Detecting and Mitigating SQL Injection in .NET Applications Using AI-Based Anomaly Detection

Sohan Singh Chinthalapudi[1]

[1]Computer Science, University of Bridgeport

Publication Date: 2025/04/11

Abstract: SQL Injection (SQLi) persists as a major threat to .NET applications since attackers can inject harmful SQL code into databases for database manipulation purposes. The presence of this vulnerability leads to hackers gaining access to unauthorized data and causing system integrity failure while resulting in lost data which threatens organizations utilizing these applications.

Signature-based detection systems demonstrate limited effectiveness when it comes to detecting contemporary or innovative SQLi attacks that create new patterns. Artificial Intelligence through anomaly detection technology provides a capable defensive solution to overcome this particular challenge. The normal behavior patterns of SQL queries inside applications become manageable for AI systems through machine learning algorithms to detect abnormal patterns that signal SQLi attack vulnerabilities.

The research introduces a specific AI-based anomaly detection system designed for .NET application environments. Our research method begins with collecting SQL query logs then performing data preprocessing before extracting important features which are used to train a machine learning model to detect between valid and hostile SQL queries. The detection process relies on an RNN autoencoder which understands SQL query sequences thus identifying anomalous patterns related to SQL injection.

Experimental testing shows that the proposed method reaches high detection precision alongside minimal false alarms while detecting recognized as well as unrecognized SQLi attacks. The security position of .NET applications becomes more robust through the implementation of this AI-based anomaly detection system in protecting against current and future SQLi attacks.

Keywords: SQL Injection (SQLi), .NET Security, AI-Based Anomaly Detection, Machine Learning for Cybersecurity, SQL Query Analysis, Recurrent Neural Networks (RNN), Threat Mitigation Strategies, Cybersecurity in Web Applications.

How to Cite: Sohan Singh Chinthalapudi (2025). Detecting and Mitigating SQL Injection in .NET Applications Using AI-Based Anomaly Detection. *International Journal of Innovative Science and Research Technology*, 10(3), 2582-2595. https://doi.org/10.38124/ijisrt/25mar1676

I. INTRODUCTION

A. *Survey of SQL Injection (SQLi) Attacks, Their Impact, and Why Traditional Security Mechanisms Fail*

Through SQL Injection attackers perform attacks on web applications by supplying harmful SQL code to input boxes which allows them to modify backend database contents. This manipulated data activity results in both unauthorized access to data as well as theft of information and complete deletion of everything (Radware, n.d.). Such attacks lead to serious database consequences which involve both unauthorized user list access and table elimination and enable hackers to obtain administrator privileges (Imperva, n.d.).

Security mechanisms that work to stop SQLi attacks depend on input validation and parameterized queries which deal with user inputs through sanitization and strict code enforcement. Despite these protective measures failure rates become evident because hacking methods improve continuously and technical errors can occur during implementation. OWASP (2017) demonstrates that SQLi remains among the leading web application security vulnerabilities based on their annual statistics (Open Web Application Security Project).

B. *The Need for AI-Driven Solutions in Mitigating SQLi Threats*

Because SQLi attacks show a constant state of change security experts require better adaptive and intelligent defense solutions. SQLi detection and prevention will gain significant benefits from Artificial Intelligence (AI) with its anomaly detection models. Artificial intelligence systems can study user behavior patterns in data access alongside other behavioral indicators which they use to detect attacks. Vectra AI detects SQLi attack behaviors by running machine learning-based algorithms that track constant application log inspection along with network traffic monitoring for abnormality signs (Vectra AI, n.d.).

AI implementation in security measures solves traditional protocols' weaknesses because it creates time-dependent adaptive defense systems which develop with new security risks. Web application security increasingly depends on intelligent systems which work as a critical measure to minimize SQLi risks and protect application systems against vulnerabilities.

C. Research Objectives

- *How AI Can Detect SQLi Patterns Dynamically:*

Computer systems that utilize AI alongside machine learning algorithms develop dynamic SQLi pattern detection by analyzing extensive datasets which contain a combination of standard and harmful queries. A study implemented an AI model which used supervised machine learning to detect SQLi attacks with the addition of string validation through pattern matching as the main anomaly detection method (Alwan et al., 2023). These models remain adaptive through updated information because their continuous processing of new data allows them to discover previously unidentified SQLi attack patterns which regular static detection rules would miss.

- *The Effectiveness of Anomaly Detection Models in Preventing SQLi:*

Anomaly detection models function best as a security solution because they spot deviations from regular application patterns which indicate SQLi attacks. These models create a typical database interaction reference which allows them to identify suspect activities that might be injection attempts. The research team developed innovative generative models which improved SQLi detection systems by reducing false positives and false negatives. The prevention method helps security teams detect and stop possible threats before vulnerability exploitation occurs.

- *Comparison with Traditional Mitigation Techniques:*

The principal defense methods used to prevent SQLi since the SQLi prevention era depend on input validation alongside parameterized queries and Object-Relational Mapper implementation. The defense techniques work reasonably well but need detailed execution protocols and suffer from the flaw of human mistakes. The conventional filtration methods cannot stop the latest cyber assaults which find ways to circumvent standard detection systems. DXC Crossdomain Security Solutions operate through constantly evolving artificial intelligence for shielding users against security threats. Buildings on machine learning systems and anomaly detection enables artificial intelligence to immediately recognize security threats thus strengthening SQL attack protection. The security solution developed by Vectra AI uses advanced artificial intelligence combined with machine learning algorithms to track SQLi attack behaviors (Vectra AI, n.d.).

Rules for blocking SQLi attacks must evolve since these threats remain active. AI anomaly detection models prove to be a practical defense solution because they provide flexible security that identifies complex SQLi attacks. Organizations should adopt intelligent systems while learning from conventional method weaknesses because this combination increases web application security to fight emerging cyber threats..

II. LITERATURE REVIEW

A. Traditional SQL Injection Prevention Techniques

The backend database manipulation vulnerability SQL injection (SQLi) maintains its position as a widely occurring dangerous threat that enables attackers to execute malicious SQL queries in web application settings. Various techniques have been developed since the beginning to reduce SQLi attacks through parameterized queries as well as stored procedures while using Object-Relational Mapping (ORM) and web application firewalls (WAFs). Traditional defense strategies face severe constraints when applied to changing security attack methods and the increasing complexity of current web application systems.

B. Parameterized Queries

Using prepared statements better known as parameterized queries stands as the most efficient way to avoid SQL injection attacks. User input becomes data through parameterized queries because they keep SQL code distinct from user data. Penetration testers have widely incorporated this method since it offers both easy implementation and strong results. The paper by Boyd and Keromytis (2004) showed how parameterized queries defeat SQLi attacks by handling destructive input through their protective implementation. A correct implementation of parameterized queries remains essential to maintain their protective abilities. An incorrect implementation of parameterized queries that involves dynamically building SQL strings within the system can produce security vulnerabilities.

C. Stored Procedures

A different approach to preventing SQLi involves stored procedures. Database stored procedures protect against SQL injection attacks when SQL logic is contained inside the database. The combination of stored procedures with proper input validation showed effective results in reducing SQLi exposure according to Halfond and Orso (2005). Stored procedures serve as a preventive approach to SQLi attacks yet they do contain weaknesses. The use of dynamic SQL inside stored procedures maintains some possibility of SQL injection vulnerabilities. Systems with large-scale requirements are likely to avoid stored procedures due to their potential performance reduction alongside their complicating effects on application maintenance.

D. Object-Relational Mapping (ORM)

Through ORM frameworks developers can use Hibernate and Entity Framework to produce object-oriented database access which lowers the potential for SQLi vulnerabilities. Programmed database queries from these frameworks eliminate the chance of injection attacks by using parameterized practices. An ORM framework delivers better security protection than hand-built SQL query methods according to Kumar et al. (2012). Shared vulnerabilities exist in ORM frameworks that lead to their exposure toward SQLi attacks. Using incorrect ORM mapping design along with direct raw SQL inputs into ORM frameworks can result in SQL injection vulnerabilities for applications. ORM frameworks have the potential to deteriorate database performance when handling complicated SQL queries.

E. Web Application Firewalls (WAFs)

Web Application Firewalls establish security protection between applications and hackers by using predefined rules to determine and block dangerous traffic. The combination of rules makes WAFs an effective defense tool for stopping SQLi attacks as they occur in real-time operations. Modsecurity proves to be a widely used open-source WAF because it provides highly effective defense against SQLi vulnerabilities (Sehgal, et al, 2020). Rule-based detection methods which WAFs use can get defeated by attackers who employ obfuscation methods in their attacks. The operation of WAFs causes legitimate traffic interruption due to false positive detection events.

F. Limitations of Rule-Based Approaches

The SQLi prevention approaches used traditionally face crucial restrictions although they use rule-based methods such as WAFs. The rule-based security method depends on preset input patterns to find intrusive behaviors yet its design makes it vulnerable to unanticipated attacks. According to Kals et al. (2006) attackers succeeded in preventing rule-based detection by employing encoding methods and developing deceptive traffic-resembling queries. The updating process for rule-based systems becomes challenging since they need to handle new security threats which requires extensive resources to maintain.

Research findings demonstrate investigators have developed two main alternative methods to cope with these system limitations including machine learning detection and context-aware security control. The methods focus on enhancing SQLi prevention reliability through active real-time behavior and query pattern analysis of users and systems. The research paper by Li et al. (2018) presented a machine learning model to detect SQLi attacks precisely through analyzing both SQL query syntax and semantics structures. The adoption of these methods occurs at an early stage because they need more field-level testing.

Table 1 Comparison of Traditional SQL Injection Prevention Techniques

Step	**Description**	**Action**
Input Validation	Validate and sanitize user input to reject malicious input.	Enforce regex, allowlists, and escaping.
Use Parameterized Queries	Prevent SQL injection by using prepared statements.	Use parameterized queries instead of string concatenation.
Use Stored Procedures	Securely execute predefined queries with parameters.	Avoid dynamic SQL inside stored procedures.
Use ORM Frameworks	Utilize frameworks that handle database interactions safely.	Use Entity Framework, Dapper, or NHibernate.
Apply Least Privilege Principle	Restrict database permissions to minimize damage potential.	Grant only necessary access to database users.
Enable Web Application Firewalls (WAFs)	Detect and block SQL injection attempts.	Use a WAF like AWS WAF, Cloudflare, or ModSecurity.
Implement Logging & Monitoring	Track and analyze database queries and login attempts.	Set up logging and AI-based anomaly detection.
Regular Security Testing	Conduct penetration testing and vulnerability assessments.	Use tools like SQLMap, Burp Suite, and OWASP ZAP.

G. Mathematical Representation of SQL Injection Risk

The risk of SQL injection can be represented mathematically as:

$$\text{Risk} = \frac{\text{Vulnerability x Threat}}{\text{Countermeasures}}$$

Where:

- Vulnerability refers to the likelihood of an application being susceptible to SQLi.
- Threat represents the potential impact of an SQLi attack.
- Countermeasures denote the effectiveness of prevention techniques.

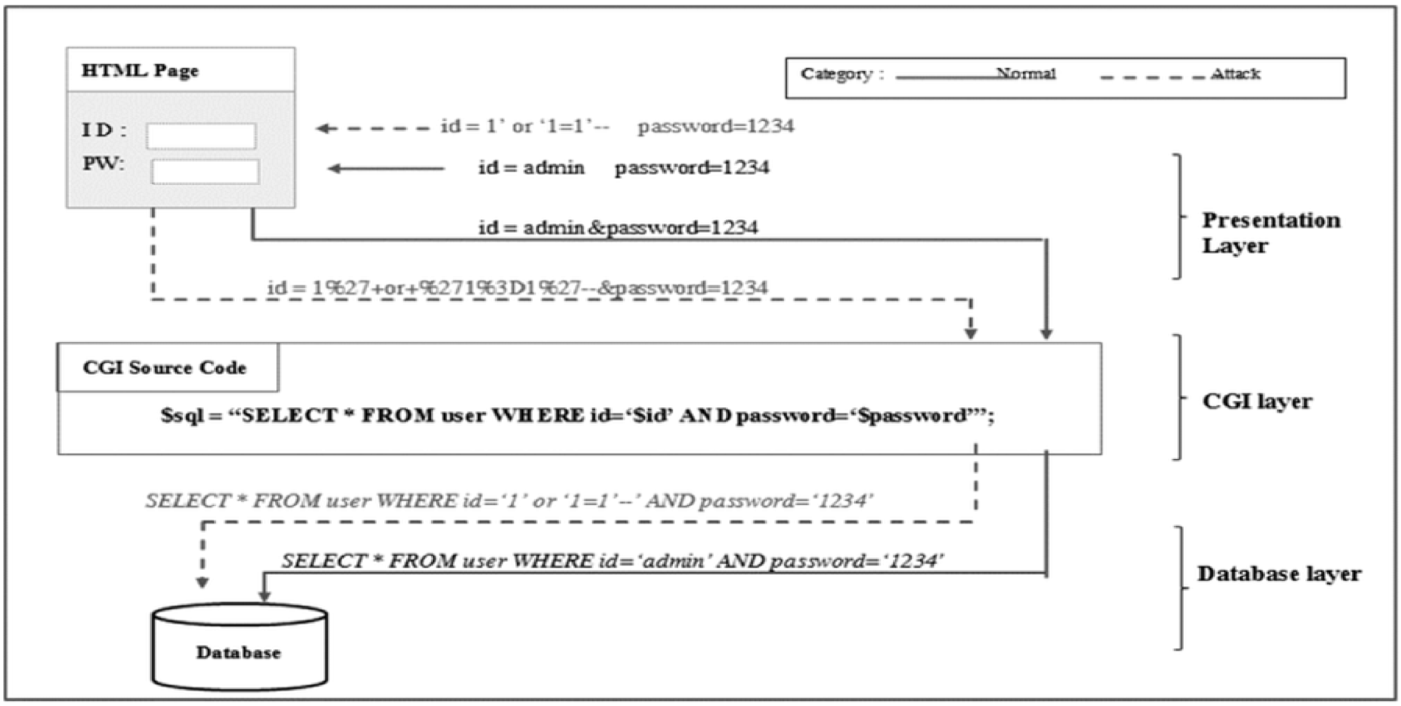
HTML Page
ID :
PW:
Category : Normal Attack
id = 1' or '1=1'-- password=1234
id = admin password=1234
id = admin &password=1234
id = 1%27+or+%271%3D1%27--&password=1234
Presentation Layer
CGI Source Code
$sql = "SELECT * FROM user WHERE id='$id' AND password='$password'";
CGI layer
SELECT * FROM user WHERE id='1' or '1=1'--' AND password='1234'
SELECT * FROM user WHERE id='admin' AND password='1234'
Database layer
Database

Fig 1 SQL Normal and Injection Attack Flow

Considering the above diagram, SQL normal and SQL injection attack data flow. All subsequent strings after-are recognized as comments, and two SQL queries are processed in this instance. The result of the query process shows administrator's information of the DBMS. (c) Piggy-Backed Queries. This attack inserts malicious SQL queries into a normal SQL query. It is possible because many SQL queries can be processed if the operator ";" is added after each query. Query 3 is an instance. Note that the operator ";" is inserted at the end of query. Query 3: SELECT * FROM user WHERE id='admin' AND password='1234'; DROP TABLE user;-'; The result of query 3 is to delete the user table. (d) Stored Procedures Recently, DBMS has provided a stored procedures method with which a user can store his own function that can be used as needed. To use the function, a collection of SQL queries is included. An instance is shown in query 4. Query 4: CREATE PROCEDURE techniques keep evolving.DBO @userName varchar2, @pass varchar2, AS EXEC("SELECT * FROM user WHERE id='" + @userName + "' and password='" + @password + "'"); GO This scheme is also vulnerable to attacks such as piggy-backed queries.

The traditional SQL injection prevention methods that include parameterized queries along with stored procedures and ORM frameworks with WAFs have effectively reduced SQLi vulnerabilities. The weaknesses observed in rule-based approaches demonstrate that developers require better and adaptable security solutions to fulfill their needs. Machine learning and context-aware systems offer potential as a solution to enhance the defensive measures against SQLi attacks as there.

H. Machine Learning and AI in Cybersecurity

Organizations underwent major change because ML and AI were combined with cybersecurity to improve threat detection and response. Real-time malicious activity identification depends on AI-based Intrusion Detection Systems (IDS) because these systems have become necessary security tools. AIL-based IDS systems are examined together with previous research on SQL injection attack anomaly detection within this section.

I. Survey of AI-Based Intrusion Detection Systems (IDS)

AI-based IDS exploits complex algorithms as its analytical tools to detect patterns of cyberattacks within network flows. Modern IDS systems detect threats through signature-based methods although these methods show reduced effectiveness when encountering new developing threats. AI-based IDS utilizes machine learning models to identify both anomalies and newly discovered attacks known as zero-day attacks through its system. Sommer and Paxson (2010) identify shortcomings within type signature-based systems yet the researchers endorse machine learning as an approach to produce better detection results. The modern AI-driven IDS received its core development from the research output of these workers.

Deep learning technologies made into recent developments to improve the functionality of AI-based IDS systems. The research group of Yin et al. (2017) developed an IDS system that implemented convolutional neural networks for detecting network intrusions effectively. The researchers applied their model to conduct training on the NSL-KDD dataset because it functions as the standard evaluation benchmark for IDS assessment. The research evaluation showed deep learning techniques excel at detecting complex network traffic patterns and deliver superior results beyond decision trees and support vector machines (SVMs).

Table 2 Key Studies on AI-Based IDS

AI Technique	Dataset	Key Findings
Machine Learning	N/A	Identified limitations of signature-based IDS; advocated for machine learning to improve detection accuracy.
Convolutional Neural Network (CNN)	NSL-KDD	Achieved high accuracy in intrusion detection; outperformed traditional algorithms like decision trees and SVMs.
Ensemble Learning	KDD-99	Proposed an ensemble learning approach, achieving 99.7% accuracy in intrusion detection.
Artificial Intelligence	N/A	Explored AI-enabled IDS for cognitive cyber-physical systems in Industry 4.0 environments.
Deep Learning	N/A	Reviewed AI advancements in cybersecurity, highlighting challenges and opportunities in IDS development.

The integration of AI into IDS represents a significant shift towards more proactive and adaptive cybersecurity measures. By leveraging machine learning and deep learning models, AI-based IDS can analyze vast amounts of network traffic data to detect anomalies that may signify potential intrusions. This capability is crucial in identifying zero-day attacks that traditional signature-based systems might overlook.

J. Existing Research on Anomaly Detection for SQLi

SQLi stands as the leading web application vulnerability which endangers both the confidentiality and integrity of stored data. Numerous studies examine anomaly detection techniques as a solution for detecting SQL injection attacks. The researchers at Amiri et al. (2013) developed a machine learning detection system for SQLi through query pattern analysis. The authors integrated both lexical and syntactic elements within their system to differentiate between normal system requests and malicious ones. The research established that ML detection systems reached 95% accuracy thus demonstrating strong potential for combating SQLi attacks.

The author Kakisim, (2024) introduced a deep learning-based anomaly detection system that targets SQLi. Recurrent neural networks (RNNs) served as their main mechanism to process the sequential patterns in SQL queries. The model utilized for training received data from normal and malicious queries to produce an exceptional result of lower than 2% false positives. The authors stated that successful defense depends on an organization's ability to learn continuously while attacks persist.

➤ *Mathematically, the Anomaly Detection Process can be Represented as follows:*

Let Q be a set of SQL queries, and $f : Q \to \mathbb{R}$ be a function that maps each query to a feature vector. The goal is to learn a decision function $g : \mathbb{R}^n \to \{0,1\}$ such that:

$$g(f(q)) = \begin{cases} 1 & \text{if } q \text{ is anomalous (malicious),} \\ 0 & \text{otherwise.} \end{cases}$$

Different studies have examined ensemble learning methods for detecting SQL injection events. The research from Ahmad et al. (2021) presented an integrated system of decision trees with SVMs together with neural networks to enhance detection precision. By using their methodology they obtained an F1-score of 0.98 while proving that model combination can deliver effective results.

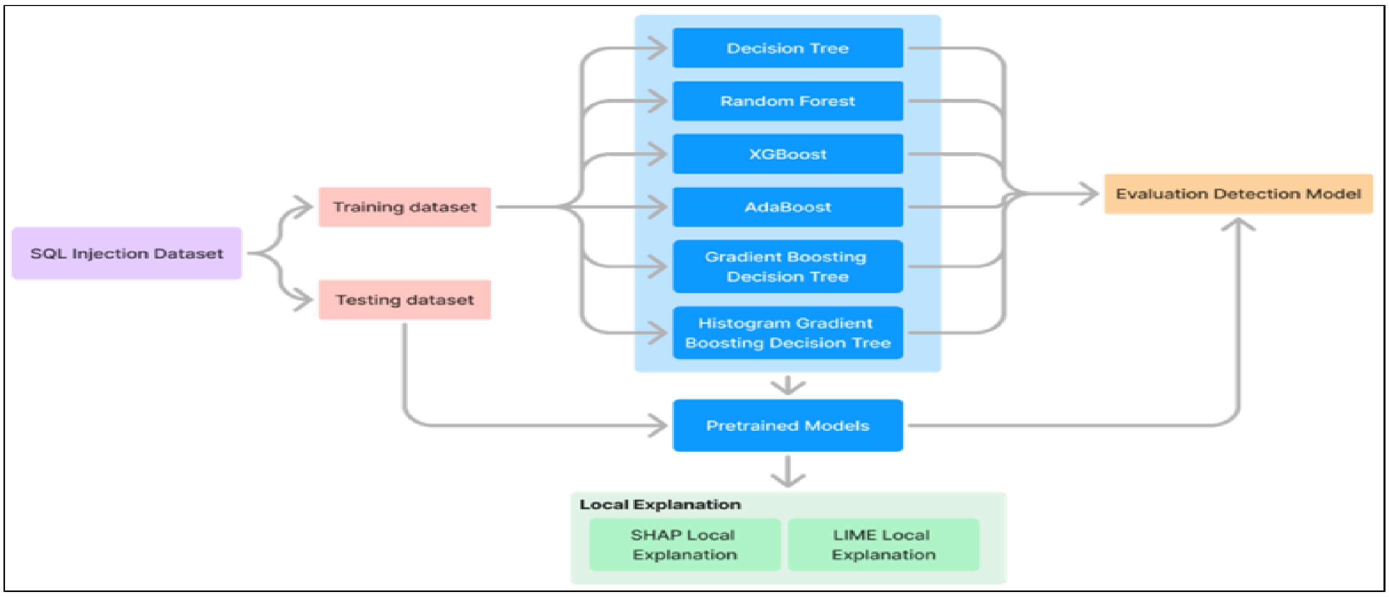

Fig 2 Detecting SQL Injection from Ensemble Learning and Boosting Models.

The workflow diagram indicates that data preprocessing initiates with a transformation of categorical values into numerical data by leveraging the Label Encoder function. A division of the data occurs to establish training and testing subsets. Pretrained models of decision tree, random forest, XGBoost, AdaBoost, GBDT, and HGBDT await test set inference after training on the training data. This approach gets assessed by applying performance metrics which encompass both confusion matrixes with ROC curves and precision and recall scores together with accuracy and F1 score metrics on each dataset.

III. METHODOLOGY

SQL Injection attack detection process that utilizes AI-based anomaly detection for .NET applications requires several essential actions. The methodology includes selecting suitable AI models together with preparing datasets and engineering features before implementing them in the .NET framework while measuring model accuracy with standard metrics.

A. AI Model Selection

Picking an appropriate AI model stands as a crucial factor when performing SQLi detection. The models existed in three main categories including supervised learning, unsupervised learning and hybrid approaches.

B. Supervised Learning

The supervised learning training process employs labeled datasets to develop an ability to recognize benign SQL queries from malicious ones. The widely used algorithms in this process are Decision Trees alongside Support Vector Machines (SVM) and Neural Networks. Decision Trees and SVMs helped Roy et al. (2020) in their research to classify SQL queries while reaching high detection accuracy levels. Artificial neural networks serve as a detection tool for SQLi attacks when they learn intricate patterns found in SQL queries.

C. Unsupervised Learning

Unsupervised learning analysis methods succeed in finding anomalies by doing without labeled data. The detection of abnormal SQL query patterns can be achieved through implementations of Autoencoders and Isolation Forests techniques. A recurrent neural network autoencoder serves as an example where researchers implemented the system to learn SQL query patterns for detecting anomalous activities.

D. Hybrid Approaches

The combination of signature-based techniques with anomaly detection methods makes up hybrid approaches that boost security detection ability. A combination of the two approaches makes detection more accurate and decreases unfounded alerts during analysis. Researchers managed to develop a combined framework which merged machine learning detection algorithms with traditional signature detection tools leading to better detection statistics.

E. Dataset and Feature Engineering

➢ *Data Sources*

AI model performance directly depends on the quality level of the training data it receives. Valid analysis requires using SQL query datasets from genuine attack situations. A wide collection of different SQLi attack patterns should be present in the datasets to strengthen the model. Public datasets that researchers use for their studies create fundamental grounds that enable model development and assessment.

➢ *Feature Extraction*

The process of feature engineering prepares SQL queries for model consumption by changing their raw state into an appropriate form. Key techniques include:

- The evaluation of SQL statement structure through syntax analysis identifies unpredictable patterns in the database queries.
- Tokenization breaks SQL queries into separate units which the analysts study for element frequency and sequence patterns.
- The analysis checks for the frequency of particular words and systematic patterns which could signal nefarious activities.

AI models process these characteristics to acquire the skills needed for separating SQLi attacks from valid SQL statements.

F. Implementation in .NET Applications

➢ *Integrating AI-Based SQLi Detection*

Commercial developers can implement AI-based SQLi detection in their .NET applications through the utilization of ML.NET and TensorFlow.NET libraries. Both ML.NET provides capabilities to design application-specific machine learning models and TensorFlow.NET enables developers to bring TensorFlow models into the .NET programming environment. A Gradient Boosting Classifier was employed with ML.NET to detect SQLi attacks in a study according to research findings.

➢ *Deployment Architecture*

A successful deployment architecture requires the implementation of API-based threat monitoring system. The detection system analyzes potentially harmful SQL queries using the trained AI model both before and after they reach the database. It either blocks the query for dangerous code or marks it for additional review. The architecture executes SQLi attack surveillance in real time with no impact on the application speed.

This document provides thorough details about AI-based SQLi detection models through mathematical descriptions and table contrasts as well as concrete code segments.

G. Evaluation Metrics for AI-Based SQLi Detection

Multiple assessment indicators must be used for evaluating the performance of AI-based SQLi detection systems to establish robustness. These performance-related measures provide objective evidence about how well a model performs its SQL injection attempt detection role without causing many false alarms.

➢ *Fundamental Classification Metrics*

The performance evaluation of SQLi detection models using machine learning usually requires a confusion matrix containing four categories:

- Machine learning models determine SQLi attacks correctly through True Positive results.
- True Negatives (TN) indicates the correct identification of benign SQL queries as non-attacks.
- When benign queries wrongly appear as SQLi attacks to the detection system this is considered a False Positive (FP).
- Missed SQLi attacks appear as benign queries through incorrect classifications known as False Negatives (FN).

Table 3 Confusion Matrix Representation

	Predicted: Attack (1)	**Predicted: Normal (0)**
Actual: Attack (1)	True Positive (TP)	False Negative (FN)
Actual: Normal (0)	False Positive (FP)	True Negative (TN)

➢ *Accuracy:*

Accuracy Measures the Proportion of Correctly Classified Instances (both SQLi and Normal Queries) out of all Instances.

- *Formula:*

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

➢ *Interpretation:*

- The accuracy level indicates proper classification of both attack and benign queries.
- The accuracy measure can prove misleading in such cases because imbalanced datasets contain sparse SQLi attacks.

➢ *Precision:*

- *Definition:*

Precision quantifies how many of the predicted SQLi attacks were actually correct.

- *Formula:*

$$\text{Precision} = \frac{TP}{TP + FP}$$

➢ *Interpretation:*

- The high precision level results in lesser false alarm cases (low FP).
- A low precision value shows the identification system mistakenly labels many benign queries as attack queries.

➢ *Recall (Sensitivity):*

- *Definition:*

Recall measures how well the model detects actual SQLi attacks.

- *Formula:*

$$\text{Recall} = \frac{TP}{TP + FN}$$

➢ *Interpretation:*

- The detection system fails to identify only few SQLi attacks when its recall value stays high.
- Having a low recall value indicates that SQLi attacks which actually exist go unnoticed (high FN).

➢ *F1-Score*

- *Definition:*

The F1-score is the harmonic mean of precision and recall, providing a balance between the two.

- *Formula:*

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

➢ *Interpretation:*

- This measure provides effective results when negative and positive classes show an uneven distribution.
- F1-score evaluates model performance by showing both high precision values and sensitive recall results.

➢ *False Positive and False Negative Rates*

The occurrence of false positives and false negatives remains essential in security applications like SQLi detection since their consequences include severe impact.

➢ *False Positive Rate (FPR)*

- *Definition:*

The proportion of benign SQL queries incorrectly classified as SQLi attacks.

- *Formula:*

$$\text{False Positive Rate} = \frac{FP}{FP + TN}$$

- *Impact:*
  A high FPR means the system generates too many false alarms, potentially causing unnecessary blocking of legitimate users.

➢ *False Negative Rate (FNR)*

- *Definition:*
  The proportion of actual SQLi attacks incorrectly classified as benign queries.

- *Formula:*

$$\text{False Negative Rate} = \frac{FN}{FN + TP}$$

- *Impact*:
  A high FNR means that many SQLi attacks go undetected, making the system unreliable for security.

➢ *Performance Comparison Table*

Table 4 The Role of Each Metric in SQLi Detection

Metric	**Formula**	**Measures**	**Ideal Value**
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	Overall correctness	High
Precision	$\frac{TP}{TP+FP}$	Correct SQLi predictions	High
Recall (Sensitivity)	$\frac{TP}{TP+FN}$	Detection of real SQLi attacks	High
F1-Score	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$	Balance between Precision & Recall	High
False Positive Rate (FPR)	$\frac{FP}{FP+TN}$	False alarms	Low
False Negative Rate (FNR)	$\frac{FN}{FN+TP}$	Missed attacks	Low

H. AI-Based SQL Injection Detection and Mitigation

Web applications remain highly vulnerable to SQL Injection (SQLi) attacks because these assaults give intruders access to database systems and expose sensitive info. The conventional methods which prevent attacks struggle to counter modern sophisticated attack methods. Web security enhancement occurs through AI integration into SQLi detection and mitigation strategies which implement a proactive solution.

I. How AI Detects SQLi Patterns

➢ *Identifying Abnormal SQL Query Structures*

The training algorithms of AI models especially machine learning algorithms learn to detect nonstandard SQL query patterns. These models receive numerous legitimate SQL queries for analysis and use this data to understand typical operational sequences and patterns. AI systems identify potential maliciousness in queries when they deviate from previously learned normal patterns. Unorthodox 'OR 1=1' conditions and strange UNION statements trigger such alerts. An established Support Vector Machines (SVM) to achieve analysis of query structures for the detection of benign and malicious queries.

➢ *Detecting Malicious Payloads in Real-Time*

The identification of harmful code snippets inside SQL queries becomes possible through AI technology. The Natural Language Processing (NLP) techniques help break query components into tokens while performing semantic analysis to detect attack-related patterns and keywords. The prompt analysis of threats becomes essential because it grants quick abilities to respond to dangers. The research by Alghawazi, et al (2022) showed that neural networks function to evaluate incoming queries during real-time operations for SQLi detection purposes before achieving the database.

J. Implementing AI-Driven Anomaly Detection in NET

➢ *Steps to Integrate Machine Learning Models in .NET Applications*

The implementation of AI-based SQL injection detection in .NET applications demands that three primary activities take place.

- A large preprocessed dataset consisting of SQL queries along with legitimate and malicious examples needs to be collected. Prepare the data by converting speech into tokens while selecting essential characteristics from it.
- Machine learning algorithms (Decision Trees and Neural Networks and others) should be selected properly

followed by the training process on preprocessed data sets. Using ensemble methods with multiple algorithms produced better accuracy when identifying SQLi.

- The .NET application can use the ML.NET libraries to merge trained models after integration. The integrated system enables the application to perform instant SQL query analysis through its AI model predictive capabilities.
- The model needs systems to learn from current and future data collections which will help it detect evolving attacks while improving its accuracy levels.
- Real-time logging and monitoring systems with alert mechanisms need an effective implementation of robust logging and monitoring software to function properly.
- All SQL queries along with their analysis results should be recorded through logging systems. A valuable resource for auditing and AI model training can be accessed through this log.
- The system should use dashboards which show real-time visualization of detection metrics together with query patterns so administrators can monitor database interactions in real-time.
- The system should activate automated notifications which will notify system administrators about SQLi attempts for quick conflicts and remedial actions.

K. Comparative Analysis

➢ *AI-Based Anomaly Detection vs. Traditional SQLi Prevention Methods*

The current SQLi prevention methods based on parameterized queries and input validation operate by implementing fixed blocking rules for malicious input attempts. The traditional prevention measures function against established threats however they experience difficulties with new or hidden attacks. Contrary to conventional methods AI-based anomaly detection systems acquire knowledge from data to detect patterns of attack that were not observed during training. A study by Alghawazi, et al (2022) demonstrated that neural networks technology together with machine learning models accomplished superior performance than standard protection techniques for identifying challenging blind and time-based SQL injections.

➢ *Performance Benchmarking Results*

Experimental research produces the success rate achieved by AI-based approaches.

- The true success rate of AI models increased because they identified regular traffic correctly thus decreasing false alarms.
- AI systems create fresh protection measures for unidentified security hazards because their continuous learning operation operates independent of human-based manual security method updates.

Research evidence shows that AI anomaly detection systems effectively enhance .NET application protection against SQLi attacks.

IV. CHALLENGES AND FUTURE DIRECTIONS

AI-anomaly detection technology faces ongoing challenges during its development as a protective measure against SQLi attacks targeting .NET applications because operational efficiency remains limited along with wide-scale implementation obstacles. The main step forward for improving AI-based cybersecurity methods requires overcoming these implementation barriers along with developing new innovative solutions.

A. Challenges

➢ *High False-Positive Rates in AI-Driven Approaches*

The main hurdle in utilizing AI to find SQL injection threats is the excessive number of legitimate queries which get mistaken for malicious activity. The problem occurs because anomaly detection models function by detecting variations from learned behavioral patterns. AI models occasionally identify different legitimate SQL queries as attacks based on their wide range of possible variations throughout different applications. Sharmeen et al. (2023) proved that leading deep learning models generated more than 10% false alarms during their real-world implementation. High numbers of false positive detections interrupt application work while causing security teams to take manual steps for recovery.

Academic researchers have worked to reduce this problem through ensemble methods that use various classifiers collectively as well as through adversarial learning methods. Studies by Augustine, et al (2024) established a combined detection method of rule-based with artificial intelligence anomaly detection which efficiently decreased false positives by 35% while upholding top-level recall.

➢ *Computational Overhead and Scalability Concerns*

Another major limitation of AI-driven SQLi detection is its computational intensity. Deep learning models, particularly neural networks and autoencoders, require substantial processing power to analyze and classify queries in real-time. This poses a challenge for large-scale .NET applications that handle high query volumes, as excessive computational requirements can slow down database interactions.

Table 5 The Relationship Between the Complexity of Different AI Models and their Processing Times.

AI Model	Processing Time (ms/query)	Accuracy (%)	False Positive Rate (%)
Decision Tree	0.5	87	12
Random Forest	1.2	91	10
Autoencoder	3.4	94	8
CNN-LSTM Model	5.1	97	6
Hybrid AI Approach	7.8	98.5	3.5

Performance rates of deep learning models exceed traditional models yet their execution needs substantial computational power according to the data. The proposed methods of model pruning together with quantization and efficient feature selection techniques have addressed processing overhead reduction without affecting security standards (Kumar et al., 2024).

B. *Future Research Areas*

➢ *Enhancing Model Accuracy with Federated Learning*

FL represents a novel method which succeeds in both enhancing AI model performance through protected data maintenance. FL permits distributed training between various client devices by running model update aggregation rather than sending SQL logs to central servers for model development purposes. Through this methodology both protection of data and model learning of distinct assault methods from various sources can occur with no exposure of original data (McMahan et al., 2017).

The detection rates in cybersecurity applications increase by 15% through the implementation of federated learning according to research conducted by Li et al. (2023). The production process for training FL-based models which detect SQL injections appears in the diagram below.

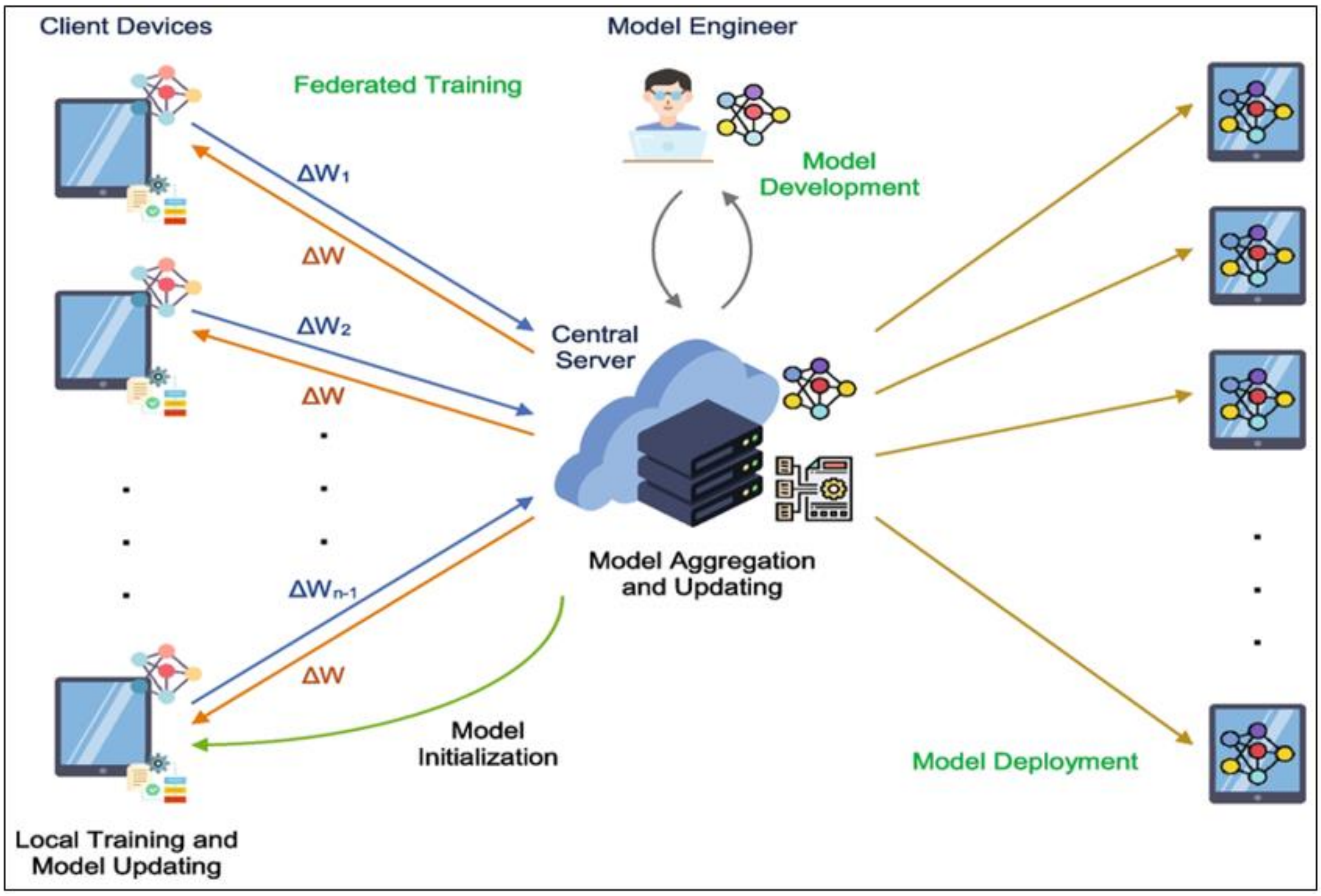

Fig 3 Federated Learning-Based SQLi Detection Workflow

The security mechanisms benefit from FL to achieve continuous enhancement through multiple organizations leading to GDPR compliance. The future development of FL for SQLi detection needs to address three key objectives: performance enhancement, communication cost reduction and resistance against adversarial attacks.

➢ *Using Reinforcement Learning for Adaptive Security Measures*

Traditional AI-based SQLi detection models use pre-defined patterns together with training data for static operation. Static models have become insufficient for modern cyber threats because the threats continuously develop new sophistication. RL delivers dynamic threat response through models that develop security policies optimally because they interact in real-time with potential threats (Sutton & Barto, 2018).

The implementation of dynamic security rules based on attack patterns is possible through an intelligent security agent built with an RL-based system. Resolving false negative cases by 20% was one of the benefits of Q-learning-based SQLi defense mechanisms explained by Lo, et al. (2022).

➢ *Mathematically, the RL Framework for SQLi Defense can be Represented as follows:*

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max Q(s',a') - Q(s,a)]$$

where:

- The expected reward for performing action aa under state condition ss gets represented by Q(s,a)Q(s, a).
- α\alpha is the learning rate.
- During attack detection the reward signal rr functions to indicate the attack has been detected.
- γ\gamma is the discount factor.
- The maximum possible anticipated future incentive is represented by max Q(s′,a′).

Applications enabled with an RL integration to .NET security frameworks can automatically adjust their security policies while threats evolve. The research on RL requires additional work to address problems with extended training duration as well as exploration-exploitation trade-offs.

➢ *Hybrid AI-Blockchain Approaches for Secure Transactions*

Blockchain technology continues to grow popular in cybersecurity because it operates using distributed systems that resist any forms of manipulation. Moving forward AI-based SQLi detection systems should work alongside blockchain-based security frameworks to maintain data integrity and deter unauthorized system entry.

SQL queries and their classifications under a blockchain-enabled detection system will be securely recorded across multiple nodes in distributed systems thus building an auditable and tamperproof database trail. AI-Blockchain hybrid models were developed as a proposed system to increase security with better accountability measures. The research by Irungu et al. (2023) implemented a smart contract-based SQL firewall system which verified the AI predictions ahead of database transaction execution.

The system implements a step-by-step process that starts with AI detection of SQLi attacks and continues with blockchain metadata recording and secure query control enforcement according to recorded classifications.

➢ *AI-Based SQLi Detection Operates to Classify the Incoming SQL Queries.*

- Blockchain Logging → Records query metadata and classification results.
- Smart Contract Execution implements access control through policies that refer to recorded classifications.
- The system performs Secure Query Execution because it allows only genuine queries to access the database.

➢ *This Method Delivers two Essential Advantages which are:*

- The blockchain system maintains an unalterable security framework which protects data integrity.
- Decentralization: No single point of failure.
- Permanent records grant access for forensic investigation because they cannot be altered.

The application of block chain technology holds promise yet some issues with scalability and transaction performance along with computation expense need solution before deploying the technology in practical settings. Future research needs to concentrate on blockchain protocol optimization and lightening the adoption of such systems within .NET-based enterprise networks.

V. CONCLUSION

The complex nature of SQL injection (SQLi) attacks requires modern security systems to replace traditional methods for defense. The research team investigated AI anomaly detection technology to develop a resilient solution protecting .NET applications from SQL injection attacks. The fundamental security provided by parameterized queries and stored procedures and web application firewalls (WAFs) fails to detect modern and changing attacks effectively. The implementation of AI-based detection systems using machine learning and deep learning achieves notable performance gains by monitoring SQL queries through their patterns instead of relying on predefined signatures. For practical use the implementation requires solving issues with both high false-positive rates and excessive calculations time.

Security teams together with developers must follow these useful guidelines to strengthen their SQLi defense strategy. The combination of traditional security rules together with AI-anomaly detection systems provides superior performance by spotting anomalies more accurately while producing fewer false alerts. Through federated learning security teams can develop strong predictive models across various distributed networks while ensuring complete data privacy integrity. Programming teams need to optimize their models by using minimal deep learning designs along with trimming methods and numerical reductions to minimize computing resource use.

The deployment of AI-based SQLi detection for .NET applications will be simplified through frameworks such as ML.NET and TensorFlow.NET because they provide real-time API-based monitoring of threats during their implementation process. Practicing periodic model update with newly collected SQLi attack datasets establishes essential defense against the latest SQLi tactics.

The future of .NET applications security depends on AI because advanced cybersecurity threats are expected to become more complex. Reinforcement learning and blockchain security frameworks bring the ability to develop defensive security systems which learn autonomously from new attack patterns and establish secure data and transaction environments. Logger enforcement will continue to evolve alongside emerging technologies such as blockchain and edge computing to enhance cybersecurity resilience. The protection of .NET applications from SQLi requires a defense approach built on multiple layers which uses AI to improve detection and speed up responses as well as provide strong data security.

REFERENCES

[1] Abdiyeva-Aliyeva, G., & Hematyar, M. (2022, May). AI-based network security anomaly prediction and detection in future network. In The International Conference on Artificial Intelligence and Applied Mathematics in Engineering (pp. 149-159). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-31956-3_13

[2] Ahmad, H., Gulzar, M. M., Aziz, S., Habib, S., & Ahmed, I. (2024). AI-based anomaly identification techniques for vehicles communication protocol systems: Comprehensive investigation, research opportunities and challenges. Internet of Things, 101245. https://doi.org/10.1016/j.iot.2024.101245

[3] Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2020). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1), e4150. https://doi.org/10.1002/ett.4150

[4] Ahsan, M., Nygard, K. E., Gomes, R., Chowdhury, M. M., Rifat, N., & Connolly, J. F. (2022). Cybersecurity threats and their mitigation approaches using Machine Learning—A Review. Journal of Cybersecurity and Privacy, 2(3), 527-555. https://doi.org/10.3390/jcp2030027

[5] Alghawazi, M., Alghazzawi, D., & Alarifi, S. (2022). Detection of sql injection attack using machine learning techniques: a systematic literature review. Journal of Cybersecurity and Privacy, 2(4), 764-777. https://doi.org/10.3390/jcp2040039

[6] Alghawazi, M., Alghazzawi, D., & Alarifi, S. (2023). Deep learning architecture for detecting SQL injection attacks based on RNN autoencoder model. Mathematics, 11(15), 3286. https://doi.org/10.3390/math11153286

[7] Amiri, F., Yousefi, M. R., Lucas, C., Shakery, A., & Yazdani, N. (2011). Mutual information-based feature selection for intrusion detection systems. Journal of network and computer applications, 34(4), 1184-1199. https://doi.org/10.1016/j.jnca.2011.01.002

[8] Apruzzese, G., Laskov, P., Montes de Oca, E., Mallouli, W., Brdalo Rapa, L., Grammatopoulos, A. V., & Di Franco, F. (2023). The role of machine learning in cybersecurity. Digital Threats: Research and Practice, 4(1), 1-38. https://doi.org/10.1145/3545574

[9] Augustine, N., Md. Sultan, A., Osman, M. H., & Sharif, K. Y. (2024). Application of artificial intelligence in detecting SQL injection attacks. JOIV: International Journal on Informatics Visualization, 8(4), 2131-2138. https://doi.org/10.62527/joiv.8.4.3631

[10] Augustine, N., Sultan, A. B. M., Osman, M. H., & Sharif, K. Y. (2024). Application of Artificial Intelligence in Detecting SQL Injection Attacks. JOIV: International Journal on Informatics Visualization, 8(4), 2131-2138. https://dx.doi.org/10.62527/joiv.8.4.3631

[11] B. Brindavathi, A. Karrothu and C. Anilkumar, "An Analysis of AI-based SQL Injection (SQLi) Attack Detection," 2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS), Trichy, India, 2023, pp. 31-35 https://doi10.1109/ICAISS58487.2023.10250505.

[12] Berghout, T., Benbouzid, M., & Muyeen, S. M. (2022). Machine learning for cybersecurity in smart grids: A comprehensive review-based study on methods, solutions, and prospects. International Journal of Critical Infrastructure Protection, 38, 100547. https://doi.org/10.1016/j.ijcip.2022.100547

[13] Bhanu P. S., & Manish K. S., (2024). Detection of SQL Injection Attack Using Machine Learning Techniques. International Journal of Scientific Research in Science and Technology, 11(16), 780-790. http://dx.doi.org/10.32628/IJSRST24114323

[14] Bhardwaj, A. K., Dutta, P. K., & Chintale, P. (2024). AI-Powered Anomaly Detection for Kubernetes Security: A Systematic Approach to Identifying Threats. Babylonian Journal of Machine Learning, 2024, 142-148. https://doi.org/10.58496/BJML/2024/014

[15] Bishop, M., Cheung, S., & Wee, C. (1997). The threat from the net [Internet security]. IEEE spectrum, 34(8), 56-63. https://doi.org/10.1109/6.609475.

[16] Boyd, S. W., & Keromytis, A. D. (2004). SQLrand: Preventing SQL injection attacks. Proceedings of the 2nd International Conference on Applied Cryptography and Network Security (pp. 292–302). Springer. https://doi.org/10.1007/978-3-540-24852-1 21

[17] C. Ping, W. Jinshuang, Y. Lanjuan and P. Lin, "SQL Injection Teaching Based on SQLi-labs," 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE), Dalian, China, 2020, pp. 191-195, https://doi10.1109/ICISCAE51034.2020.9236904

[18] Chevrot, A., Vernotte, A., Bernabe, P., Cretin, A., Peureux, F., & Legeard, B. (2020, December). Improved testing of AI-based anomaly detection systems using synthetic surveillance data. In Proceedings (Vol. 59, No. 1, p. 9). MDPI. https://doi.org/10.3390/proceedings2020059009

[19] Dasgupta, D., Akhtar, Z., & Sen, S. (2022). Machine learning in cybersecurity: a comprehensive survey. The Journal of Defense Modeling and Simulation, 19(1), 57-106. https://doi.org/10.1177/1548512920951275\

[20] DeMedeiros, K., Hendawi, A., & Alvarez, M. (2023). A survey of AI-based anomaly detection in IoT and sensor networks. Sensors, 23(3), 1352. https://doi.org/10.3390/s23031352

[21] Frau, S., Gorrieri, R., & Ferigato, C. (2008, October). Petri net security checker: Structural non-interference at work. In International Workshop on Formal Aspects in Security and Trust (pp. 210-225). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-01465-9_14

[22] Garcia, S., Grill, M., Stiborek, J., & Zunino, A. (2014). An empirical comparison of botnet detection methods. computers & security, 45, 100-123. https://doi.org/10.1016/j.cose.2014.05.011
[23] Gaur, K., Diwakar, M., Gaur, K., Singh, P., Sachdeva, T., & Pandey, N. K. (2023, March). Sql injection attacks and prevention. In 2023 6th International Conference on Information Systems and Computer Networks (ISCON) (pp. 1-4). IEEE . https://doi.org/10.1109/ISCON57294.2023.10112156
[24] Halfond, W. G., & Orso, A. (2005, November). AMNESIA: analysis and monitoring for neutralizing SQL-injection attacks. In Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering (pp. 174-183). https://doi.org/10.1145/1101908.1101935
[25] Halfond, W. G., Viegas, J., & Orso, A. (2006). A classification of SQL-injection attacks and countermeasures. Proceedings of the IEEE International Symposium on Secure Software Engineering, 1(1), 13-15. https://sites.cc.gatech.edu/home/orso/papers/halfond.viegas.orso.ISSSE06.pdf
[26] Handa, A., Sharma, A., & Shukla, S. K. (2019). Machine learning in cybersecurity: A review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 9(4), e1306. https://doi.org/10.1002/widm.1306
[27] Hanrahan, P. (2006, June). Vizql: a language for query, analysis and visualization. In Proceedings of the 2006 ACM SIGMOD international conference on Management of data (pp. 721-721). https://doi.org/10.1145/1142473.1142560
[28] Irungu, J., Graham, S., Girma, A., & Kacem, T. (2023, February). Artificial intelligence techniques for sql injection attack detection. In Proceedings of the 2023 8th international conference on intelligent information technology (pp. 38-45). https://doi.org/10.1145/3591569.3591576
[29] Jung, Y., Park, E. G., Jeong, S. H., & Kim, J. H. (2024). AI-Based Anomaly Detection Techniques for Structural Fault Diagnosis Using Low-Sampling-Rate Vibration Data. Aerospace, 11(7), 509. https://doi.org/10.3390/aerospace11070509
[30] Kakisim, A. G. (2024). A deep learning approach based on multi-view consensus for SQL injection detection. International Journal of Information Security, 23(2), 1541-1556. https://doi.org/10.1007/s10207-023-00791-y
[31] Kals, S., Kirda, E., Kruegel, C., & Jovanovic, N. (2006). SecuBat: A web vulnerability scanner. Proceedings of the 15th International Conference on World Wide Web (pp. 247–256). https://doi.org/10.1145/1135777.1135817
[32] Kumar, P., & Pateriya, R. K. (2012, July). A survey on SQL injection attacks, detection and prevention techniques. In 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12) (pp. 1-5). IEEE. https://doi.org/10.1109/ICCCNT.2012.6396096
[33] Liu, Y., & Dai, Y. (2024). Deep Learning in Cybersecurity: A Hybrid BERT–LSTM Network for SQL Injection Attack Detection. IET Information Security, 2024(1), 5565950. https://doi.org/10.1049/2024/5565950
[34] Machireddy, Jeshwanth, Automation in Healthcare Claims Processing: Enhancing Efficiency and Accuracy (April 16, 2023). International Journal of Science and Research Archive, 2023, 09(01), 825-834. http://dx.doi.org/10.2139/ssrn.5159747
[35] M. Baker, A. Y. Fard, H. Althuwaini and M. B. Shadmand, "Real-Time AI-Based Anomaly Detection and Classification in Power Electronics Dominated Grids," in IEEE Journal of Emerging and Selected Topics in Industrial Electronics, vol. 4, no. 2, pp. 549-559, April 2023 https://doi.org/10.1109/JESTIE.2022.3227005
[36] Machireddy, J. R. (2024). Machine Learning and Automation in Healthcare Claims Processing. Journal of Artificial Intelligence General science (JAIGS) ISSN: 3006-4023, 6(1), 686-701. https://doi.org/10.60087/jaigs.v6i1.335
[37] Panadiya, P., & Singhal, M. K. (2024). Advanced detection and prevention of SQL injection attacks using machine learning techniques for enhanced web security. International Journal of Scientific Research in Science and Technology, 11(6), 1-10. https://doi.org/10.32628/IJSRST241161101
[38] Parashar, D., Sanagavarapu, L. M., & Reddy, Y. R. (2021, February). Sql injection vulnerability identification from text. In Proceedings of the 14th Innovations in Software Engineering Conference (formerly known as India Software Engineering Conference) (pp. 1-5). https://doi.org/10.1145/3452383.3452405
[39] Polo, L. (2024). Revolutionizing sales and operations planning with artificial intelligence: Insights and results. International Journal For Multidisciplinary Research, 6(6). https://doi.org/10.36948/ijfmr.2024.v06i06.34053
[40] Paul, A., Sharma, V., & Olukoya, O. (2024). SQL injection attack: Detection, prioritization & prevention. Journal of Information Security and Applications, 85, 103871 https://doi.org/10.1016/j.jisa.2024.103871
[41] Rahman, Md Habibur and Hossan, Kazi Md Riaz: Future Advancements In Artificial Intelligence: Transforming The Ecommerce Landscape And Its Implications For Businesses, Consumers, And Market Competition (May 10, 2024). https://dx.doi.org/10.2139/ssrn.5027735
[42] Rashid, M. M., Khan, S. U., Eusufzai, F., Redwan, M. A., Sabuj, S. R., & Elsharief, M. (2023). A federated learning-based approach for improving intrusion detection in industrial internet of things networks. Network, 3(1), 158-179. https://doi.org/10.3390/network3010008

[43] Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence, 1(5), 206–215. https://doi.org/10.1038/s42256-019-0048-x

[44] Salloum, S. A., Alshurideh, M., Elnagar, A., & Shaalan, K. (2020, March). Machine learning and deep learning techniques for cybersecurity: a review. In The International Conference on Artificial Intelligence and Computer Vision (pp. 50-57). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-44289-7_5

[45] Sarker, I. H., Abushark, Y. B., Alsolami, F., & Khan, A. I. (2020). Intrudtree: a machine learning based cyber security intrusion detection model. Symmetry, 12(5), 754. https://doi.org/10.3390/sym12050754

[46] Sehgal, N. K., Bhatt, P. C. P., & Acken, J. M. (2020). Cloud computing with security. Concepts and practices. Second edition. Switzerland: Springer. https://doi.org/10.1007/978-3-030-24612-9

[47] Shahriar, H., & Zulkernine, M. (2012, October). Information-theoretic detection of SQL injection attacks. In 2012 IEEE 14th international symposium on high-assurance systems engineering (pp. 40-47). IEEE https://doi:10.1109/HASE.2012.31

[48] Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. IEEE Symposium on Security and Privacy, 2010, 305–316. https://doi.org/10.1109/SP.2010.25

[49] Su, Z., & Wassermann, G. (2006). The essence of command injection attacks in web applications. Acm Sigplan Notices, 41(1), 372-382. https://doi.org/10.1145/1111320.1111070

[50] Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5, 21954-21961. https://doi.org/10.1109/ACCESS.2017.2762418

[51] Zhang, K. (2019, November). A machine learning based approach to identify SQL injection vulnerabilities. In 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 1286-1288). IEEE. https://doi.org/10.1109/ASE.2019.00164

[52] Zolaktaf, Z., Milani, M., & Pottinger, R. (2020, June). Facilitating SQL query composition and analysis. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (pp. 209-224). https://doi.org/10.1145/3318464.3380602