MJoSHT
Malaysian Journal of Science, Health & Technology
[mjosht.usim.edu.my]

USIM

Article

SQL Injection Detection using Machine Learning: A Review

Mohammed A M Oudah and Mohd Fadzli Marhusin

Cyber Security and Systems (CSS) Research Unit, Faculty of Sciences and Technology, Universiti Sains Islam Malaysia, 71800 Nilai, Negeri Sembilan, Malaysia.

Correspondence should be addressed to:
Mohd Fadzli Marhusin; fadzli@usim.edu.my

Article Info
Article history:
Received: 5 July 2023
Accepted: 15 February 2024
Published:5 April 2024
Academic Editor:
Shahrina Ismail

Malaysian Journal of Science, Health & Technology

MJoSHT2024, Volume 10, Issue No. 1
eISSN: 2601-0003

https://doi.org/10.33102/mjosht.v10i1.368

***Abstract*— SQL injection attacks are critical security vulnerability exploitation in web applications, posing risks to data, if successfully executed, allowing attackers to gain unauthorised access to sensitive data. Due to the absence of a standardised structure, traditional signature-based detection methods face challenges in effectively detecting SQL injection attacks. To overcome this challenge, machine learning (ML) algorithms have emerged as a promising approach for detecting SQL injection attacks. This paper presents a comprehensive literature review on the utilisation of ML techniques for SQL injection detection. The review covers various aspects, including dataset collection, feature extraction, training, and testing, with different ML algorithms. The studies included in the review demonstrate high levels of accuracy in detecting attacks and reducing false positives.**

Keywords— Cybersecurity; Machine Learning; SQL Injection Detection

I. INTRODUCTION

SQL injection (SQLI) attacks are a common type of web application security vulnerability that can allow attackers to execute malicious SQL statements and gain unauthorised access to sensitive data. The increasing complexity of web applications and the need for robust and effective security measures make the use of robust techniques for detecting SQLI attacks. According to OWASP, SQLI is one of the top 10 most critical web application security risks. As shown in Fig.1, SQLI was recognised as the first most critical risk in their 2017 Top 10 report, with an estimated frequency of roughly 25%. In 2021, it drops to third place, with 94% of applications examined for some sort of injection, with a maximum incidence rate of 19% and an average incidence rate of 3.37 [1].

The lack of a standard structure for SQLI attacks makes it difficult to develop traditional signature-based detection methods that rely on predefined patterns or rules [26]. Machine learning (ML) algorithms can overcome this challenge by learning from patterns and relationships in the data, allowing them to detect previously unknown SQLI attacks and adapt to changes in attack techniques over time.

In this paper, our primary objective is to conduct an extensive literature review, systematically comparing various recent research studies that utilise ML algorithms to prevent and detect SQLI attacks. We aim to critically evaluate the feasibility and effectiveness of these algorithms as documented in existing research rather than conducting new experiments. Our goal is to synthesise this information to provide a detailed

assessment and classification of different ML models based on their performance in real-time SQLI attack detection and prevention. This analysis will serve as a roadmap, highlighting key findings and insights derived from previous research to guide future studies and developments in this field.

This paper will provide valuable insights into the effectiveness of using ML to detect SQLI attacks and contribute to the field of web application security. This paper will also compare the effectiveness and limitations of using different ML models for SQLI detection and suggest directions for future work in this area.

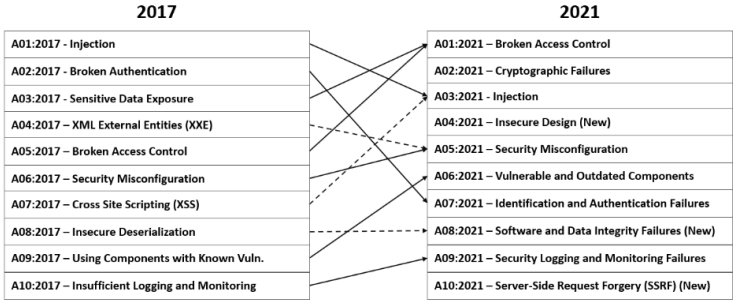

Fig. 1 SQLIA OWASP ranking in 2017 and 2021 [1]

II. SQL Injection Overview

SQLI is a type of security vulnerability that arises when user input is not appropriately validated or sanitised before being incorporated into dynamic SQL statements. This can occur in any code that accepts user input to construct dynamic SQL statements [2]. The primary goal of an SQLI attack is to gain unauthorised access to a database or manipulate its data in unintended ways.

A. How SQL Injection works

Developers can accidentally introduce SQLI vulnerabilities into their applications when they use unfiltered user input in their SQL statements [3]. This can allow an attacker to inject malicious SQL code into the database and execute it, potentially compromising sensitive information or taking over the database.

By understanding the different layers in a web application and how SQLI attacks occur, developers can take steps to prevent these attacks and protect their web applications from being compromised. Fig. 22 shows the layers of web applications. In the presentation layer, the users interact with the web application and send requests to the logic layer. The logic layer contains the programming code that processes the user input and generates dynamic SQL statements that are sent to the storage layer for execution. Suppose the logic layer does not properly validate or sanitise the user input. In that case, an attacker can inject malicious SQL code into the dynamic SQL statements, potentially compromising the database or sensitive information stored in the database [4].

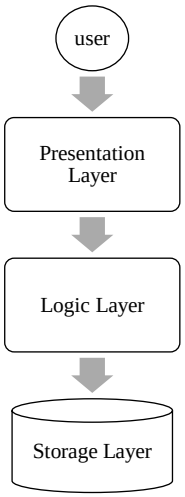

Fig. 2 Web application layers

B. SQL Injection Causes

. There are several common causes of SQLI attacks [5]:

- Inadequate input validation: Failure to validate user input properly, allowing attackers to inject malicious code into a query.
- Unparameterised queries: Unparameterised queries are used where user input is inserted directly into an SQL query, allowing attackers to inject malicious code into the query.
- Degree of granted user privileges more than required: Granting excessive privileges to users or roles, allowing them to perform actions that are not necessary for their job functions, increasing the risk of SQLI attacks.

C. SQL Injection Classifications

SQLIA is classified into two main types, classical and advanced SQLIA:

1) Classical SQL injection attacks: There are several types of classical SQLI attacks, as listed in Table 1, such as tautology attacks, union queries, blind SQLI, piggy-backed queries, illegal or logically incorrect queries, and alternate encoding. Each of these attack types involves different methods of exploiting the application's input validation process and can lead to severe consequences.

2) Advanced SQL injection attacks refer to modern techniques used to bypass traditional detection and prevention methods [6]. These attacks can be difficult to detect and prevent, as they often involve more complex and sophisticated techniques than classical attacks. Some examples include Fast Flux and Compounded SQLI Attacks.

TABLE I . CLASSIC SQL INJECTION ATTACK EXAMPLES

SQLIA Type	SQL injection attack example	Cause
Tautology attacks	SELECT * FROM users WHERE username = '' OR '1'='1' AND password = '<password>'	The injected condition (OR '1'='1') is always true, which overrides other conditions in the SQL query, bypassing security checks.
Union query	SELECT name, email FROM users WHERE id = ' UNION SELECT cc_number, cc_expiry FROM credit_cards WHERE user_id = 1 --'	Appends an additional query (UNION …) to the original query. Retrieve data from a different table (credit_cards) that was not intended to be accessed in the original query.
Blind SQL injection	SELECT * FROM products WHERE name LIKE '% or 1=1-- %';	Inserts a condition (or 1=1--) that is always true into the WHERE clause. This alters the query's logic, forcing it to return all entries in the table. The -- comments out the rest of the SQL statement, ensuring that only the injected part is executed.
Piggy-backed query	SELECT * FROM users WHERE username = '' OR 1=1; DROP TABLE users --' AND password = '<password>'	Adds an additional malicious query (DROP TABLE users) after the legitimate one, using a semicolon to separate them. The injected OR 1=1 ensures the first query always returns true, and the -- comments out the rest of the legitimate query.
Illegal/logically incorrect queries	' OR 1=1; SELECT credit_card_number FROM users WHERE username = '<username>' AND password = '<password>'	Injects an always-true condition (' OR 1=1;) to bypass authentication checks, followed by a separate query to extract sensitive data (SELECT credit_card_number FROM users). The semicolon separates the injected query from the original, allowing the execution of an additional, unauthorised query that compromises data security.
Alternate encoding	' OR 1=1; SELECT * FROM users WHERE username = 0x61646D696E AND password = 0x61646D696E --	Uses hexadecimal encoding to disguise the input, bypassing filters that may only recognise standard alphanumeric input. This type of attack can bypass security measures that are not designed to decode and inspect alternate encodings.

III. MACHINE LEARNING

ML is one of the disciplines of artificial intelligence that enables machines to employ intelligent software to learn how to do tasks, giving machines the capacity to learn and decide on new tasks. It aims to develop algorithms and models that enable computers to learn and improve from experience without being explicitly programmed. This is accomplished by training ML models on large datasets and using statistical and mathematical techniques to identify patterns and relationships within the data. Once trained, these models can be used to make predictions and decisions based on new data without the need for explicit programming for each specific case.

A. Machine Learning Classifications

ML is generally classified into four types: *supervised* and *unsupervised learning*, *semi-supervised learning*, and *reinforcement learning*.

- Unsupervised learning uses an unlabelled dataset to identify patterns and anomalies in data. This method has two categories: clustering, which groups data based on patterns, and association, which uncovers relationships between data sets. Association identifies a relationship between two datasets; if A is discovered, then B must be true.
- Supervised learning is a type of ML that uses a labelled training dataset to learn the relationship between data and the label. After that, it evaluates new data, called a test dataset, to determine the accuracy of supervised learning. The two categories of supervised learning are classification, which predicts categorical values such as true/false, colour, or type of disease, and regression, which predicts numerical values such as pressure, price, financial stocks, and other variables.
- Semi-supervised learning involves using both labelled and unlabelled data in the training process. The goal of semi-supervised learning is to build a classification model using the unlabelled data first, then use the labelled data for training to improve the model's accuracy. It combines the benefits of supervised and unsupervised learning methods and can be useful when obtaining labelled data is difficult or expensive.
- Reinforcement learning involves using data obtained through interactions with the environment to create intelligent agents [7]. It includes model types such as Temporal difference learning and Q learning.

B. Machine Learning Applications

ML algorithms are employed for a variety of tasks and are currently prevalent in the fields of big data sciences, image processing, and medical sciences [7]:

- Computer-Aided Diagnosis: ML pattern recognition techniques support a wide range of image recognition applications, including the diagnosis of various medical conditions through the interpretation of various medical images.
- Computer Vision: ML can help machines operate discretionarily according to the situation by analysing incoming photos for facial recognition, security applications, and driverless autos.

- Speech recognition uses ML to recognise spoken words and translate them into text.
- Text mining: ML may impact unstructured data or text to extract information that is valuable for a variety of applications, including social media monitoring, spam filtering, injection detection, business intelligence, and national security.
- Malware and Spam mail filtering: ML algorithms can learn patterns in large datasets of known malware or spam and use this knowledge to identify and block new instances of malicious software or unwanted emails as proposed in [8].

C. Machine Learning for SQL injection detection

ML has become an increasingly popular approach for various applications, including cybersecurity. The absence of a standard structure for SQLI attacks has made it difficult to develop effective traditional methods of signature-based detection. However, ML algorithms can learn from patterns and relationships in the data to identify benign or malicious SQL queries in web requests. Different ML algorithms can be utilised for this purpose, but the quantity and quality of training and testing data remain a significant concern for most classifiers. Despite these challenges, ML has shown promise in improving the accuracy and efficiency of SQLI detection.

IV. DISCUSSION

This section aims to present a comprehensive overview of recent research on SQLI detection using ML techniques.

Demilie & Deriba [9] presented a novel framework for detecting SQLI attacks using various ML algorithms, including supervised, unsupervised, and semi-supervised approaches. Their proposed approach involved feature extraction, preprocessing, and classification using diverse ML algorithms. The authors used a dataset of over 54,000 pieces of data, with 70% used for training and 30% for testing. They collected the data from weblogs, cookies, session usage, and HTTP request files and divided it into genuine and malicious queries. The hybrid approach, which combined artificial neural networks and support vector machines, outperformed other ML approaches, achieving high accuracy, precision, recall, and F1-score. The authors also discussed the nature of SQLI attacks, prevention and detection mechanisms, and proposed solutions based on deep learning, ML, and hybrid techniques. Their proposed framework is flexible and can be applied to different protocols, including HTTP and HTTPS.

Krishnan *et al.* [10] propose a framework that uses ML algorithms to detect SQLI attacks on the client side. The authors use tokenisation as the text parsing method and evaluate the performance of five different ML algorithms, including Naive Bayes, Logistic Regression, SVM, CNN, and Passive Aggressive Classifier. The results show that CNN performs the best with an accuracy rate of 97%. In comparison, Naive Bayes has a 95% accuracy rate, Logistic Regression has a 92% accuracy rate, and SVM and Passive Aggressive Classifiers have 79% accuracy rates. The authors conclude that the proposed framework improves the detection of SQLI attacks and enhances application security. The study highlights the importance of using ML algorithms for detecting SQLI attacks and provides insights into the effectiveness of different algorithms in this context.

Hernawan *et al.* [11] proposed a system that combines two methods, SQLI Free Secure (SQL-IF) and Naïve Bayes, to prevent SQLI attacks. SQL-IF is a method that checks for the presence of special characters, keywords, and Boolean in the input data to detect SQLI attacks. The Naïve Bayes method was used for probabilistic classification of the input data. The study collected attack log data through a simulation laboratory and conducted penetration testing on vulnerable web applications. The hybrid approach using both methods in sequence produced the highest accuracy value of 90% in preventing SQLI attacks. However, it also resulted in longer load times for web pages due to increased checks and arithmetic operations. The study suggests that a small constant value and a large dataset can increase the accuracy of the system. The proposed system provides a new approach to preventing SQLI attacks and can contribute to improving the security of web applications.

In their study, Khanuja *et al*. [12] focused on the development of a system aimed at detecting SQLI and Cross-Site Scripting vulnerabilities in web applications utilising ML algorithms. Their system consists of four modules: Web Crawler, SQLI detection, Cross Site Scripting detection, and Report generation. To establish the relationship between dependent and independent variables, the researchers employed two ML algorithms, namely Logistic Regression and Naïve Bayes Classifier, to create the most suitable model. The proposed system identifies common vulnerabilities found in web applications by scanning URL query parameters, forms, and cookies on web pages, thereby generating a comprehensive report highlighting the detected vulnerabilities. By using ML algorithms, the study offers a novel approach to detecting vulnerabilities, which could help improve the overall security of web applications.

In [13] Pham *et al*. proposed a machine-learning model for detecting SQLI attacks on the client side. The model consists of five steps, including data preparation, splitting data into training and testing sets, text parsing using tokenisation, Natural Language Processing, and frequency measurement of the occurrence of words. They evaluated their model using five ML algorithms and measured the results using various metrics such as Precision, Recall, F1-score, Weighted Average, and Confusion Matrix Accuracy. The experimental results showed that three out of the five tested algorithms, including Logistic Regression, Random Forest, and Extreme Gradient Boosting, achieved 100% accuracy. However, the proposed model requires larger datasets to obtain reliable evaluation results.

In this study [14], Abdulmalik proposed a model to enhance the effectiveness of SQLI attack detection using ML techniques by combining Dynamic and Static Analysis. The model comprises three phases: Dataset, Static and Dynamic Analysis, and Model Construction. The Dataset phase involves inserting a record consisting of only symbols in every table of the database using a suggested algorithm. At the same time, a program is employed in the CGI interface to monitor all input queries. In the Static and Dynamic Analysis phase, semantic features are extracted, while in the Model Construction phase, ML algorithms such as Random Forest, Artificial Neural Network, Support Vector Machine, and Logistic Regression

are used to construct the model. Although the evaluation and validation of the proposed model are still in progress, the proposed method's performance will be evaluated based on Detection Overhead and Error Rate using a tenfold approach.

In this paper, Morufu *et al.* [15] proposed a Naive Bayes-based pattern recognition model for detecting and categorising SQLI attack types. The model was trained and evaluated using 16,050 data instances that included both vulnerable and non-vulnerable web pages. The evaluation was done using ML algorithms, and the validation was performed using 10-fold stratified cross-validation with 1-10 random seeds. The results showed a high accuracy rate of 98% for detection and 99% for categorisation. The model was compared to existing techniques and was found to perform better.

Akinsola *et al*. [16] compared the performance of various supervised learning classification algorithms for binary classification using 10-fold cross-validation and hold-out methods. The tested algorithms included Logistic Regression (LRN), Stochastic Gradient Descent (SDG), Sequential Minimal Optimisation (SMO), Bayes Network (BNK), Instance-Based Learner (IBK), Multilayer Perceptron (MLP), Naive Bayes (NBS), and J48. The study found that the optimal algorithm cannot be chosen based on a single metric such as accuracy. In both hold-out and cross-validation methods, SDG and J48 performed equally well with 100% accuracy, followed by LRN. IBK had the lowest conceivable running time to build both at Hold-Out and 10-Fold Cross-Validation, with values of 0.16 seconds and 0.06 seconds, respectively, and NBS had the next lowest total training time.

T.P. Latchoumi *et al.* [17] propose using the Support Vector Machine (SVM) algorithm to prevent SQLI attacks in web applications. The system trains the SVM algorithm with all possible malicious expressions and generates a model that predicts whether a new query contains any malicious expressions. The attack signatures are identified as SQLIA tokens and SQLIA positive symbols, while legitimate requests take the form of expected data. The proposed system includes an admin and user login page for registration, and the admin can securely upload datasets for training. The paper also describes an Android application for household services that uses XML, Java, SQL, and Firebase connectivity. The implementation of the application required overcoming challenges in database connections, API design, and synchronisation.

The paper by Kavitha *et al.* [18] proposes an unsupervised machine-learning approach to detect SQLI attacks in web applications using the K-Means clustering algorithm. The system consists of three steps: the URL intercept engine, context-free grammar for SQLI attacks, and pattern classification through ML. The first level uses predefined patterns created by context-free grammar, while the second level clusters patterns and classifies them into one of those clusters to avoid injection. The system is evaluated based on accuracy, response time, and time complexity and can detect three categories of SQLI attacks: boolean, piggy-backed, and union attacks. If an attack pattern is detected, the system prevents the query's execution.

In their paper, Oudah et al. [3] explore the effectiveness of four ML models in detecting SQLI attacks using multi-tokenisation levels. The proposed system involves data preparation, feature extraction, dataset splitting, model building, training, and testing. The study used a dataset of 37,093 records of web requests and implemented different feature extraction levels such as word level, character level, and N-gram level. The results revealed that Extreme Gradient Boosting (EGB) had the highest recall and precision using word-level feature extraction, while the SVM model achieved the best accuracy using character-level feature extraction. Naïve Bayes was the fastest model, taking six milliseconds to train the classifier at the N-gram level and seven milliseconds at the word level, followed by eight milliseconds at the character level. The study encountered challenges in using NLP for text preprocessing for SQLI detection and identifying irrelevant text in the data that needed removal.

Alkhatami & Alzahrani [19] investigate the detection of SQLI attacks using ML techniques, including K-Nearest Neighbors, Multinomial Naive Bayes, Decision Tree, and Support Vector Machine algorithms. The study focuses on detecting SQLI attacks in cloud computing platforms, where ML can offer superior detection methods compared to traditional approaches. The authors use a dataset of SQL queries and SQLI attack queries, which are preprocessed and feature-extracted before being used to train the ML model. The results demonstrate that the SVM algorithm achieves the highest accuracy of 99.42%, followed by the Decision Tree with an accuracy of 99.4%. In comparison, MNB and KNN algorithms have accuracy values of 97.09% and 92.45%, respectively. The proposed model achieves an average accuracy of more than 99% with a low error rate.

Azman *et al.* [20] propose a signature-based SQLI detection system that utilises ML techniques. The system is trained on a dataset of benign and malicious web requests extracted from access log files, and its performance is evaluated on a separate testing dataset. The architecture of the proposed system includes three main functions, namely extraction, training, and detection. The access log file is extracted and divided into a training set and a test set. The training set is used to train the detector to create a knowledge base, which is then utilised to classify the test set into benign or malicious web requests. The tool extracts the URLs and queries from the access log file and converts them into sets of signatures for training and testing. The experimental results reveal high detection accuracy for the testing dataset, with a few false positives due to an inadequate number of signatures in the training set.

In Farooq's paper [21], a model is proposed to detect SQLI attacks, which exploit database vulnerabilities. The model comprises four stages: dataset collection of SQLI attack queries, feature extraction and selection using tokenisation, training with 70% of the dataset, and testing with the remaining 30%. The study uses a manually collected labelled dataset of 35,198 queries categorised into normal SQL queries, SQLI attack queries, and plain text with 21 features. The proposed model employs ensemble learning algorithms, including Gradient Boosting Machine, Adaptive Boosting, Extended Gradient Boosting Machine, and Light Gradient Boosting Machine. The model achieves over 99% average accuracy in detecting SQLI attacks with minimal error rate, with the best accuracy being 0.993371 using a Light Gradient Boosting Machine. The proposed model is efficient in distinguishing SQLI attacks

from normal SQL queries and plain text, making it suitable for real-world detection systems.

Mishra [22] used the Naïve Bayes ML approach for detecting SQLI attacks. She explained that this approach is simple to implement, requires fewer computational resources, and can be trained even on small datasets. However, it has the drawback of failing when a new type of SQLI is encountered for the first time. To address this, Mishra employed an ensemble learning technique to combine multiple models, each with its strengths and weaknesses, to reduce bias error and variance error and improve the model's accuracy and generalisation ability. Mishra trained her model on a dataset consisting of 6000 SQLI examples, including both plain text and SQLI data. To test the model's performance, the researcher used an open-source tool called Libinjection to generate additional SQLI examples. Mishra used tokenisation to preprocess the text-based dataset and performed feature extraction using the G-Test score for all token values. The resulting model achieved higher accuracy than the Naïve Bayes classifier but required more computational resources due to the use of a gradient-boosting classifier.

Triloka *et al.,* in their paper [23], tested five algorithms, including Naïve Bayes, Logistic Regression, Gradient Boosting, K-Nearest Neighbor, and Support Vector Machine, to detect SQLI attacks. The Support Vector Machine had the highest level of accurate detection, with a 99.77% detection accuracy and 0.00100 microseconds per query time. The study used natural language processing to increase detection accuracy. The researchers carried out five stages in producing the detection method, including data preparation, preprocessing and modelling, feature engineering, data training, and testing. The SVM algorithm had the highest level of accuracy, which was 0.9977 and proved reliable in detecting SQLI attacks. The optimisation stage increased the accuracy rate of SVM from 0.89 to 0.9977 with a margin of 6.5. SVM was proven reliable in detecting SQLI attacks.

Uwagbole *et al.* [4] proposed a technique that utilises predictive analytics in big data to detect SQLI attacks in web requests. They built a dataset by extracting known attack patterns and trained a Two-Class Support Vector Machine (TCSVM) using supervised learning to classify requests. The injection point is identified after the WHERE clause in the SQL query. The technique achieved an accuracy of 0.986, precision of 0.974, recall of 0.997, and F1 Score of 0.985. The trained classifier can be deployed as a web service that is consumed by a custom .NET application implementing a web proxy API to intercept and accurately predict SQLIA in web requests, preventing malicious requests from reaching the database. The approach was tested on a web application, and empirical evaluations were presented in Confusion Matrix (CM) and Receiver Operating Curve (ROC).

In this paper [8], Matin and Rahardjo propose an architecture for detecting malware using honeypots and ML. The proposed architecture consists of network components, routers, honeypots, data analysis, and a real system. Honeypots capture traffic packets and store them on an artificial server for analysis. The data analysis module uses supervised ML algorithms, including decision trees and support vector machines (SVM), to classify the captured packets as malware or benign. The architecture also includes a retraining process to improve malware detection accuracy. The authors used the EMBER dataset [24] and conducted cross-validation tests with a k-fold validation of 10 to evaluate the classifier algorithm's performance.

Wang and Wang [25] propose a detection method for SQLI attacks based on the improved TFIDF algorithm and ML classifiers (SVM, KNN, and DT). The method analyses and compares a large number of attacks and normal SQL statements, summarising the characteristics of SQL statements and vectorising the text. The improved TFIDF algorithm is used in the data preprocessing stage, and three different classifiers are used in the classification stage. The results show that the improved TFIDF algorithm combined with SVM has a higher accuracy rate and a lower false alarm rate. The proposed method solves the problem of low detection accuracy when the number of sensitive and insensitive words is similar. The SVM algorithm has the best classification effect among the three classifiers tested.

Overall, the reviewed studies have employed various ML models with distinct data preparation and text processing techniques. Fig. 3 depicts the frequency of each machine-learning model used in these papers.

TABLE II serves as a reference guide for summarising the reviewed papers, including the techniques used, publication dates, and limitations of each approach. It can be a valuable resource for researchers and practitioners working on the detection of SQLI attacks using ML. TABLE III provides a comprehensive summary of the evaluation metrics for each reviewed paper, including the ML models used, datasets utilised, and the accuracy, precision, and recall metrics for each model's performance evaluation in detecting SQLI attacks. Together, these tables offer an overall understanding of the various techniques and their effectiveness in detecting SQLI attacks using ML.

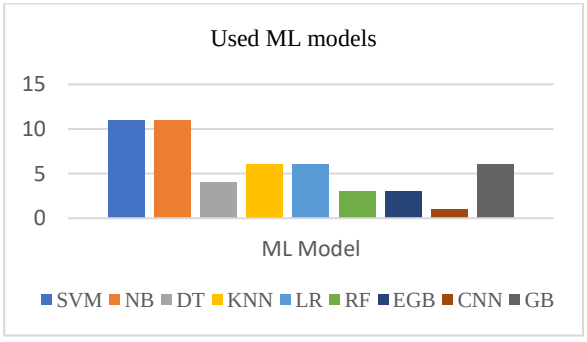

Fig. 3 Frequency of ML models used in reviewed papers

TABLE II
SUMMARY OF OTHER WORKS EVALUATION

#	Author(s)	Publish Year	Used Technique	Limitation(s)
1.	Demilie & Deriba [9]	2022	The hybrid approach combines ANN, SVM, NB, DT, and RF	The proposed system is limited to its dependence on the specific characteristics of the target system
2.	Krishnan *et al.* [10]	2021	Five machine learning algorithms: NB, LR, SVM, CNN, and PAC	found that the Naive Bayes and SVM classifiers have relatively low accuracy rates compared to the other classifiers tested, which may limit their usefulness in certain applications
3.	Hernawan et al. [11]	2020	A combination of the SQLI Free Secure (SQL-IF) method and the Naïve Bayes model	The study only evaluates the proposed approach in a limited number of vulnerable web applications, so the results may not be generalisable to all web applications. The longer average load times for web pages due to increased checks and arithmetic operations may negatively impact user experience and website performance.
4.	Khanuja et al. [12]	2021	Utilised two Machine Learning algorithms, LR and NB Classifier	The proposed system may not detect all vulnerabilities present in a web application, as vulnerabilities may exist in areas that are not referenced in collected URLs.
5.	Pham *et al*. [13]	2020	The use of five machine learning models are NB, SVM, DT, LR, and EGB	The proposed model should be tested with larger datasets to improve the reliability of the results. It also did not mention how to deal with new types of attacks and what types can be detected.
6.	Abdulmalik [14]	2021	Used four different machine learning algorithms: Random Forest (RF), ANN, SVM, and LR.	The paper did not present the results of this approach. Moreover, the models may not be capable of detecting unseen attacks with new semantic features that were not included in the training datasets.
7.	Morufu et al. [15]	2018	Naive Bayes-based classifier	The model is evaluated only on a single dataset of 16,050 instances, which may not be representative of all possible scenarios. Thus, the generalizability of the model to other datasets is uncertain.
8.	Akinsola et al. [16]	2020	Various ML models included LR, Stochastic Gradient Descent (SDG), Sequential Minimal Optimisation (SMO), Bayes Network (BNK), Instance-Based Learner (IBK), Multilayer Perceptron (MLP), Naïve Bayes (NB), and J48.	Compared to the performance of various supervised learning algorithms for SQLI detection, the study did not consider other factors, such as the size and diversity of the datasets, the complexity of the queries, or the impact of false positives and false negatives.
9.	Latchoumi et al. [17]	2020	The attack signatures are identified as SQLIA tokens and SQLIA positive symbols, while legitimate requests take the form of expected data, and SVM is used to classify user requests in real-time.	The evaluation of the SVM algorithm is limited to testing on a single dataset. The system implementation is also limited to an Android application for household services, and it is unclear how well it would perform in other contexts or applications.
10	Kavitha et al. [18]	2021	An unsupervised machine learning approach that uses the K-Means clustering algorithm to detect SQLIA	The evaluation is conducted on a limited number of SQLI attacks, and it is unclear how the proposed system would perform on a larger and more diverse dataset.
11.	Oudah *et al.* [3]	2022	Applied four ML algorithms using different feature extraction TF-IDF levels	More machine learning models should be compared with more feature extraction techniques.
12.	Alkhatami, Alzahrani [19]	2022	Focuses on detecting SQLI attacks in cloud computing platforms using K-NN, NB, DT, and SVM algorithms.	The study's scope is limited to detecting SQLI attacks in cloud computing platforms, and therefore, the results may not apply to other types of web applications.
13.	Azman et al. [20]	2021	An updated Knowledge Base using machine learning. They used the Boyer's Moore string matching algorithm to compare malicious features in log strings to detect injections.	The paper's lack of clarity on the specific ML algorithms used and the relatively small size of the tested datasets are limitations. Additionally, caution should be exercised when examining user log files as they may impact server responsiveness. An accuracy of 100% could indicate overfitting, which should be considered when interpreting the results.

#	Author(s)	Publish Year	Used Technique	Limitation(s)
14.	Farooq [21]	2021	The model uses ensemble learning algorithms, specifically Gradient Boosting Machine, Adaptive Boosting, Extended Gradient Boosting Machine, and Light Gradient Boosting.	The use of a manually collected dataset may not accurately reflect all possible scenarios. Although the proposed model exhibited high accuracy on the testing dataset, it remains uncertain whether the model is overly specialised to the particular dataset employed in the study, raising the possibility of overfitting.
15.	Mishra [22]	2019	Used ensemble ML Gradient Boosting algorithm.	Ensemble Learning takes a longer time in the learning phase but gives better learning results. Also, the GB classifier achieves better accuracy than the NB classifier, but it needs more computational resources.
16.	Triloka, Hartono [23]	2022	Compared the use of five algorithms, including NB, LR, GB, K-NN, and SVM in SQLI detection	It only evaluates the proposed approach in a limited number of vulnerable web applications, so the results may not be generalisable to all web applications.
17	Uwagbole *et al.* [4]	2017	A web proxy API system that intercepts requests and checks user requests using a trained SVM classifier	The proposed technique is tailored to a specific web application and database system, which limits its generalizability to other systems.
18	Matin & Rahardjo [8]	2019	Proposed to use ML SVM and DT models integrated with honeypot to detect malware attacks based on malware features and behaviours.	The paper just mentioned the architecture of the proposed approach and did not show experiment results.
19	Wang & Wang [25]	2022	Used honeypots and analysed them using machine learning algorithms such as Decision Tree and Support Vector Machine (SVM).	The small dataset used may not represent all possible SQLI attacks and could limit the generalizability of the proposed detection method. Additionally, the architecture of the proposed approach was not presented.

TABLE III
SUMMARY OF OTHER WORKS EVALUATION

Ref.	ML Algorithm(s)	Dataset Size	Evaluation		
			Accuracy	Precision	Recall
[9]	NB	54, 306	87.2	0.875	0.863
	DT		94.8	0.917	0.908
	SVM		97.3	0.964	0.956
	RF		93.4	0.943	0.930
	ANN		98.7	0.988	0.991
	Hybrid		99.4	0.992	0.994
[10]	NB	*	95	0.85	0.98
	LR		92	0.97	0.76
	CNN		97	0.92	0.96
	SVM		79	1.0	0.20
	Passive Aggressive		79	1.0	0.20
[11]	NB	250	90	*	*
[12]	LR	92	*	*	*
	NB				
[13]	NB	1,483	77	0.71	0.46
	LR		100	1.0	1.0
	SVM		57	0.57	0.0
	RF		100	1.0	1.0
	EGB		100	1.0	1.0
[14]	RF	*	*	*	*
	ANN		*	*	*

Ref.	ML Algorithm(s)	Dataset Size	Evaluation			
			Accuracy		Precision	Recall
	SVM		*		*	*
	LR		*		*	*
[15]	NB	16,050	98		*	*
			10-Fold Cross-Validation	Holdout-70		
[16]	LR	*	99.02	99.99	*	*
	Stochastic Gradient Descent (SDG)	*	99.99	100	*	*
	Sequential Minimal Optimisation (SMO)	*	99.98	99.98	*	*
	Bayes Network (BNK)	*	99.70	99.70	*	*
	Instance-Based Learner (IBK)	*	99.99	99.98	*	*
	Multilayer Perceptron (MLP)	*	*	*	*	*
	NB	*	99.38	99.41	*	*
	J48	*	99.99	100	*	*
[17]	SVM	*	*		*	*
[18]	K-Means clustering	*	*		*	*
[3]	NB	37,093	99.3		0.994	0.991
	LC		99.2		0.994	0.988
	SVM		99.7		0.997	0.995
	EGB		99.5		0.996	0.992
[19]	K-NN	6,184	92.45		*	*
	NB		97.09		*	*
	DT		99.4		*	*
	SVM		99.42		*	*
[20]	*	58	93		*	*
		57	100		*	*
		58	100		*	*
		57	100		*	*
		56	100		*	*
[21]	GB	35,198	99.1		0.991	0.990
	Adaptive Boosting		99.1		0.990	0.989
	EGB		99.2		0.991	0.990
	Light Gradient Boosting		99.3		0.993	0.993
[22]	NB	3,707	92.8		*	*
	GB		97.4		*	*
[23]	SVM	33,727	99.7		*	*
	K-NN		99.7		*	*
	LR		99.6		*	*
	GB		99.4		*	*
	NB		97.5		*	*
[4]	SVM	479,000	98.6		0.974	0.997

Ref.	ML Algorithm(s)	Dataset Size	Evaluation		
			Accuracy	Precision	Recall
[8]	SVM	900,000	*	*	*
	DT		*	*	*
[25]	SVM	3000	*	0.990	0.992
	DT		*	0.982	0.976
	K-NN		*	0.982	0.982

* Means not mentioned

The studies have employed different ML practices such as dataset collection, feature extraction, training, and testing with different ML algorithms.

Results show that these techniques have achieved high levels of accuracy in detecting attacks and reducing false positives. The SVM algorithm is the most effective in detecting SQLI attacks, while Naïve Bayes and Ensemble Learning techniques have also been shown to improve detection accuracy.

V. CONCLUSIONS

In conclusion, this literature review provides an overview of several studies on the use of ML for detecting SQLI attacks and malware. The studies presented here use a variety of approaches, including different algorithms, dataset sizes, and feature selection techniques to achieve high detection accuracy rates.

Upon reviewing these studies, we have identified three key findings that hold significant implications for future research. Firstly, the proposed techniques emphasise the importance of dealing with passed SQLI queries in web requests, particularly when system developers fail to handle requests before accessing the system database because certain techniques focus on identifying SQLI attack signatures or predefined patterns in user logs to prevent injection requests. So, by failing to handle these queries adequately, there is a possibility that some threats could be passed through, posing potential risks to the system's security. The second key finding is that the size and quality of the training dataset significantly impact the detection accuracy. It has been observed that a larger and higher-quality dataset leads to an improved ability to accurately identify SQLI attacks. Lastly, combining multiple ML models has demonstrated an increase in detection accuracy. By leveraging the strengths of various models, researchers have achieved higher levels of accuracy in detecting SQLI vulnerabilities.

As a way forward, it is imperative for future research to target specific facets of SQLI attack detection and prevention. This includes addressing evasion-centric attack patterns and behaviour-related anomalies, refining approaches to manage encoding-driven attacks, and augmenting dataset quality by establishing accessible public repositories. Furthermore, the exploration of hybrid machine-learning models and the development of innovative, behaviour-centric techniques for real-time detection is critical. Such concerted efforts will be pivotal in advancing the field, keeping abreast of the constantly evolving landscape of SQLI threats, and bolstering the effectiveness of cybersecurity strategies.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest regarding the publication of this paper.

ACKNOWLEDGEMENT

The authors acknowledged Fakulti Sains dan Teknologi, Universiti Sains Islam Malaysia (USIM), for the facilities provided.

REFERENCES

[1] "OWASP Top10 - 2021," 2021. [Online]. Available: https://owasp.org/Top10/. [Accessed 14 May 2023].

[2] J. Clarke, SQL Injection Attacks and Defense, vol. 2, Waltham: Elsevier, 2012.

[3] M. A. Oudah, M. F. Marhusin and A. Narzullaev, "SQL Injection Detection Using Machine Learning with Different TF-IDF Feature Extraction Approaches," in International Conference on Information Systems and Intelligent Applications, Springer, Cham, 2022, pp. 707-720. DOI: 10.1007/978-3-031-16865-9_57.

[4] S. Uwagbole, W. J. Buchanan and L. Fan, "Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention," in 3RD IEEE/IFIP Workshop on Security for Emerging Distributed Network Technologies (DISSECT), Lisbon, Portugal, 2017. DOI: 10.23919/INM.2017.7987433.

[5] M. Soni, A. Prakash, H. Mittal and M. Tiwari, "Honeypot Approach for Web Security," International Journal of Engineering Research in Computer Science and Engineering (IJERCSE), pp. 128-132, 19 April 2018.

[6] J. P. Singh, "Analysis of SQL Injection Detection Techniques," Theoretical and Applied Informatics, May 2016. DOI : 10.48550/arXiv.1605.02796.

[7] M. Mohammed, M. B. Khan and E. B. Mohammed Bashier, Machine Learning: Algorithms and Applications, CRC Press, 2016. DOI: 10.1201/9781315371658.

[8] I. M. M. Matin and B. Rahardjo, "Malware Detection Using Honeypot and Machine Learning," in The 7th International Conference on Cyber and IT Service Management (CITSM 2019), Kuala Lumpur, 2019. DOI: 10.1109/CITSM47753.2019.8965419.

[9] W. B. Demilie and F. G. Deriba, "Detection and prevention of SQLI attacks and developing compressive framework using machine learning and hybrid techniques," Journal of Big Data, 2022. DOI:10.15199/48.2022.07.30.

[10] S. A. Krishnan, A. N. Sabu, P. P Sajan and A. Sreedeep, "SQL Injection Detection Using Machine Learning," Revista Gestão Inovação e

Tecnologias, pp. 300-310, June 2021. DOI:10.47059/revistageintec.v11i3.1939.
[11] F. Y. Hernawan, I. Hidayatulloh and I. F. Adam, "Hybrid method integrating SQL-IF and Naïve Bayes for SQL injection attack avoidance," Journal of Engineering and Applied Technology, vol. 1, no. 2, pp. 85-96, August 2020. DOI:10.21831/jeatech.v1i2.35497.
[12] H. K. Khanuja, P. Gadekar, S. Kulkarni, S. Kulkarni and S. More, "Web Application Security Scanning using Machine Learning," International Journal of Engineering Research in Computer Science and Engineering (IJERCSE), vol. 8, no. 8, pp. 21-27, August 2021. DOI : 01.1617/vol8/iss8/pid37860
[13] B. A. Pham and V. H. Subburaj, "An Experimental setup for Detecting SQLi Attacks using Machine Learning Algorithms," Journal of The Colloquium for Information Systems Security Education, vol. 8, no. 1, pp. 1-5, 2020. DOI:10.1007/978-3-031-28975-0_1.
[14] Y. Abdulmalik, "An Improved SQLInjection Attack Detection Model Using Machine Learning Techniques," International Journal of Innovative Computing, vol. 11, no. 1, pp. 53 - 57, 2021. DOI: 10.11113/ijic.v11n1.300.
[15] O. Morufu , R. A. Egigogo, I. Idris and R. G. Jimoh, "A Naïve Bayes Based Pattern Recognition Model for Detection and Categorization of Structured Query Language Injection Attack," International Journal of Cyber-Security and Digital Forensics (IJCSDF), pp. 189-199, 2018. DOI: 10.17781/P002396.
[16] J. E. T. Akinsola, O. Awodele and S. A. Idowu, "SQL Injection Attacks Predictive Analytics Using Supervised Machine Learning Techniques," International Journal of Computer Applications Technology and Research, vol. 9, no. 04, pp. 139-149, 2020. DOI:10.7753/IJCATR0904.1004.
[17] T.P. Latchoumi, M. S. Reddy and K. Balamurugan, "Applied Machine Learning Predictive Analytics to SQL Injection AttackDetection and Prevention," European Journal of Molecular & Clinical Medicine, vol. 7, no. 02, pp. 3543-3553, 2020.
[18] M. Kavitha, V. Vennila, G. Padmapriya and A. R. Kannan, "Prevention Of Sql Injection Attack Using Unsupervised Machine Learning Approach," International Journal of Aquatic Science ISSN: 2008-8019 vol. 12, no. 03, pp. 1413-1424, 2021.
[19] J. M. Alkhatami and S. M. Alzahrani, "Detection Of Sql Injection Attacks Using Machine Learning In Cloud Computing Platform," Journal of Theoretical and Applied Information Technology, E-ISSN: 1817-3195 vol. 100, no. 15, pp. 5446-5459, 15 August 2022.
[20] M. A. Azman, M. F. Marhusin and R. Sulaiman, "Machine Learning-Based Technique to Detect SQL Injection Attack," Journal of Computer Science, pp. 296-303, 2021. DOI:10.3844/jcssp.2021.296.303.
[21] U. Farooq, "Ensemble Machine Learning Approaches for Detection of SQL Injection Attack," in International Conference on Convergence of Smart Technologies IC2ST-2021, Pune, 2021. DOI: 10.31803/tg-20210205101347.
[22] S. Mishra, "SQL Injection Detection Using Machine Learning, Master's Projects," SJSU ScholarWorks, 5 May 2019.
[23] J. Triloka, H. Hartono and S. Sutedi, "Detection of SQL Injection Attack Using Machine Learning Based on Natural Language Processing," International Journal of Artificial Intelligence Research, vol. 6, no. 2, December 2022. DOI: 10.29099/ijair.v6i2.355.
[24] H. S. Anderson and P. Roth, "EMBER: An Open Dataset for Training Static PE Malware Machine Learning Models," 2018.
[25] M. Wang and C. Wang, "Detection of SQL Injection Attack Based on Improved TFIDF Algorithm," in International Conference on Mechanisms and Robotics (ICMAR 2022), Zhuhai, 2022. DOI: 10.1117/12.2652203.
[26] W.B. Demilie, F.G. Deriba, Detection and prevention of SQLI attacks and developing compressive framework using machine learning and hybrid techniques. J Big Data 9, 124 (2022). DOI: 10.1186/s40537-022-00678-0.