Artificial Intelligence for Human-Technologies-Economy Sustainable Development
23–25 May 2024
Lublin, Poland
make learn
TIIM
PIConf
Management, Knowledge and Learning
International Conference
Technology, Innovation and Industrial Management
Sustainable Development

Enhancing Web Security Through Comprehensive Evaluation of SQL Injection Detection Models

Abdulrasheed Jimoh
Gombe State University, Nigeria
jimohabdulrasheed1969@gmail.com

Muhammed Kabir Ahmed
Gombe State University, Nigeria
mkahmed@gsu.edu.ng

Suraj Salihu
Gombe State University, Nigeria
surajsalihu@gsu.edu.ng

Bala Mod
Gombe State University, Nigeria
bmodi@gsu.edu.ng

Mohammed Nasir Salihu
Tetfund Abuja, Nigeria
salihumn@gmail.com

Abstract

In the realm of cybersecurity, safeguarding web applications against SQLi attacks is important; it remains a severe threat to web security, necessitating robust detection methods. This study presents a comparative study of the efficacy of various deep learning and machine learning approaches in detecting SQLI attacks. The models evaluated in this research include ResNet, LSTM, CNN, RNN, GRU, Decision Tree, SVM, Random Forest, NaiveBayes and Logistic Regression based on critical performance metrics. Theresearch leveraged a Kaggle dataset containing 33,721 SQLi queries and normal texts. Our findings reveal that CNN emerges as a standout performer with an impressive Accuracy of 97.86% and a high Precision of 99.56%. RNN and LSTM exhibit commendable performance, with Accuracy and F1 Score exceeding 94%, emphasizing their adaptability to SQLi detection. Logistic Regression, Random Forest, and ResNet exhibit notable precision, while SVM achieves perfect Recall, indicating its strength in identifying harmful SQL queries. However, Naive Bayes demonstrates limitations in detection effectiveness, with an Accuracy of 58.94%. These findings underscore the efficiency of various models in SQLi detection, with uniqueadvantages and limitations. This research provides valuable insights for enhancing web security through optimized SQLi detection methodologies.

Keywords: SQLi attacks , Deep learning, Machine learning, Cybersecurity

1. Introduction

The advent of the digital age has brought about unprecedented advancements in technology, transforming the way individuals, businesses, and organizations operate and communicate. In today's digital landscape, where data fuels decision-making, cybersecurity stands as an essential pillar of safeguarding sensitive information. This digital revolution has ushered in an era of convenience, efficiency, and connectivity, with web-based applications and databases serving as the backbone of countless processes, from online shopping and financial transactions to healthcare management and communication[1]. Sensitive information are kept in large quantities in specialized databases on servers or elsewhere. Today, users may easily carry out their routine activities online as a result of the Internet's rapid expansion and its data-driven web services and applications[2]. These online applications are now used by the majority of contemporary businesses and people to conduct daily transactions. The risks associated with these applications are becoming more obvious as their use in social media networking, financial transactions, provision of healthcare services, etc.., grow rapidly[3]. Because any breach of cyber security would have a severe impact on the users' privacy, the institution's reputation, and its financial situation, organizations and industries must take extreme care to avoid unauthorized access to sensitive data[4].

The increased reliance on web-based applications has also opened the door to a new breed of cyber threats, posing significant risks to data security and privacy. However, cyber threats continue to evolve, and the perilous spectre of Structural Query Language Injection (SQLi) attacks looms large. SQL attacks, a nefarious form of cyber assault, exploit vulnerabilities in software applications to gain unauthorized access to databases[5]. SQLi attacks have emerged as a persistent and formidable adversary[6]. SQLi attacks work by inserting harmful SQL code into input fields [7]. The implications of successful SQLi assaults can be severe and far-reaching; some of the consequences include data breaches, unauthorized data manipulation, identity theft, reputational damage and significant financial losses to organizations. [8].In response to these threats, cybersecurity experts and researchers have been tirelessly working to develop effective countermeasures to detect and mitigate SQLi attacks[9]. Intrusion detection systems and firewalls are examples of conventional security measures that have proven insufficient in safeguarding against the evolving tactics employed by cybercriminals[10].

The objective of this study is to delve into the realm of cybersecurity and conduct a comprehensive comparison of SQLi attack detection frameworks. This exploration aims to shed light on the strengths, limitations, and practicality of each approach in thwarting SQLi attacks, ultimately contributing to a safer digital ecosystem. Throughout this study, we will investigate the intricacies of SQLi attacks, dissect the principles underpinning techniques of deep learning and machine learning, and evaluate their effectiveness in detecting SQLi intrusions using factors such as accuracy, precision, recall, and scalability in our comparison, providing valuable insights for organizations seeking robust cybersecurity solutions[11]. This challenge has given rise to innovative approaches, utilizing deep learning and machine learning methods emerging as a new frontier in SQLi attack detection. Systems may learn from data and generate predictions or judgments without explicit programming thanks to machine learning, which makes use of statistical models and algorithms.[12].

Open Web Application Security Project (OWASP)[13]indicate that injection-related weaknesses represent the most substantial and concerning security risk faced by websites and web applications in the current threat landscape. Among these injection vulnerabilities, SQLi stands out as the predominant method exploited by cybercriminals within this category,

highlighting the critical importance of addressing and mitigating these risks to guarantee the security and integrity of online platforms [14]. Injection vulnerabilities, including but not limited to SQLi, occur when attackers manipulate input fields or parameters of web applications to inject malicious code or data. This injected code is subsequently executed by the application's underlying systems, often leading to unauthorized access, data breaches, data manipulation, and potentially complete compromise of the targeted system[15]. The prevalence of injection attacks can be attributable to various elements: Ubiquity of SQL Databases, Inadequate Input Validation, Automated Attack Tools, and Financial Gain. To effectively reduce the dangers posed by injection assaults, web developers, security professionals, and organizations should adopt several best practices: Use of Prepared Statements and Query Parameterization, Validation and Sanitization of Input, Regular Security Audits, Security Training, Web Application Firewalls (WAFs), the OWASP's findings underscore the paramount significance of addressing injection vulnerabilities, with SQL injection being a leading concern. The combination of strong security measures, continuous monitoring, and robust development practices is crucial in safeguarding web applications against injection attacks and assuring the availability, integrity, and secrecy of sensitive information [16]. The rate at which cybercrime grows on a daily basis is quite worrisome, this noticeable alarming trend has been hurting several businesses, organizations, institutions is a great source of concern. According to Astra Global Cybercrime statistics (2022), $6 trillion damages have been incurred as a result of cybercrime in the year 2022[17].

Table 1: below shows a brief description of some few notable cyber-attacks worldwide from the year 2006 to 2021 [18].

YEAR	VICTIM	EFFECT	SUSPECT
2006	United nations, Olympic committee	Attempted Cyber attack	China
2014	Hold Security	5 million e-mail accounts hacked	Russian hackers' team
2014	Google	5 million Google passwords exposed	Russia
2014	Korean credit bureau	5 million credit cards have been compromised	Employee
2017	Notpetya	The attack on key infrastructure around the world has resulted in about $1 billion in losses.	The Russian military
2017	Wannacry	computers in 150 countries were compromised, The United Kingdom's national health services most severely harmed	Officials from North Korea's military intelligence
2020	Solarwind	A sophisticated operation involving 100 firms was launched	Russia has been blamed
2021	Microsoft	30,000 organizations in the United States are affected	Chinese cyber espionage suspected.
2021	Colonial pipeline	The suspension of a US gasoline pipeline has impacted 50 million users.	Russian-based group Darkside, according to the FBI

2. Overview of SQL

SQL is a powerful language for interacting with galleries, enabling users to carry out a variety of task SQLite[19].SQL is a domain-specific language that is standardized to be used for managing
relational databases and carrying out a variety of operations, such as adding, modifying, and deleting sets of data while extracting various subcategories of data stored in a database for transaction processing and analytic applications[20].SQL is a computer language that is used for interacting while using a relational database. It is also an instrument for organizing, retrieving as well asmanaging data from a database, Structured English Query Language was the original name for this technology, which is often shortened to SEQUEL by IBM. SQL utilized to create the request to obtain information from the gallary. The Database Management System, or the SQL query is processed, the necessary data is retrieved, and returned to the user. In other words, SQL statements specify what data should be extracted from or added to the database, as well as how a collection of data should be arranged [21].

2.1 Why use SQL?

SQL may be seen as a vital [22]indispensable tool for interacting with relational database due to the following [23]:

i. SQL functions as a client/server programming language, serving as a network protocol for computer programs to communicate with database servers storing shared data, commonly used in enterprise-class applications[24].
ii. SQL is an internet access language frequently used to access business databases on internet web servers and utility servers, often embedded within scripting languages like PHP or Perl[24].
iii. SQL is employed by distributed database control systems to distribute information among connected computer systems, with each device's DBMS software using SQL for communication and information retrieval[25].
iv. SQL functions as a database gateway language, enabling communication between different DBMS systems within computer networks, making it a valuable tool for connecting individuals, computer programs, and structures to relational database information[22].

2.2 SQL commands

To work with data contained in relational databases, developers utilize SQL commands particularly keywords or SQL statements[26]. The categories for SQL commands are listed below.

1. Data Definition Language (DDL)

These SQL commands are utilized to create the database structure using DDL; database engineers build and alter database objects according to business requirements. For example, a database engineer can construct database objects, including tables, views, and indexes, using the construct command. Table 2.1 below is a list of some DDLs [27].

Table 2.1: Example of DDL

S/N	Command	Function	Example
1.	CREATE	Builds a new database, table, a view from a table or another database object.	CREATE DATABASE test DB; CREATE TABLE Persons( );
2.	COMMENT	Used to make a remark or comment to the data dictionary.	--Select all: SELECT * FROM Customers;
3.	ALTER	This alters a database object that already exists.	ALTER TABLE Customers ADD Email varchar();
4.	DROP	deletes a view from a table or a complete table as well as supplementary database items.	ALTER TABLE Customers DROP COLUMN ContactName;
5.	TRUNCATE	This command removes the entire records from a table, including spaces allocated for records.	TRUNCATE TABLE Customer_details;

2. Data Manipulating Language: (DML)

DML commands can be used to add new data to a relational database, while the INSERT command used to create new records in the database, [28]examples;

Table 2.2 Data Manipulating Language.

S/N	Command	Function	Example
1	SELECT	This command retrieves information from a table or tables	SELECT column1,column2, ... FROM table_name;
2	INSERT	Insert creates or inputs records.	INSERT INTO Accounts (Name, Balance) VALUES (‘XYZ’,1234)
3	UPDATE	Update modifies an existing record.	UPDATE Accounts SET Balance = 5678 WHERE Name = ‘XYZ’
4	DELETE	Removes records.	DELETE FROM Accounts WHERE Name = ‘XYZ’
5	LOCK	This is used for table control concurrency.	SELECT COUNT(*) FROM artist WITH (TABLOCKX)

3. Data Query Language (DQL)

Retrieve datacommands are outlined in therelational database management language known as DQL[29]. In order to filter retrieve particular results from a SQL table, software applications employ the SELECT command. For example:
SELECT Name, FROM Accounts, WHERE Balance > 1234

4. Data Control language

Database administrators can manage or grant access to another users by using the programming language DCL. By way of the GRANT command, users can, for instance, permit particular applications to alter a number of tables,[30]example include:
Tale 2.3: Data Control Language.

S/N	Command	Function	Example
1	GRANT	This gives a privilege to the user.	GRANT SELECT ON employee TO user1;
2	REVOKE	This takes back the privilege given to the user.	REVOKE SELECT ON employee FROM user1;

5. Transaction Control Language(TCL)

TCL is used by the relational engine to continually update databases. For instance, the database can undo an erroneous transaction by using the ROLLBACK command.

2.3 Structured Query Language Injection

SQLi may be termed as allowing web applications to be used by hackers to send malicious code or query in order to obtain access to a database server to gain unauthorized access to sensitive information[31]. SQLi flaws can affect more than just a single website; they can also compromise entire database architecture and inhibit network system that hosts associated applications and the website itself. [32]and [33]. These problems may result in the spread of malware, remote server control, network instability, or even a breach in the confidentiality, integrity, and accessibility of important information. SQLi attacks can target any parameter from an application that can be utilised in a query to the database, including user input, cookies and server variables[34] and[35].

2.3.1 Sources of SQL Injection Attacks

Different hackers have different ways of launching attacks. Some inject harmful code by exploiting user input, either through the process of filling out a web application form or otherwise. Because most recent applications use cookies to store user's preferences, some attackers exploit that by embedding harmful code, thereby manipulating the application using the cookies to generate an injectable result[36], some leverage unvalidated databases to their advantage by inserting SQL injection into server variables, such as HTTP metadata and environmental elements used by the computer to audit usage statistics and discover trends in internet activity[37]. Others use what is known as second-order injection, which involves embedding a harmful input into a database. This, in turn, indirectly launches an SQL injection attack each time that input is executed[38].

2.3.2 Working of SQL Injection

According to Abdullayev and Chauhan (2023), SQLi attacks vary depending on the type of database engine; it normally executes based on the dynamic SQL statements using a web form or the URL query string, produced dynamically. The following statement is, however, vulnerable because the code directly utilizes the value in the $_POST[] array despite that it was encrypted using the md5 hashing algorithm.-

SELECT* FROM customers WHERE email = $POST['email'] where the password = md5($_POST['password']);

If for instance, an administrator enteredadministrator@gmail.com as the email address and amps as the password as below:

SELECT* FROM user WHERE email = administrator@gmail.com with a password = md5(amps). The intruder only requires commentingpart the password, thereby adding a dynamically conditional statement that will always execute to true, thereby bypassing the security measures employed.

SELECT* FROM customers WHERE email = 'xxxx@xxxx.xxxx' OR 1 = 1 LIMIT 1 --'] AND password = md5('password')

2.3.3 SQLi Attack Types

There are numerous types of SQLI attacks. They typically include conventional SQLi, as well known as in-band SQLi attacks; blind SQLi attacks, which are also described as inference SQLi attacks and band attacks, referred to as DNS attacks [35].

A. Classic or basic SQLi

A classic or basic SQL injection attack is when a user submits an SQL command and receives a result as a direct response via the same communication channel[39]. e.g., browser. Classic SQLi is of two types: union-based and injection-based on errors.

i. Union-based SQLi

Union-based SQLi techniques employ the SQL UNION operator for combining the outcomes of two or more chosen queries to produce a single outcome, that is to say, given back included in the HTTP reply. [40] example;

https://sqli.com/users/id=xyz 'UNION SELECT * FROM users, courses --

A malicious SQL query that can retrieve all user records is present in the URL mentioned above. All statements after the comment,"--" are ignored and would not be executed.

ii. Error-based SQLi

Error-based SQLi technique where the database server's error messages are gathered by SQLi to provide details about the database structure. A database can occasionally be completely enumerated by an attacker using simply error-based SQL injection[41].It can appear innocent to provide an attacker an error string. However, depending on the design of the application and the kind of database, the attacker could utilize the received error text to[42] and [43].

B. Inferential or blind - Based SQL Injection

Contrary to in-band SQLi, inferential SQLi might require more time for an intruder to accomplish, yet it is nevertheless every bit as harmful. Inferential SQLi attacks are sometimes known as "blind SQL injection attacks" since, in reality, no data is passed through the web-based program to observe the result of an in-and attack. Instead, by delivering payloads monitoring the reply from the web-based application, and tracking the server's subsequent actions, an intruder may modify the structure of the database [44], forms of inferential SQLi;

i. Boolean-based SQL Injection

In order for boolean-based SQLi to function, a SQL query must be submitted to the repository; based on whether the investigation returns a VALUE or FALSE outcome. Afterwards, an alternate reply must be provided by the application. The outcome will determine whether the HTTP response's content changes or stays the same. This enables the offender to ascertain whether the used the payload responded a value or false despite the fact no data in the database is transmitted[45].

ii. Time-based SQL injection

Time-based SQLi is a method that operates by submitting a query from SQL to the gallery that makes it delayed for a specified length of time (measured in seconds) that passed before responding. The attacker will be informed by the response time whether the query responds A VALUE or FALSE. An answer from HTTP will be given right away, depending on the outcome. Even though no database data is transferred, this enables us to ascertain whether the attacker utilized payload replied a value or false.[7].

c. Out-of-band SQL injection

Due to the intruder's incapacity to launch an attack and receive feedback on the same channel SQL injection happens. However, the intruder can still utilise an out-of-band route and retrieve data like the DNS or HTTP protocol. It also depends on the database server's enabled functionality, which the web application uses. As a result, this kind of approach is rarely used[46]. The attacker exploits the capability of the database server for sending DNS queries, such as the tree command in Microsoft SQL Server or through the utilising the UTL_HTTP package, HTTP requests can be sent from PL/SQL and SQL in Oracle database, to a server under the control of attacker in order to carry out an out-of-band SQL injection.

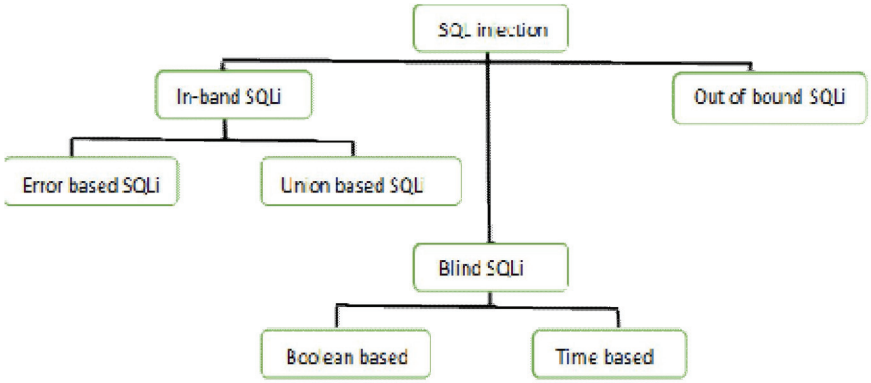

Figure 1*: SQL Injection Types*

2.4 Machine Learning Algorithms

Computers can learn from experience to perform better without being specifically programmed for each task; thanks to machine learning (ML),it is a method of training a model to automatically perform a given task with little or no human intervention[47]. Supervised learning is a learning paradigm involving training a computer system to gain knowledge and forecast using labelled data. The technique of imparting knowledge to a computer via labelled data is known as supervised learning[48], the term "supervised" signifies that the algorithm is guided and supervised throughout the learning process by providing it with a dataset in which both the intended output (or labels) that correspond to the input data are known,[47]. Unsupervised learning is a model trained to discover inherent patterns, structures, or relationships within a dataset that lacks labelled [49]. This is valuable when dealing with data for which the desired outcome is not known in advance or when seeking to reveal hidden insights from complex and unstructured datasets[50]. It learns by analyzing patterns in a given set of data and subsequently grouping the outcome by association of common features present in the data items. [48]. Reinforcement learning (RL) represents a dynamic and influential machine learning paradigm characterized by its capacity to teach computer agents how to make decisions and take actions by continuously interacting with their environment. [51]. This approach leverages the inherent intelligence of the agent, enabling it to make informed choices and actions within a given environment[52].

2.4.1 Logistic Regression

A type of regression called logistic regression is a fundamental supervised learning algorithm primarily used for classification jobs. it focuses on predicting categorical outcomes[53]. It is specifically designed to deal with binary classification tasks, where the objective is to classify data into either of the two groups, often represented as 0 or 1, true or false, yes or no, and so on; the resultant outcome is a discrete or categorical value, like 0 or 1, true or false, yes or no, e.t.c.The core concept of Logistic Regression lies in its ability to model probabilities, [54].

2.4.2 Decision tree

Decision trees are like structures (non-parametric trees) that are mostly used for classification and regression purposes in supervised learning techniques. They are flexible and used to build a machine-learning model which makes predictions by learning simple decision rules[55].In classification tasks, Decision Trees assign data points to specific categories or classes. At each node, the tree evaluates a feature and determines which branch to follow based on predefined criteria; this process keeps on until a leaf node is reached [54].In regression tasks, Decision Trees forecast a continuous quantitative value as the output. Instead

of class labels, the leaf nodes in a regression tree contain predicted numeric values. The tree structure guides the prediction process by considering the values of input features[56].

2.4.3 Random Forest

Random forest could be seen as an ensemble learning algorithm utilised for categorization and regression. It operates by logically combining a group based on decision trees together in the training phase [57]. The ensemble approach strives to enhance the overall performance, reduce overfitting, and enhance generalization as opposed to employing just one decision tree, while Decision Tree Base Learners at the core of Random Forest are decision trees similar to the ones discussed previously[50]. However, Random Forest leverages a forest of decision trees.

2.4.4 Naive Bayes

One probabilistic algorithm that is commonly used for classification problems belongs to the family of probabilistic classifiers, and it depends on the Bayes theorem, which is a basic idea in statistics and probability theory (Raschka, 2014). Despite the "naive" assumption and its simplicity, Naive Bayes has shown to be successful in a number of practical applications; it offers class labels to task instances depicting values of the vector from the source of the class labels.

2.4.5 Support Vector Machine (SVM)

SVM is an ML approach which handles both regression and classification tasks but is more efficient in solving classification problems. SVMs are a versatile and powerful class of algorithms primarily utilized for solving classification problem[58]. They are also capable of regression tasks, but their efficiency and effectiveness shine particularly in classification problems. SVMs are renowned for their ability to find an ideal hyperplane in high-dimensional spaces that maximises the margin between classes.

2.5 Deep Learning Algorisms

An artificial neuron, or perceptron, is the most fundamental unit or building block of the artificial neural network, influenced by the biological neurons in the brain and the nervous system. They receive a set of input values, each of which is then multiplied by a weight, the resultant value is summed up, and a bias is added before transmission to a subsequent neuron[59].An artificial neural network, a computational model that performs tasks such as prediction, classification, decision making, e.t.c., influenced by the principles of the biological neural network, which is the most basic unit of the brain and the nervous system, and also responsible for accepting input, processing it, and subsequently transmitting the output[60]. Learning could be supervised, unsupervised, or reinforced. In classical Machine learning, however, relevant features ought to be identified and manually extracted. Conversely, in deep learning, feature extraction is automatic, and is done in the hidden layer of the network [61].

2.5.1 Convolutional Neural Network (CNN)

CNN is a regularized form of feed-forward NN (Alom et al., 2019). CNN learns feature engineering on its own by means of filter (or kernel) optimization. Regularised weights are preferred over fewer connections to prevent vanishing gradients and expanding gradients, which are observed in earlier neural networks during backpropagation. In Feed Forward Network (FFN), all input data are passed to the network via the nodes layer of input, followed by the hidden layers or layer, and subsequently to the layer of output. [62]. There are no connections to redirect the output back to the input layer; hence, it is unidirectional. CNN finds its most

common application in image, audio and even word classification. In image classification, for instance, images are filtered by the convolutional layer, then converged by the pooling layer for easy computation, and then finally, the fully connected layer is not predicted [63].

2.5.2 Residual Network (ResNet)

ResNet is a deep learning framework where the weight layers acquire residual functions by means of the input layer. A structure with skip connections is referred to as residual for identity mappings, which complement the layer outputs to produce residual structures.It behaves as if its gates were opened by substantially positive bias weights, much such as highway network. This facilitates the rapid training of deep learning architectures with tens or hundreds of layers and approaches with greater accuracy[64].

2.5.3 Recurrent Neural Network (RNN)

An ANN that is bi-directional is called an RNNis one main type of artificial neural network identified by the direction in which information flows between their layers[65], which means that it permits the result of some nodes' output to influence additional data sent to the same nodes, as opposed to a unidirectional feedforward neural network. RNN suitability for assignments like speech recognition stems from their capacity to handle randomly chosen input sequences using internal state memory [66]. In RNN, the results obtained from the prior step are retransmitted back as input to the present step.

2.5.4 Long Short Time Memory (LSTM)

The LSTM network is a variant of RNN designed to overcome the vanishing gradient problem that typically befalls RNNs[67]. Its benefit Its advantage over hidden Markov models, other sequence learning techniques, and other RNNsis is that it is quite insensitive to gap length. The goal is to give RNN a "long short-term memory" that can withstand thousands of time steps. LSTM can be applied to voice recognition, video games, handwriting, speech activity detection, machine translation, robotic control as well as healthcare data classification, processing, and prediction based on time series [68].

2.5.6Gated Recurrent Unit (GRU)

Since it lacks an output gate, the GRU possesses fewer parameters compared to LSTM, but it functions similarly to one when it has a forget gate.[69]. Results from speech signal modelling, natural language processing, and polyphonic music modelling tasks revealed that GRU and LSTM fared comparably. GRUs demonstrated that gating is effective broadly speaking, but Bengio's team was unable to determine which of the two gating units was superior[70].

2.6 Related works

A two-class SVM (TCSVM) model was presented by[71] to forecast binary labelled outcomes regarding the positive or negative result of an SQLi attack in an internet request. To anticipate SQLi attacks, this model applied machine learning predictive analytics to absorb requests from the web at the proxy level.Two approaches are the foundation of the detection system[72] proposed. The initial form of detection centred on pattern matching, and it's equivalent to a detection method based on signatures technique that the classifier only looks at the HTTP URL within in an effort to locate an association from a database of SQL attack signatures. Machine learning methods formed the basis of the second detection technique employed. The authors gathered malicious data and used it to train the classifier by identifying

the characteristics that represented attacks in order to create this model. The SVM, naïve Bayes, and K-nearestneighbour algorithms were used.[73]recommended an SQLi detection system based on ResNet, capable of identifying SQL injection threats. The process involves tokenizing words, counting word occurrences, and vectorizing tokens into integer sequences. A user interface is designed for testing, classifying user questions as either malicious or innocent. Input data is normalized using a label encoder to convert non-numerical labels into numerical ones. Tokenization is performed using TF-IDF, word counts, or token frequency in the text to generate integer sequences. The TF-IDF approach assesses word frequency based on document occurrence, and term frequency (tf) measures token duplicates in a document. Inverse data frequency (idf) computes unusual token occurrences. The trained ResNet model excels in distinguishing between legitimate and fraudulent input requests, surpassing LSTM in various SQL injection attack identifications in their experiments.

A model named ATTAR was proposed by[74] to identify SQLi attacks through the extraction of SQLi attack features from web access log analysis. Recognition and access of grammar patterns and behaviour mining were used to select the features. This model's primary goal was to identify SQLi statements that were unfamiliar and hadn't been used in the training set before. The training was conducted using five machine learning techniques: random forest, k-means, ID3, SVM, and naive Bayesian. The outcomes of the experiment showed that the models with the highest accuracy in identifying SQLi attacks were those based on random forest and ID3.[75]introduced an Intelligent Transportation System Method for LSTM-based SQLi detection, combining online testing and offline training. In the offline phase, they collected diverse samples, categorized them as positive and negative, and preprocessed them after generalization, word segmentation, and word vector creation. The model was trained using this preprocessed data. Online testing mirrored the offline phase, where the trained classifier identified SQL injections. Their model was compared to popular deep learning methods (MLP, RNNCNN,) and machine learning (KNN, SVM, Decision Tree, RF, NB), utilizing Word2vec-based feature vectors and three hidden layers. CNN used a unique structure for object detection, while LSTM and RNN relied on a data sequence. The method excelled on dataset DS1 and outperformed RNN on DS5, primarily due to LSTM's ability to capture long-range dependence, making it more effective for SQL injection detection. This highlights the effectiveness of their data expansion strategy in deep learning.

In order to identify SQL injection attacks,[76] reviewed over 14 published studies that used both ML and DL techniques, such as CNN, LSTM, DBN, MLP, and Bi-LSTM, Cubic SVM, and Gaussian SVM. Additionally, they offered a comparison of the approaches' aims, strategies, features, and datasets. In a study by[37], a CNN and MLP approach for SQLI Attack Detection and Prevention was introduced. They employed word vector models created through lexical analysis and trained neural network models using both CNN and MLP. The model's training process was visualized with Tensorboard in TensorFlow to assess speed and usability. They used Google's Word2Vector, focusing on the CBOW model with 16-dimensional word embeddings. Comparing the models revealed differing performances in various contexts. The dataset was split into test and training subsets, with 4,000 SQL injection samples and 4,000 regular HTTP request samples intentionally chosen for testing. CNN was slower than MLP, taking around 26 seconds to process 4,000 data pieces. The MLP had two hidden layers, and the model contained 695,875 parameters.[77]suggested a technique for detecting SQL injection, involving tests conducted online and in offline mode. Training training information from from honeypots and vulnerability platforms was processed for feature extraction, considering factors like SQL statement length, keywords, and special characters. These features were used as inputs for the deep forest model. In online testing, decoded SQL statements were analyzed with the same features for classification. This method

enhances SQL injection security.In a performance comparison study by [78], feed-forward (MLP and CNN) and recurrent (LSTM and GRU) deep neural networks (DNNs) were evaluated for multiclass Network Intrusion Detection System (NIDS) tasks. MLP, CNN, and LSTM were chosen due to their frequent use in NIDS research. Additionally, GRU was included to explore potential advantages over LSTMs. The study found that GRUs might offer quicker training or better generalization with less data due to their reduced parameter count. All DNNs were trained using Adam optimization, promoting faster generalization with reduced cross-entropy loss between labels and predictions.

3. Material and Methodology

A comprehensive examination of five algorithms, each of CNN, RNN, GRU, LSTM, ResNET, Decision Tree, Logistic Regression, SVM, Random Forest, and Naive Bayes, was done to address the critical task of SQLi detection. SQLi remains a prevalent and persistent threat in the realm of web security, necessitating innovative and robust solutions for its detection and prevention. In pursuit of this objective, we embarked on a methodical journey encompassing the practical application and evaluation of ten distinct algorithms. The framework chosen for this study represents a diverse spectrum of machine learning techniques, each possessing unique characteristics and capabilities. They have been meticulously chosen to encompass traditional statistical approaches, ensemble methods, and cutting-edge deep learning architectures [79].

3.1 Dataset description

Data is a fundamental requirement, particularly in supervised machine learning. The main factor affecting the model's performance and behaviouron unobserved data is the training data.In this work a secondary sample dataset from https://www.kaggle.com/datasets was utilized in training the models. This dataset comprises two columns and 33,721 rows in the CSV file format. The first column is a field of values, which are either normal SQL statements, mere plain text, which are harmless or malicious codes used for SQL injection. The second column contains the labels for the statements in the first column. The value 1 depicts a malicious SQL statement while the value 0 means a normal SQL statement. The sample dataset comprises of 22,305 positive and 11,456 negative examples.

	Sentence	Label
33751	gorin.mccracken@evelin.am	0
33752	terrateig	0
33753	24220	0
33754	41	0
33755	yeti llamus	0
33756	syrett	0
33757	arrechea bellveh	0
33758	1664	0
33759	almaluez	0
33760	f6lo40r06	0

(A)

	Sentence	Label
0	NaN	1
1	" or pg_sleep (__TIME__) --	1
2	create user name identified by pass123 tempora...	1
3	%29	1
4	' AND 1 = utl_inaddr.get_host_address ((S...	1
5	select * from users where id = '1' or @ @1 = ...	1
6	select * from users where id = 1 or 1#" (un...	1
7	' select name from syscolumns where id = (...	1
8	select * from users where id = 1 +$+ or 1 = 1...	1
9	1; (load_file (char (47,101,116,99,47,112,9...	1

(B)

Figure 3.1: *(A) Sample without SQLi attack, (B) Sample with SQLi attack*

3.2 Data Preprocessing

In data preprocessing phase, null values are removed, sentences need to be converted to integer values using count vectorizer class, and tokenization is also carried out, which involves breaking down sequence of characters into smallerunitsknown as ‘tokens. It also involves removing certain characters sometimes. Regular expressions were employed for tokenizing all expressions as either normal plain text or injection.

3.3 Methodology for DetectingSQLi Attacks

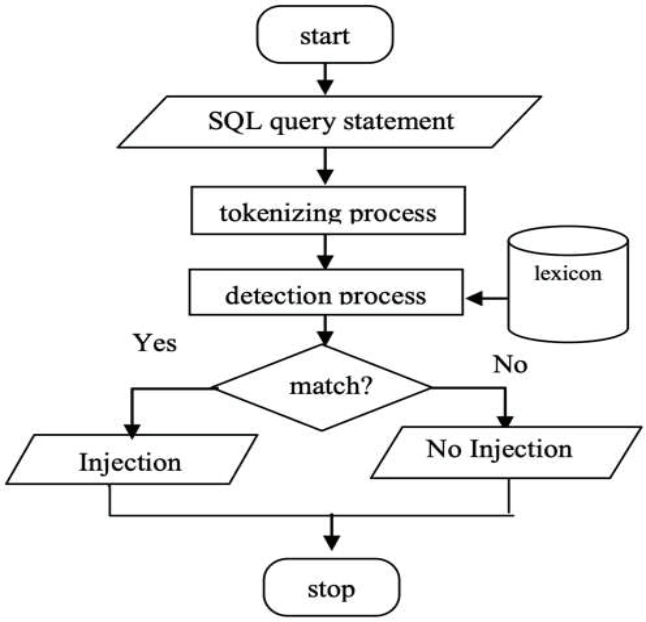

Figure 3.2:*Methodology for Identifying SQLi attacks*

Stage 1: The methods for identifying SQLi attacks described here aim to distinguish between safe and risky SQL query statements.

Stage 2: The second stage of this process begins with user inquiry tokenization, empty space, sharp signs (#), double dashes (-) as well as every string preceding every symbol is a token throughout the tokenization phase.

Stage 3: After the query was tokenized, stage 3 of the detection procedure involved comparing each string token to the words in a given lexicon. The lexicon contains the mainreserved phrases and number of logical operators.

Stage 4: Matching process, most of the injected commands or phrases that are gathered and used in SQL injection attacks. During execution, the input data is compared to the lexicon's contents to determine whether it is valid. An effort is made to use SQL injection if they match other terms. If the response is no, there won't be an injection.

Stage 5: End the process

3.4 Evaluation Metrics

To ascertain the efficiency of our models, we employed the metrics of accuracy, F1 score, precision, and recall in correctly identifying SQLi attacks while minimizing false positives and false negatives. True Negative Rate (TN) displays how many regularly anticipated requests were correctly made.The True Positive (TP) rate, provides the quantity ofmalicious queries

were seen, False Negative Rate (FN) provides the quantity erroneously anticipated. The false Positivity (FP) rate provides the number of harmful requests that were mistakenly forecasted.

Accuracy deals with the exactness of an SQLi model while quantifying the proportion of perfectly predicted SQLi attacks and safe requests to the entire number of forecasts the model generated. High accuracy indicates that a model correctly identifies both SQLi attacks and safe requests as mathematically represented in equation 3.1.

$$\text{Accuracy} = \frac{\text{Total Number of SQLi Predictions}}{\text{Number of Accurately Predicted SQLi}} \quad 3.1$$

Precision calculates how many of the predicted SQLi attacks were actually correct. It calculates the proportion of true positives or accurately anticipated SQLi attacks to, the total of requests that are predicted to be SQLi attacks but are not, or TP and FP, as defined by equation 3.2.

$$Precision = \frac{TP}{TP + FP} \quad 3.2$$

Recall, sometimes referred to asSensitivity or the rate of True Positive, and determines how well the model can identifySQLi attacks among all actual attacks.It calculates the amount of real positives to the amount of false negatives and true positives(SQLi attacks which were missed by the model) as showed in equation 3.3.

$$\text{Recall (Sensitivity)} = \frac{TP}{TP + FN} \quad 3.3$$

F1 Score offers a fair evaluation of recall and precision of SQLi detection by taking the harmonic mean of the two.It takes into account both false positives (safe requests misclassified as SQLi) and false negatives (missed SQLi attacks) as contained in equation 3.4

$$F_1 = \frac{2 * Precision * Recall}{Recision + Recall} \quad 3.4$$

4. Results and Discussion

In this study, various ML and DL models were evaluated, Our results are shown in table 4.1.

Table 4.1: Results of the 10 selected Machine Learning and Deep Learning SQLi Classifiers

Model	Accuracy	F1 Score	Precision	Recall
Logistic Regressi	96	94	88	99

on	. 1 6 %	. 0 4 %	. 9 5 %	. 7 6 %
Random Forest	9 6 . 1 4 %	9 4 . 0 6 %	8 9 . 9 0 %	9 8 . 6 2 %
Support Vector Machines	8 6 . 3 8 %	7 4 . 9 7 %	5 9 . 9 7 %	1 0 0 . 0 0 %
Naive Bayes	5 8 . 9 4 %	6 1 . 7 7 %	9 8 . 5 0 %	4 4 . 9 9 %
Decision Trees	9 5 . 9 6 %	9 3 . 6 6 %	8 9 . 7 9 %	9 7 . 8 6 %
RNN	9 6 . 1 4 %	9 4 . 1 2 %	9 0 . 1 6 %	9 8 . 4 4 %
CNN	9 7 . 8 6 %	9 6 . 2 0 %	9 9 . 5 6 %	9 3 . 0 6 %

LSTM	96.19%	94.21%	90.25%	98.54%
GRU	96.34%	94.37%	89.38%	99.95%
ResNet	96.33%	94.33%	89.89%	99.23%

In this evaluation, all models perform exceptionally well, with accuracy scores ranging from 58.94% to 97.86%. The CNN model stands out with the highest accuracy of 97.86%, indicating that it correctly classifies the majority of data points. On the other hand, Support Vector Machines (SVM) achieve the lowest accuracy of 86.38%, which is still a respectable performance. These findings demonstrate that the models generally have a strong ability to make correct predictions. Precision measures the capacity to predict outcomes accurately and classify positive cases in point. In our results, precision scores range from 59.97% to 99.56%. CNN also maintained the overall precision score of 99.56%, suggesting that it excels in identifying true positive instances while minimizing false positives. This is particularly crucial in applications where the cost of false positives is high. Naive Bayes also demonstrates impressive precision of 98.50%, indicating its ability to avoid misclassifying negative instances as positive. In contrast, SVM achieves the lowest precision of 59.97%, which implies a higher rate of false positives and recall values range from 44.99% to 100.00%.

The SVM model exhibits the highest recall of 100.00%, suggesting that it excels in identifying all true positive instances without missing any. This is a crucial characteristic in applications where avoiding false negatives is paramount. The Naive Bayes model has the least recall of 44.99%, proving that it may miss a significant number of true positive instances. The Recall and precision are combined to create the F1 Score, making it a valuable metric for binary classification tasks such as SQLi detection, especially when dealing with imbalanced datasets. In this evaluation, all models demonstrate competitive F1 Scores, ranging from 61.77% to 96.20%. Also CNN model achieves the highest F1 Score of 96.20%, indicating its balance between precision and recall. Naive Bayes, despite having lower accuracy, achieves a respectable F1 Score of 61.77%. This shows that Naive Bayes maintains the ideal ratio of recall to precision, even in the face of imbalanced data. In Overall performance, CNN performs exceptionally well across all metrics, achieving the highest accuracy, F1 Score, and precision.

4.2 Comparative Studies

Table 4.2:Presents a comparative analysis of our result with some from literature

Model	Ref.	Dataset	Dataset Size	Accuracy	F1 Score	Precision	Recall
LR	[our results]	Public Kaggle Dataset	33,721 22,305 positive and 11,456 negative	96.16%	94.04%	88.95%	99.76%
RF				96.14%	94.06%	89.90%	98.62%
SVM				86.38%	74.97%	59.97%	100.00%
NB				58.94%	61.77%	98.50%	44.99%
DT				95.96%	93.66%	89.79%	97.86%
RNN				96.14%	94.12%	90.16%	98.44%
CNN				97.86%	96.20%	99.56%	93.06%
LSTM				96.19%	94.21%	90.25%	98.54%
GRU				96.34%	94.37%	89.38%	99.95%
ResNet				96.33%	94.33%	89.89%	99.23%
Model	**Ref.**	**Dataset**	**Dataset Size**	**Accuracy**	**F1 Score**	**Precision**	**Recall**
DT	[74]	The Dataset were collected from 2 sources	2,750 SQLi cases of which 950 harmful 8800 non	93.4%	65.0%	76.6%	56.5%
RF				93.6%	66.0%	77.4%	57.7%
SVM				95.4%	73.2%	98.6%	58.3%
LR				95.1%	71.2%	98.5%	56.0%
RNN				95.3%	74.2%	92.2%	62.4%
LSTM				95.2%	73.4%	91.4%	61.4%
CNN				95.3%	74.3%	95.4%	59.9%
Cubic SVM	[76]	Open Source Dataset	616 SQL Statements	93.7%	-	-	-
Gaussian SVM				93.5%	-	-	-
LSTM	[72]	Open Source Dataset	Not mentioned	93.47%	93.99%	93.56%	92.43%
SVM	[71]	Private Dataset	450 malicious plus 59 benign SLQ queries	84.9%	87.6%	84.8%	91.1%
DT				89.4%	90.6%	96.3%	85.6%
RF				89.6%	91.7%	87.5%	96.4%

From Table 4.2: Logistic Regression**:** In "[our result]," exhibits high Accuracy (96.16%) and Recall (99.76%) whereas [15], maintains an Accuracy of (95.1%) and Precision (98.5%), emphasizing the importance of dataset choice. Random Forest: "[Our result]" demonstrates strong performance with high Accuracy (96.14%) and Recall (98.62%). Similar success is observed in [15], with RF achieving an Accuracy of (93.6%) and Precision (77.4%). Moreover, on the private dataset from [33], RF maintains competitive accuracy (89.6%) and precision (87.5%). SVM**:** "[Our result]" recorded a lower Accuracy of (86.38%) but perfect Recall (100.00%), showing its effectiveness in identifying harmful SQL queries. In [15], SVM exhibits equally high

Accuracy of (95.4%) and Precision (98.6%), emphasizing its adaptability across datasets. Decision Trees**:** "[Our result]" attains the best Accuracy (95.96%) and Precision (89.79%). Similarly, in [15], DT exhibits contending Accuracy of (93.4%) and Precision (76.6%). While from [33], DT maintains Accuracy (89.4%) with higher Precision of (96.3%). RNN**:** "[Our result]" demonstrates competitive performance with high Accuracy (96.14%) and F1 Score (94.12%). In [15], RNN achieves solid accuracy (95.3%) and high precision (92.2%).CNN**:** "[Our result]" excels with high Accuracy (97.86%) overall and Precision (99.56%). In [15], CNN maintains high Accuracy (95.3%) and Precision (95.4%). The results emphasize the adaptability of CNN in detecting SQL injection.LSTM**:** "[Our result]" performs well with high Accuracy (96.19%) and F1 Score (94.21%). In [15], LSTM maintains high Accuracy (95.2%) and F1 Score (73.4%).

5. Conclusion

This research offers insightful information about various algorism for the SQLI attacks detection within web-based applications. The results highlight the significance of choosing the right model to fortify web application security, with CNN, and GRU emerging as strong contenders. As the digital landscape continues to evolve, understanding the strengths

and weaknesses of these models is vital for ensuring the robustness of web applications against SQLI attacks. To effectively neutralize SQLi attack, the adoption of advanced techniques Capsule Networks, Transformer-based Models is strongly advised in future while a larger dataset not only enhances the model's ability to detect and classify SQLI attacks but also improves its generalization capabilities. It allows the model to learn intricate patterns, variations, and anomalies associated with SQLI attacks, thereby boosting its accuracy and reliability in real-world scenarios.

Reference

[1] Y. Hajjaji, W. Boulila, I.R. Farah, I. Romdhani, A. Hussain, Big data and IoT-based applications in smart environments: A systematic review, Computer Science Review. 39 (2021) 100318.

[2] F. Tao, Q. Qi, A. Liu, A. Kusiak, Data-driven smart manufacturing, Journal of Manufacturing Systems. 48 (2018) 157–169.

[3] O.C. Abikoye, A. Abubakar, A.H. Dokoro, O.N. Akande, A.A. Kayode, A novel technique to prevent SQL injection and cross-site scripting attacks using Knuth-Morris-Pratt string match algorithm, EURASIP Journal on Information Security. 2020 (2020) 1–14.

[4] A.B. Garba, J. Armarego, D. Murray, Bring your own device organizational information security and privacy, ARPN Journal of Engineering and Applied Sciences. 10 (2015) 1279–1287.

[5] D. Ghelani, T.K. Hua, S.K.R. Koduru, Cyber Security Threats, Vulnerabilities, and Security Solutions Models in Banking, Authorea Preprints. (2022).

[6] A. Akinola, A. Afonja, Introduction to Cyber-Security, ChudacePublishing, 2022.

[7] T. Pattewar, H. Patil, H. Patil, N. Patil, M. Taneja, T. Wadile, Detection of SQL injection using machine learning: a survey, Int. Res. J. Eng. Technol.(IRJET). 6 (2019) 239–246.

[8] M.M. Rashid, J. Kamruzzaman, M.M. Hassan, T. Imam, S. Gordon, Cyberattacks detection in iot-based smart city applications using machine learning techniques, International Journal of Environmental Research and Public Health. 17 (2020) 9347.

[9] A. Ketema, DEVELOPING SQL INJECTION PREVENTION MODEL USING DEEP LEARNING TECHNIQUE, (2022).

[10] V. Bandari, Enterprise Data Security Measures: A Comparative Review of Effectiveness and Risks Across Different Industries and Organization Types, International Journal of Business Intelligence and Big Data Analytics. 6 (2023) 1–11.

[11] D. Mitropoulos, P. Louridas, M. Polychronakis, A.D. Keromytis, Defending against web application attacks: approaches, challenges and implications, IEEE Transactions on Dependable and Secure Computing. 16 (2017) 188–203.

[12] A. Thakkar, R. Lohiya, A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges, Archives of Computational Methods in Engineering. 28 (2021) 3211–3243.

[13] J. Li, Vulnerabilities mapping based on OWASP-SANS: a survey for static application security testing (SAST), ArXiv Preprint ArXiv:2004.03216. (2020).

[14] I. Riadi, R. Umar, W. Sukarno, Vulnerability of injection attacks against the application security of framework based bebsites open web access security project (OWASP), J. Inform. 12 (2018) 53–57.

[15] D.P. Mozumder, J.N. Mahi, M. Whaiduzzaman, M.J.N. Mahi, Cloud computing security breaches and threats analysis, International Journal of Scientific & Engineering

Research. 8 (2017) 1287–1297.
[16] C.-A. Staicu, Enhancing the Security and Privacy of Full-Stack JavaScript Web Applications, (2020).
[17] H. Song, A. Holzer, Online Safety Laws by Country, Journal of Nonprofit Innovation. 3 (2023) 3.
[18] S. Swinford, B. Huesken, A. Mierow, H. Ariyaratne, S. Ismail, Computer Crimes, Am. Crim. L. Rev. 60 (2023) 579.
[19] P. Garner, J. Mariani, Learning SQL in steps, Learning. 12 (2015) 23.
[20] A. Rauber, A. Asmi, D. Van Uytvanck, S. Proell, Identification of reproducible subsets for data citation, sharing and re-use, Bulletin of IEEE Technical Committee on Digital Libraries, Special Issue on Data Citation. 12 (2016) 6–15.
[21] R. Batra, R. Batra, A history of SQL and relational databases, SQL Primer: An Accelerated Introduction to SQL Basics. (2018) 183–187.
[22] J.R. Groff, P.N. Weinberg, A.J. Oppel, SQL: the complete reference, McGraw-Hill/Osborne, 2002.
[23] S. Feuerstein, B. Pribyl, Oracle pl/sql Programming, " O'Reilly Media, Inc.," 2005.
[24] A. Oussous, F.-Z. Benjelloun, A.A. Lahcen, S. Belfkih, Big Data technologies: A survey, Journal of King Saud University-Computer and Information Sciences. 30 (2018) 431–448.
[25] C. Coronel, S.A. Morris, Database systems: design, implementation and management, Cengage learning, 2022.
[26] Y.N. Silva, I. Almeida, M. Queiroz, SQL: From traditional databases to big data, in: Proceedings of the 47th ACM Technical Symposium on Computing Science Education, 2016: pp. 413–418.
[27] R. Amornchewin, The development of sql language skills in data definition and data manipulation languages using exercises with Quizizz for students' learning engagement, Indonesian Journal of Informatics Education. 2 (2018) 85–90.
[28] J. Heller, Modify data with advanced DML, in: Pro Oracle SQL Development: Best Practices for Writing Advanced Queries, Springer, 2022: pp. 195–222.
[29] J.M. Patel, J.M. Patel, Relational databases and SQL language, Getting Structured Data from the Internet: Running Web Crawlers/Scrapers on a Big Data Production Scale. (2020) 225–275.
[30] S. Jahaj, Create a new login authentication and user authorization using MS SQL Server, (2021).
[31] S. Kini, A.P. Patil, M. Pooja, A. Balasubramanyam, SQL Injection Detection and Prevention using Aho-Corasick Pattern Matching Algorithm, in: 2022 3rd International Conference for Emerging Technology (INCET), IEEE, 2022: pp. 1–6.
[32] D. Sam, K. Nithya, S.D. Kanmani, A. Sheeba, A.S. Ebenezer, B.U. Maheswari, J.D. Amesh, Survey of risks and threats in online learning applications, Secure Data Management for Online Learning Applications. (2023) 2.
[33] N. Gharpure, A. Rai, Vulnerabilities and Threat Management in Relational Database Management Systems, in: 2022 5th International Conference on Advances in Science and Technology (ICAST), IEEE, 2022: pp. 369–374.
[34] F. Deriba, A.O. Salau, S.H. Mohammed, T.M. Kassa, W.B. Demilie, Development of a compressive framework using machine learning approaches for SQL injection attacks, PRZEGLĄD ELEKTROTECHNICZNY. 1 (2022) 183–189.
[35] N. Singh, P. Tiwari, SQL Injection Attacks, Detection Techniques on Web Application Databases, in: Rising Threats in Expert Applications and Solutions: Proceedings of FICR-TEAS 2022, Springer, 2022: pp. 387–394.

[36] S. Alagumuthukrishnan, L. Prabhu, C.A. Kumar, A.N. Kumar, Classification and tokenization algorithm to perceive and thwart injection attack by query in cyber systems, in: AIP Conference Proceedings, AIP Publishing, 2023.
[37] D. Chen, Q. Yan, C. Wu, J. Zhao, Sql injection attack detection and prevention techniques using deep learning, in: Journal of Physics: Conference Series, IOP Publishing, 2021: p. 12055.
[38] Y. Chen, Z. Pan, Y. Chen, Y. Li, DISOV: Discovering Second-Order Vulnerabilities Based on Web Application Property Graph, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences. 106 (2023) 133–145.
[39] A.K. Singh, Detection and Prevention of SQL Injection Attack in Web Application, (2011).
[40] D. Kar, S. Panigrahi, S. Sundararajan, SQLiGoT: Detecting SQL injection attacks using graph of tokens and SVM, Computers & Security. 60 (2016) 206–225.
[41] G. Su, F. Wang, Q. Li, Research on SQL injection vulnerability attack model, in: 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), IEEE, 2018: pp. 217–221.
[42] S. Roy, A.K. Singh, A.S. Sairam, Detecting and defeating SQL injection attacks, International Journal of Information and Electronics Engineering. 1 (2011) 38.
[43] J. Clarke-Salt, SQL injection attacks and defense, Elsevier, 2009.
[44] D. Ekeh, Detect and prevent SQL injection vulnerability, (2022).
[45] S. TOPRAK, A.G. YAVUZ, Web application firewall based on anomaly detection using deep learning, Acta Infologica. 6 (2022) 219–244.
[46] H.A. Noman, O.M.F. Abu-Sharkh, Code Injection Attacks in Wireless-Based Internet of Things (IoT): A Comprehensive Review and Practical Implementations, Sensors. 23 (2023) 6067.
[47] A. Pramod, H.S. Naicker, A.K. Tyagi, Machine learning and deep learning: Open issues and future research directions for the next 10 years, Computational Analysis and Deep Learning for Medical Care: Principles, Methods, and Applications. (2021) 463–490.
[48] B. Mahesh, Machine learning algorithms-a review, International Journal of Science and Research (IJSR).[Internet]. 9 (2020) 381–386.
[49] M. Wang, Y. Cui, X. Wang, S. Xiao, J. Jiang, Machine learning for networking: Workflow, advances and opportunities, Ieee Network. 32 (2017) 92–99.
[50] S. Sagiroglu, D. Sinanc, Big data: A review, in: 2013 International Conference on Collaboration Technologies and Systems (CTS), IEEE, 2013: pp. 42–47.
[51] E. Akanksha, N. Sharma, K. Gulati, Review on reinforcement learning, research evolution and scope of application, in: 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), IEEE, 2021: pp. 1416–1423.
[52] T.T. Nguyen, N.D. Nguyen, S. Nahavandi, Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications, IEEE Transactions on Cybernetics. 50 (2020) 3826–3839.
[53] G. Hackeling, Mastering Machine Learning with scikit-learn, Packt Publishing Ltd, 2017.
[54] S. Ray, A quick review of machine learning algorithms, in: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), IEEE, 2019: pp. 35–39.
[55] M. Hansen, R. Dubayah, R. DeFries, Classification trees: an alternative to traditional land cover classifiers, International Journal of Remote Sensing. 17 (1996) 1075–1081.
[56] M.P. Vayssières, R.E. Plant, B.H. Allen□Diaz, Classification trees: An alternative non□parametric approach for predicting species distributions, Journal of Vegetation

Science. 11 (2000) 679–694.
[57] O. Sagi, L. Rokach, Ensemble learning: A survey, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 8 (2018) e1249.
[58] M. Praveena, V. Jaiganesh, A literature review on supervised machine learning algorithms and boosting process, International Journal of Computer Applications. 169 (2017) 32–35.
[59] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Communications of the ACM, 2017: pp. 84–90. https://doi.org/10.1145/3065386.
[60] L.C. Voumik, R. Karthik, A. Ramamoorthy, A. Dutta, A Study on Mathematics Modeling using Fuzzy Logic and Artificial Neural Network for Medical Decision Making System, in: 2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), IEEE, 2023: pp. 492–498.
[61] S. Saeedi, S. Rezayi, H. Keshavarz, S. R Niakan Kalhori, MRI-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques, BMC Medical Informatics and Decision Making. 23 (2023) 1–17.
[62] I.G. and Y.B. and A. Courville, Deep learning 简介 ー 、什么是 Deep Learning ？, Nature. 29 (2016) 1–73. http://deeplearning.net/.
[63] Z. Zhou, M. Liu, W. Deng, Y. Wang, Z. Zhu, Clothing image classification algorithm based on convolutional neural network and optimized regularized extreme learning machine, Textile Research Journal. 92 (2022) 5106–5124.
[64] S.S. Saini, P. Rawat, Deep residual network for image recognition, in: 2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE), IEEE, 2022: pp. 1–4.
[65] S. Zhang, E. Shi, L. Wu, R. Wang, S. Yu, Z. Liu, S. Xu, T. Liu, S. Zhao, Differentiating brain states via multi-clip random fragment strategy-based interactive bidirectional recurrent neural network, Neural Networks. 165 (2023) 1035–1049.
[66] W. Lyu, Z. Zhang, C. Jiao, K. Qin, H. Zhang, Performance evaluation of channel decoding with deep neural networks, in: 2018 IEEE International Conference on Communications (ICC), IEEE, 2018: pp. 1–6.
[67] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, S. Valaee, Recent advances in recurrent neural networks, ArXiv Preprint ArXiv:1801.01078. (2017).
[68] S. Dutta, An overview on the evolution and adoption of deep learning applications used in the industry, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 8 (2018) e1257.
[69] A. Bhavani, A.V. Ramana, A.S.N. Chakravarthy, Comparative Analysis between LSTM and GRU in Stock Price Prediction, in: 2022 International Conference on Edge Computing and Applications (ICECAA), IEEE, 2022: pp. 532–537.
[70] Y. Li, Y. Chen, H. Shao, H. Zhang, A novel dual attention mechanism combined with knowledge for remaining useful life prediction based on gated recurrent units, Reliability Engineering & System Safety. 239 (2023) 109514.
[71] S.O. Uwagbole, W.J. Buchanan, L. Fan, Applied machine learning predictive analytics to SQL injection attack detection and prevention, in: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), IEEE, 2017: pp. 1087–1090.
[72] A. Makiou, Y. Begriche, A. Serhrouchni, Hybrid approach to detect SQLi attacks and evasion techniques, in: 10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, IEEE, 2014: pp. 452–456.
[73] S. Nagasundari, P.B. Honnavali, SQL injection attack detection using ResNet, in: 2019 10th International Conference on Computing, Communication and Networking

Technologies (ICCCNT), IEEE, 2019: pp. 1–7.
[74] H. Gao, J. Zhu, L. Liu, J. Xu, Y. Wu, A. Liu, Detecting SQL injection attacks using grammar pattern recognition and access behavior mining, in: 2019 IEEE International Conference on Energy Internet (ICEI), IEEE, 2019: pp. 493–498.
[75] Q. Li, F. Wang, J. Wang, W. Li, LSTM-based SQL injection detection method for intelligent transportation system, IEEE Transactions on Vehicular Technology. 68 (2019) 4182–4191.
[76] M.T. Muslihi, D. Alghazzawi, Detecting SQL Injection on Web Application Using Deep Learning Techniques: A Systematic Literature Review, in: 2020 Third International Conference on Vocational Education and Electrical Engineering (ICVEE), IEEE, 2020: pp. 1–6.
[77] Q. Li, W. Li, J. Wang, M. Cheng, A SQL injection detection method based on adaptive deep forest, IEEE Access. 7 (2019) 145385–145394.
[78] S. Caner, N. Erdoğmuş, Y.M. Erten, Performance analysis and feature selection for network-based intrusion detectionwith deep learning, Turkish Journal of Electrical Engineering and Computer Sciences. 30 (2022) 629–643.
[79] M.S. Yadav, M.S. Kumar, Web Application Security: Protection from Advanced Persistent Threat, (2018).