Detecting SQL injection attacks based on text analysis", 3rd International Conference on Computer Engineering, Information Science and Application Technology (ICCIA 2019) (2019) 95.

[25] Y. Pan, F. Sun, Z. Teng, J. White, D. C. Schmidt, J Staples & L. Krause, "Detecting web attacks with end-to-end deep learning", Journal of Internet Services and Applications **10** (2019) 1.

[26] S. A. Krishnan, A. N. Sabu, P. P. Sajan & A.L Sreedeep, "SQL injection detection using machine learning", Revista Geintec-Gestao Inovacao E Tecnologias **11** (2021) 300.

[27] U. Farooq, "Ensemble machine learning approaches for detection of SQL injection attack", Tehnički glasnik **15** (2021) 112.

[28] M. Gowtham & H. B. Pramod, "Semantic query-featured ensemble learning model for SQL-injection attack detection in IoT-ecosystems", IEEE Transactions on Reliability (2021) 1.

[29] P. Aggarwal, A. Kumar, K. Michael, J. Nemade & S. Sharma, "Random decision forest approach for mitigating SQL injection attacks", IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT) (2021) 1.

[30] H. C. Wu, R. W. P. Luk, K. F. Wong & K. L. Kwok, "Interpreting tf-idf term weights as making relevance decisions", ACM Transactions on Information Systems (TOIS) **26** (2008) 1.

[31] V. N. Gudivada, *Computational analysis and understanding of natural languages: principles, methods and applications (1st edition)*, Elsevier (2018).

[32] A. C. Finkelstein, G. Kappel & W. Retschitzegger, "Ubiquitous web application development-a framework for understanding", 6^{th} World Multiconference on Systemics, Cybernetics and Informatics (2002) 1.

[33] J. Y.-C. Peng, L.K. Lee & M. G. Ingersoll. "An introduction to logistic regression analysis and reporting", Journal of Educational Research **91** (2002) 3.

[34] G. A. Seber & A. J. Lee, *Linear regression analysis* (Vol. 329), John Wiley & Sons (2012).

[35] D. W. Hosmer Jr, S. Lemeshow & R.X, Sturdivant, *Applied logistic regression*, John Wiley & Sons (2013).

[36] W. Wang & Y. Tang, "Improvement and application of TF-IDF algorithm in text orientation analysis", Proceedings of the International Conference on Advanced Material Science and Environmental Engineering (2016) 230.

[37] S. Syed & H. Hussain, "SQL injection dataset," https://www.kaggle.com/syedsaqlainhussain/sql-injection-dataset. Accessed 10 December 2021.

[38] S. Abaimov & G. Bianchi,llilk "CODDLE: Code-injection detection with deep learning", IEEE Access**7** (2019) 128617.

[39] L. Wahab & H. Jiang. "A comparative study on machine learning based algorithms for prediction of motorcycle crash severity," PLoS one **14** (2019) 1.