IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 23 October 2024, accepted 7 November 2024, date of publication 19 November 2024,
date of current version 4 December 2024.

*Digital Object Identifier 10.1109/ACCESS.2024.3502293*

RESEARCH ARTICLE

# SIDNet: A SQL Injection Detection Network for Enhancing Cybersecurity

**DEBENDRA MUDULI[1], (Member, IEEE), SHANTANU SHOOKDEB[1], ABU TAHA ZAMANI[2], (Member, IEEE), SURABHI SAXENA[3], ANURADHA SHANTANU KANADE[4], NIKHAT PARVEEN[5], AND MOHAMMAD SHAMEEM[6]**

[1]Department of Computer Science and Engineering, C. V. Raman Global University, Bhubaneswar, Odisha 752054, India
[2]Department of Computer Science, Northern Border University, Arar 91431, Saudi Arabia
[3]Department of Computer Science, CHRIST University, Bengaluru, India
[4]Department of Computer Science and Applications, Dr. Vishwanath Karad-MIT-World Peace University, Pune 411038, India
[5]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation Deemed to be University, Guntur, Andhra Pradesh 522302, India
[6]Interdisciplinary Research Center for Intelligent and Secure Systems, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia

Corresponding author: Debendra Muduli (muduli.debendra@gmail.com)

This work was supported by the Deanship of Scientific Research, Northern Border University, Arar, Saudi Arabia, under Project NBU-FFR-2024-1850-02.

**ABSTRACT** SQL (Structured Query Language) injection is one of the most prevalent and dangerous forms of cyber-attacks, posing significant threats to database management systems and the overall security of web applications. By exploiting vulnerabilities in web applications, attackers can execute malicious SQL statements, potentially compromising the integrity and confidentiality of critical data. To combat these threats, in this study, we introduce two novel CNN models, SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2), specifically designed for the classification of SQL injection attacks to bolster web application security. Our comprehensive evaluation includes a comparison of the performance of these customized CNN models against traditional machine learning approaches, highlighting improvements in classification accuracy and reductions in false alarm rates. The proposed models have been experimented with two publicly available dataset SQLI (SQL-Injection) and SQLV2 (SQL-Injection version2). Specifically, SIDNet-1 achieves an impressive accuracy of 98.02% on the SQLI dataset, while SIDNet-2 closely follows with 97.54%. Furthermore, on the SQLIV2 dataset, SIDNet-1 attains 97.77%, and SIDNet-2 achieves 97.83% accuracy respectively.

**INDEX TERMS** SIDNet, SQLI, cyber security, CNN.

## I. INTRODUCTION

SQL injection is a prevalent attack vector where malicious actors exploit vulnerabilities in a database-driven application by injecting malicious SQL statements into input fields. This vulnerability arises when user inputs are not adequately sanitised, allowing the attacker to manipulate the SQL queries executed by the database. The primary goal of SQL injection attacks is to access, manipulate, or exfiltrate sensitive data from the database, potentially leading to unauthorised data disclosure, data loss, or corruption. Additionally, attackers may aim to gain administrative privileges, execute remote commands, or disrupt database operations [14]. SQL injection detection involves the use of various techniques to identify and block malicious SQL code. Methods such as signature-based detection, anomaly-based detection, and the use of honeypots can help in identifying suspicious activities. Web Application Firewalls (WAFs) play a crucial role in SQL injection mitigation by filtering out malicious inputs and providing an additional layer of security. However, sophisticated attackers can sometimes bypass WAFs using advanced evasion techniques, rendering these defences less effective [7]. The impact of SQL injection on industries is profound, as it can lead to significant financial losses,

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa M. Fouda.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

reputational damage, and legal consequences. Data leaks resulting from SQL injection attacks often result in sensitive information being sold on the dark web, exacerbating the damage to affected organisations and individuals. High-profile breaches have underscored the need for robust database security practices and have driven the adoption of more stringent regulatory frameworks. Effective mitigation strategies, including regular code reviews, parameterized queries, and comprehensive security training, are essential to protect against SQL injection and ensure the integrity and confidentiality of data. Whereas, the database is a significant part of the web application [42] where database collection users information from the online web platform, retrieval and manage information. It plays important role as the backbone of web app [28], storing and managing data such as digital information of users, content and other vital components that directly essential for web application functionality [12]. Database stores structured and unstructured data, consist of users profiles, content, transactions, and session data [15]. Database retrieving data using queries, data filters, and sorting algorithms to fulfill user requests and application functionalities. It verifying user identities and managing access permissions to secure authentication and verification for application functionalities [26]. Additionally it controls user access based on roles and privileges [11]. In the context of relational statement in controls and privileges, SQL (Structured Query Language) injection attacks [32] describe a dangerous threat on web security policy where web applications are exploiting based on vulnerabilities that allow them to interfere with the queries that an application makes to its database [13]. Its a technique where an attacker injects SQL (Structured Query Language) commands into an input field for execution to intercept on the database to inject their data or digital information for unauthorized access to sensitive data, data theft, or destruction [14]. In the context of prevention, it ensures only expected data accepted by the application [24]. In parameterized queries, attackers used prepared statements that distinguish between code and data [37]. this technique encapsulates database queries within the database itself. In object-relational mappings, it utilizes frameworks that reduce the risk of injection [39]. It could be prevented by implementing web application firewalls and intrusion for vulnerabilities. Machine learning techniques have revolutionized medical image analysis by enabling more accurate, efficient, and automated diagnosis, thereby enhancing patient care and clinical decision-making [19], [20], [21], [22], [35]. Inspired by these advancements, we have applied similar methodologies [23] in our application focused on cybersecurity.

In this study, We focus on the detection and identification based on our used dataset. We proposed two customized CNN models: SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2). Machine learning algorithms and customized CNNs models are developed for identification and classification of SQL (Structured Query Language) injection and normal data inputs. In the sense of classification, our proposed model performed well in the context of categorizing the SQL (Structured Query Language) injection and normal data inputs in web applications [30]. Furthermore more we have implemented research and advancements in several machine learning algorithms that represent a promising robust defense mechanism against SQL (Structured Query Language) injection attacks which ensure the security and integrity of web applications.

- In this study, we propose two customized CNN models, SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2), that ignore handcrafted feature extraction and selection.
- We reduce the overfitting issues in our own data sample, and with a number of fewer parameters, we successfully complete the compilation and prediction of the data class.
- We discuss and analyze the significance and importance of the proposed model with other popular machine learning models compared with our customized proposed SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2) model.
- This research paper aims to investigate the efficacy of two novel deep neural networks tailored for detecting SQL (Structured Query Language) injection classification.
- This study's experimental result has been analyzed and evaluated in tabular format and compared with another existing model.

In order to address the above objectives, the following research questions have been developed:

- **RQ1:** How do the detection accuracies of SIDNet-1 and SIDNet-2 compare to traditional machine learning models in SQL injection scenarios?
  **Objective:** The objective of this research question is to evaluate and compare the detection accuracies of two neural network models, SIDNet-1 and SIDNet-2, against traditional machine learning models, specifically in SQL injection scenarios. This comparison aims to ascertain whether the newer neural network approaches offer significant improvements in identifying and mitigating SQL injection attacks compared to conventional methods.
  **RQ2:** How can the architectural features enable SIDNet models to efficiently handle feature extraction and selection compared to handcrafted methods?
  **Objective:** The purpose of this research question is to evaluate the capacity of architectural features within SIDNet models to enhance the efficiency of feature extraction and selection. It aims to compare these automated methods against traditional handcrafted approaches. The study seeks to identify specific architectural components that contribute to improved

performance. Ultimately, the goal is to establish a foundation for developing more robust and efficient automated feature-handling techniques in various applications.

**RQ3:**Which specific strategies in SIDNet models most effectively reduce overfitting, and how do these impact model performance on unseen data?

Objectives: This research question focuses on identifying and evaluating specific strategies within SIDNet models that effectively mitigate overfitting. It aims to analyze the influence of these strategies on the model's performance when applied to unseen data, thereby enhancing generalizability and reliability.

The reminder is about our study organization in capture wise as follow: Section II literature overview with other existed related work, Section III details about our proposed SIDnet methodology and block-diagrams, Section IV contains dataset, model evaluation and model performance presenting by graphical analysis & confusion metrics V discussion of model performance and experimental result on bar plots, comparison with other exiting model. The Section VI explained the summary & conclusion of the proposed work.

## II. RELATED WORK

Over the years, the trends related to SQL (Structured Query Language) injection attacks it was found that 42% of the cyber attacks in this world attempts on public-facing systems are SQL (Structured Query Language)-injection-based. Above 21% of organizations and industries and institutes are still vulnerable to SQL (Structured Query Language) injection threats. In history, the largest attack caused by SQL (Structured Query Language) injection stole more than 1 billion credentials like user IDs and passwords. On the contrary, by these attacks, attackers stole 130 million card details of users. In 2002, a well-populated online fashion web application store known as guess.com was attacked by a major SQL (Structured Query Language) vulnerability, and as a result, attackers stole more than 200,000 users' credit card numbers. Based on this activity, SQL (Structured Query Language) injection attacks have continued to evolve. They used sophisticated methods like obfuscated code and traditional encryption mechanisms to avoid detection [3]. Overall, SQL (Structured Query Language) injection remarks a persistent cyber threat across the globe through defensive technologies implemented on web applications around 20 years [1]. Sun et al. [38] proposed a deep learning-based detection technology for SQL (Structured Query Language) injection identification where they implement enhanced TextCNN and LSTM methods for SQL (Structured Query Language)IA detection. To capture the characteristics of the samples, they utilized a Bidirectional LSTM (Bi-LSTM) network, resulting in improved model performance. Their comparative experimental deep learning-based model SQL (Structured Query Language)IA detection approach reduced both false prediction and false negative rate.

Falor and colleagues [10] propose a deep learning method for detecting SQL (Structured Query Language) injection using a CNN model. Their research reviews various kinds of SQL (Structured Query Language) injection types and description and comparative analysis between their proposed CNN model and other experimental machine learning models. In their study, they presented model performance metrics such as accuracy, precision, recall, and the ROC curve. Alazzawi [2] represent an RNN model based on a deep learning model for the purpose of SQL (Structured Query Language) injection detection. They proposed a novel method using RNN (Recurrent Neural Network) to capture the SQL (Structured Query Language) queries syntax and semantics feature maps that classified the SQL (Structured Query Language) injection. In their research, they introduced models applicable to natural language processing tasks, including query analysis. Luo et al. [18] introduced a CNN-based method for detecting SQL (Structured Query Language) injection attacks, extracting relevant payloads associated with SQL injection from network flow data. Their CNN-based model incorporates high-dimensional features of SQL (Structured Query Language) behavior for performing SQL injection classification. They represented complete experimental results which contain high accuracy model performance rate along with precision, recall rates and also compared to rule-matching-based methods such as ModSecurity. Lu et al. [17] impose a novel research on SQL (Structured Query Language) injection detection model based on CNN. Their proposed mythology is divided into training and classification detection stages. They have done SQL (Structured Query Language) query word vectorization and CNN has been used for model training and identification. Their proposed approach provides a developed research which represent theoretical insights and Cyber security maintainers over database. A systematic literature review on detection of SQL (Structured Query Language) injection has been proposed by Alghawazi et al. [4]. In their study they covers the benefits and application of artificial intelligence and ML models in purpose of control SQL (Structured Query Language) injection on web platform. They have showed a promising result to control SQL (Structured Query Language) injection by detection techniques. Some reviews about SQL (Structured Query Language) injection identification and detection has been proposed by Olalere et al. [25] where they discuss on their study about different types of machine learning and deep learning model along with model performance evaluation. Sharma et al. [36] proposed a comprehensive study on machine learning algorithms where they explore several classifiers as like NLF, LSTM, Naibe Bayes (NB), SVM to invent best classifier among all classifiers for SQL (Structured Query Language) injection detection. Recio-Garcia et al. [30] investigated the interpretability of various machine learning models in the context of detecting SQL injection attacks.

In their study, they investigate the capability and performance various kinds of classifier (Machine Learning Algorithms) like as ADA-Boost, NB, SVM etc. and represent the best performance model to select for SQL (Structured Query Language) injection identification.

In our study, we have proposed our major deep learning model SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2) with extended network model. On the other side we have analyze experimental solution with various types machine learning models such as SVM, KNN, DT (decision Tree) etc., collect the classification report and compared with our proposed customized CNN model SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2).

## III. METHODOLOGY

To enhance SQL injection detection, we propose using our customized CNN model, SIDNet, as a filtration layer following traditional security measures such as firewalls and regex policies. Initially, incoming traffic is filtered through these layers to eliminate obvious threats, reducing the volume of potentially malicious requests. After this initial filtering, the remaining requests are passed to SIDNet, which has been trained on known SQL injection patterns that commonly exploit vulnerabilities using alphanumeric characters. The model analyzes the filtered data to identify more subtle or sophisticated attack vectors that may have bypassed earlier defenses. By employing SIDNet in this layered security approach, we can leverage its deep learning capabilities to effectively classify the remaining inputs as either normal or malicious. This method enhances overall detection accuracy, ensuring a robust defense against evolving SQL injection techniques while minimizing false positives from legitimate traffic. We proposed a two-layer filtration model based on the architecture of CNNs, comprising SIDNet-1 (SQL Injection Attack Detection Network-1) and SIDNet-2 (SQL Injection Attack Detection Network-2). This deep learning model consists of convolutional layers, pooling layers, fully connected layers, and other components. SIDNet-1 (SQL Injection-attack Detection Network-1) consist without dropout where SIDNet-2 (SQL Injection-attack Detection Network-2) consist of dropout to verify the differentiate between both model performance. We incorporated dropout in SIDNet-2 to mitigate overfitting, which can occur when a model learns to memorize training data rather than generalizing from it. By randomly deactivating a subset of neurons during training, dropout promotes more robust feature learning and enhances the model's ability to perform well on unseen data. This approach not only improves generalization but also increases the overall performance of SIDNet-2 in detecting SQL injection attacks. The Table-3 summarizes the key hyperparameters for our proposed customized Convolutional Neural Network (CNN) model SIDNet-1 and SIDNet-2 designed for SQL injection queries classification tasks. Each parameter plays a crucial role in determining the model's performance, such as the number of layers, filter sizes, and learning rate. With Proper tuning of these hyperparameters, it significantly enhance the model's accuracy and generalization capabilities. In Figure-1 shows block-diagram of whole methodology where after WAF (Web Application Firewall), our proposed model could be implement as a custom policy based web proxy or after sanitation from WAF (Web Application Firewall) our proposed model can be used as a Filtration function which classify the rest high level malicious SQL (Structured Query Language) Injection queries which is unable to sanitized by WAF (Web Application Firewall). Two algorithm Algo-1 and Algo-2 implemented in this study which follows the filtration function after WAF (Web Application Firewall) or inside WAF (Web Application Firewall) as a model based sanitation function which is trained by such kind of malicious SQL (Structured Query Language) injection attack patterns.

**TABLE 1. The detail configuration of the proposed SIDNet-1 (SQL Injection-attack Detection Network-1).**

| Layer (Type) | Filter Size | Kernel Size | Strides | Output Shape | Number Of Parameter |
|---|---|---|---|---|---|
| Conv2D | 32 | $3 \times 3$ | - | $62 \times 62 \times 32$ | 320 |
| MaxPooling2D | - | - | $2 \times 2$ | $31 \times 31 \times 32$ | 0 |
| Conv2D (1) | 64 | $3 \times 3$ | - | $29 \times 29 \times 64$ | 18496 |
| MaxPooling2D (1) | - | - | $2 \times 2$ | $14 \times 14 \times 64$ | 0 |
| Conv2D (2) | 128 | $3 \times 3$ | - | $12 \times 12 \times 128$ | 73856 |
| MaxPooling2D (2) | - | - | $2 \times 2$ | $6 \times 6 \times 128$ | 0 |
| Conv2D (3) | 256 | $3 \times 3$ | - | $4 \times 4 \times 256$ | 295168 |
| MaxPooling2D (3) | - | - | $2 \times 2$ | $2 \times 2 \times 256$ | 0 |
| Flatten | - | - | - | 1024 | 0 |
| Dense | - | - | - | 256 | 262400 |
| Dense (1) | - | - | - | 128 | 32896 |
| Dense (2) | - | - | - | 64 | 8256 |
| Dense (3) | - | - | - | 32 | 2080 |
| Dense (4) | - | - | - | 1 | 33 |
| **Total parameter** | | | | | **693505** |
| **Trainable parameter** | | | | | **693505** |
| **Non-trainable parameter** | | | | | **0** |

### A. DATA PREPOSSESSING

In our data preprocessing phase, we commenced by loading a CSV file containing a diverse set of SQL queries, which encompassed both benign and malicious samples indicative of SQL injection attacks. Each query was meticulously transformed into a numerical array, employing techniques such as tokenization and vectorization to facilitate the encoding of textual data. Subsequently, we reshaped the resultant numerical arrays into a three-dimensional format, aligning with the input specifications of our customized CNN architecture. This restructuring not only optimized the data for effective processing but also ensured compatibility with the model's input shape requirements. We have split 80% samples for train and 20% samples for test purpose. By systematically preparing the dataset in this

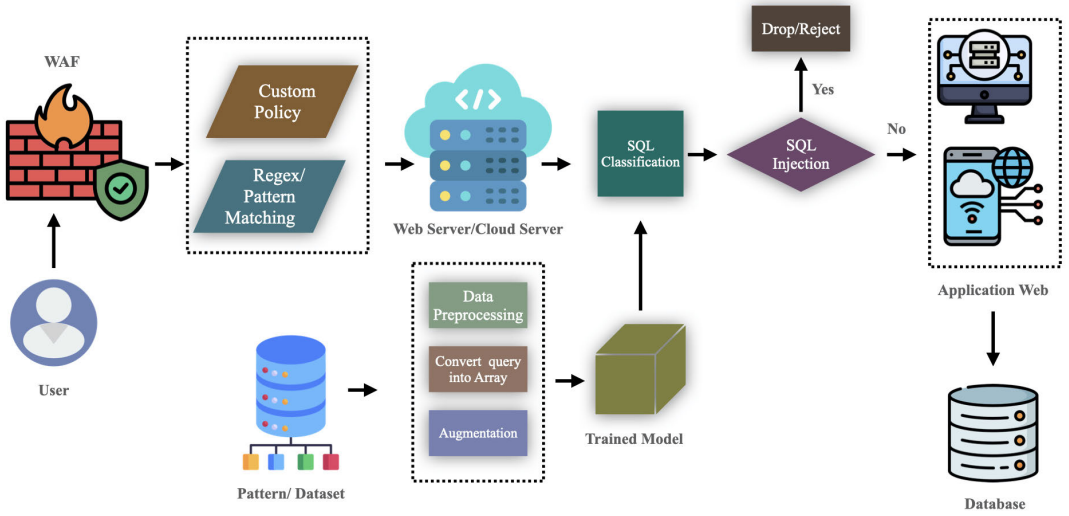



**FIGURE 1.** Proposed block diagram of SQL (Structured Query Language) injection detection techniques.

**TABLE 2.** The detail configuration of the proposed SIDNet-1 (SQL Injection-attack Detection Network-1).

| Layer (Type) | Filter Size | Kernel Size | Strides | Output Shape | Number Of Parameter |
|---|---|---|---|---|---|
| Conv2D | 32 | $3 \times 3$ | - | $62 \times 62 \times 32$ | 320 |
| MaxPooling2D | - | - | $2 \times 2$ | $31 \times 31 \times 32$ | 0 |
| Dropout | - | - | - | $31 \times 31 \times 32$ | 0 |
| Conv2D (1) | 64 | $3 \times 3$ | - | $29 \times 29 \times 64$ | 18496 |
| MaxPooling2D (1) | - | - | $2 \times 2$ | $14 \times 14 \times 64$ | 0 |
| Dropout (1) | - | - | - | $14 \times 14 \times 64$ | 0 |
| Conv2D (2) | 128 | $3 \times 3$ | - | $12 \times 12 \times 128$ | 73856 |
| MaxPooling2D (2) | - | - | $2 \times 2$ | $6 \times 6 \times 128$ | 0 |
| Dropout (2) | - | - | - | $6 \times 6 \times 128$ | 0 |
| Conv2D (3) | 256 | $3 \times 3$ | - | $4 \times 4 \times 256$ | 295168 |
| MaxPooling2D (3) | - | - | $2 \times 2$ | $2 \times 2 \times 256$ | 0 |
| Dropout (2) | - | - | - | $2 \times 2 \times 256$ | 0 |
| Flatten | - | - | - | 1024 | 0 |
| Dense | - | - | - | 256 | 262400 |
| Dense (1) | - | - | - | 128 | 32896 |
| Dense (2) | - | - | - | 64 | 8256 |
| Dense (3) | - | - | - | 32 | 2080 |
| Dense (4) | - | - | - | 1 | 33 |
| **Total parameter** | | | | | **693505** |
| **Trainable parameter** | | | | | **693505** |
| **Non-trainable parameter** | | | | | **0** |

manner, we established a robust foundation for training and evaluating the performance of our SQL injection detection framework.

### B. CONVOLUTIONAL NEURAL NETWORKS

A Convolutional Neural Network (CNN) is a specialised class of deep learning models designed to process data with a grid-like topology [27]. Convolutional Neural Networks (CNNs) have become instrumental in enhancing cybersecurity, particularly in detecting sophisticated threats such as SQL injection attacks, Distributed Denial of Service (DDoS) attacks, and malware. CNNs can analyze network traffic and application logs to identify patterns and anomalies that signify malicious activities [5]. By processing vast amounts of data, CNNs can detect SQL injection attempts by recognizing abnormal query patterns and deviations from typical database interactions. For DDoS attacks, CNNs can distinguish between normal traffic and malicious overload attempts through deep learning-based traffic analysis. In malware detection, CNNs excel by analyzing code structures and behaviors, identifying threats that evade traditional signature-based detection methods. Their ability to continuously learn and adapt to new attack vectors makes CNNs a critical component in proactive cybersecurity strategies. By employing CNNs, organizations can achieve higher accuracy in threat detection and faster response times, thereby significantly enhancing their defensive capabilities against a wide range of cyber threats [16]. This advanced approach not only fortifies the security posture but also mitigates the risk of data breaches and system compromises in an increasingly complex cyber landscape.

#### 1) CONVOLUTIONAL LAYER

Convolutional layers are the cornerstone and important part of CNNs model [40]. The system is designed to autonomously and dynamically learn spatial feature hierarchies from input images. The primary operation performed by the convolutional layer in CNN model. In convolution, a filter (also referred to as a kernel) slides across the input image, extracting local patterns or features [8]. Let consider input image tensor (n) dimension ($\mathbf{V}_h \times \mathbf{V}_w \times \mathbf{V}_c$). After each convolutional layer adding in the proposed model the shape reformed with a new volume represented as ($\mathbf{V}_h{}^{new} \times \mathbf{V}_w{}^{new} \times \mathbf{V}_d{}^{new}$) where the filter size represented as ($\mathbf{F}_h \times \mathbf{F}_w \times \mathbf{V}_c$). The filter size construct with the number of hyper-parameter. ($\mathbf{P}$) denoted as number of zero padding and strides defined

as (**S**). Therefore, in terms of mathematical representation of convolutional layer:

$$\mathbf{V}_h{}^{new} = (\mathbf{V}_h - \mathbf{F}_h + 2 \times \mathbf{P})/\mathbf{S} + 1 \quad (1)$$

$$\mathbf{V}_w{}^{new} = (\mathbf{V}_w - \mathbf{F}_w + 2 \times \mathbf{P})/\mathbf{S} + 1 \quad (2)$$

$$\mathbf{V}_d{}^{new} = \mathbf{F} \times (\mathbf{V}_c/\mathbf{R}_c) \quad (3)$$

#### 2) POOLING LAYER

Pooling layers are used to downsample feature maps by summarizing features within patches of the map [9]. These layers decrease the spatial dimensions (width and height) of the incoming input feature maps from the preceding initialized layer output, while retaining the depth (number of channels) of the input shape [40]. In our proposed SIDNet, we incorporate three max-pooling layers, where the output value for each pooling region corresponds to the maximum value of the input values within that region. Mathematically, given a feature map with dimensions ($\mathbf{M}_h \times \mathbf{M}_w \times \mathbf{M}_c$) where output dimensions after max pooling as follow

$$(\frac{\mathbf{M}_h - f + 1}{s}) \times (\frac{\mathbf{M}_w - f + 1}{s}) \times \mathbf{M}_c \quad (4)$$

where, $\mathbf{M}_h$ represent the feature map, $\mathbf{M}_w$ denoted as width of the feature map, $\mathbf{M}_c$ number of feature channels. The filter size denoted as $f$ and $s$ represent the stride length.

#### 3) FULLY CONNECTED LAYER

Fully connected layers (also known as dense layers) in neural networks [33]. A fully connected layer constitutes a crucial building block within neural networks. In this layer, each neuron (or perceptron) establishes connections with every neuron in both the preceding and subsequent layers [31]. It performs a linear transformation on the input vector using a weights matrix, followed by a non-linear activation function. Let's denote: Input vector: ($x \in \mathbb{R}^m$) (where ($m$) represents the input size). Output of the ($i$)-th neuron: ($y_i \in \mathbb{R}$). Weights associated with the ($i$)-th neuron: ($w_1, w_2, \ldots, w_m$). Activation function: ($\sigma(\cdot)$). The output of the ($i$)-th neuron in the fully connected layer is computed as:

$$y_i = \sigma(w_1x_1 + w_2x_2 + \ldots + w_mx_m) \quad (5)$$

where,the dot product between the weights vector ($\mathbf{w} = [w_1, w_2, \ldots, w_m]$) and the input vector ($\mathbf{x} = [x_1, x_2, \ldots, x_m]$) captures the linear transformation. The activation function ($\sigma(\cdot)$) introduces non-linearity. In the context of Visualization, Let consider a fully connected layer with an input size of (m) and an output size of (n). The operation can be visualized as follows: Each input value ($x_i$) is multiplied by its corresponding weight ($w_i$). The weighted inputs are summed up. The result is passed through the activation function to obtain the output ($y_i$). Example: If we have an input vector of size 9 and an output vector of size 4, the operation can be represented as:

$$y_i = \sigma(w_1x_1 + w_2x_2 + \ldots + w_9x_9) \quad (6)$$

#### 4) SIGMOID

The sigmoid function serves as an activation function within artificial neural networks, introducing non-linearity properties [6]. In the purpose of binary classification and logistic regression problems sigmoid function is very useful because it maps any input value to a range between 0 and 1. The sigmoid function is also known as the logistic function, represent its mathematical function with an S-shaped curve [19]. The functions characteristics is monotonic means the function is monotonically increase or decrease. It has a bell-shaped first derivative. This function is invertible, with its inverse corresponding to the logit function. It exhibits convex behavior for values below a certain threshold and concave behavior for values exceeding that threshold, often centered around 0 [41]. The sigmoid is expressed mathematically as:

$$f(x) = \frac{1}{1 + e^{-x}} \quad (7)$$

**TABLE 3. Hyperparameters for the customized CNN model.**

| Hyperparameter | Value |
|---|---|
| Activation Function | ReLU , Sigmoid |
| Dropout Rate | 0.5 |
| Learning Rate | 0.001 |
| Batch Size | 32 |
| Epochs | 20 |
| Optimizer | Adam |
| Input Image Size | 224 x 224 |
| Weight Initialization | He Initialization |

### C. CUSTOMISED CNNs

The customized CNN models, SIDNet-1 (SQL Injection-attack Detection Network-1) (Figure-2) and SIDNet-2 (SQL Injection-attack Detection Network-2) (Figure-3) are designed to enhance SQL (Structured Query Language) injection detection and prevention. Both models incorporate core convolutional neural networks (CNN) components such as convolution layers, pooling layers, flatten layers and dense layers providing a robust framework for feature extraction and classification. SIDNet-1 (SQL Injection-attack Detection Network-1) follows a straightforward CNN architecture, ensuring efficient processing feature mapping. In contrast, SIDNet-2 (a network designed to detect SQL injection attacks) incorporates a dropout layer, which plays a crucial role in preventing overfitting by randomly deactivating a portion of the neurons during training. The enhancement in SIDNet-2 (SQL Injection-attack Detection Network-2) allows for improved generalization and robustness, ensuring the model performs well on unseen data. Together, these models offer a comprehensive approach to effectively detect and prevent SQL (Structured Query Language) injection attacks with high accuracy and reliability.

#### 1) PROPOSED CNN MODELS

In Figure-2 and Figure-3 two customized CNN models follows the architecture of neural network. Both model has comprehensive layer distribution of neural networks consist

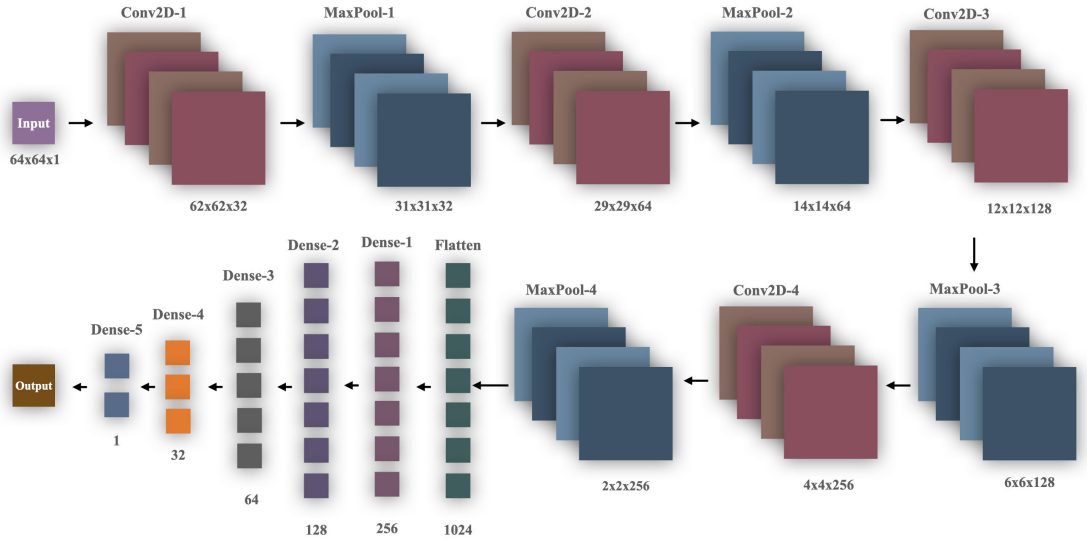



**FIGURE 2.** Proposed SIDNet1 architecture.

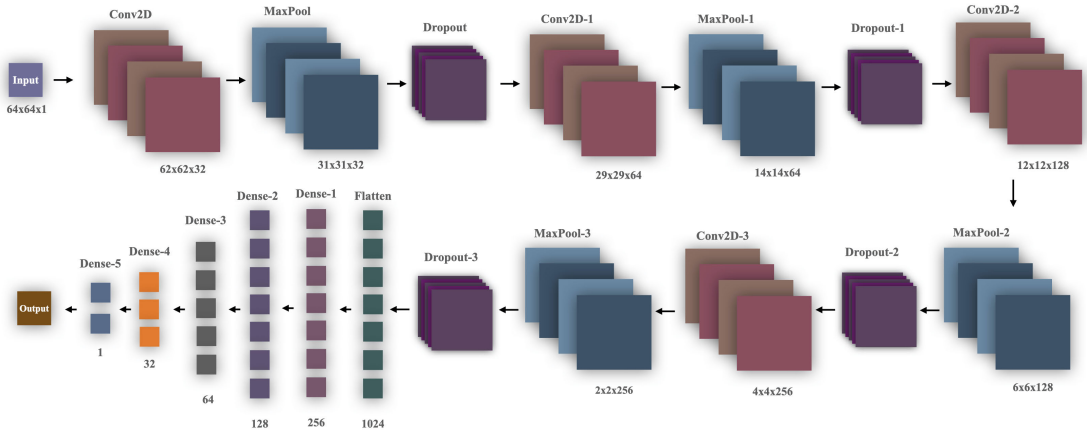



**FIGURE 3.** Proposed SIDNet2 architecture.

of convolutional layer, maxpooling layer, fully connected layer or dense layer with activation functions [40]. Both model description of distributed layer represent in Table-1 and Table-2 respectively. We convert the actual text of both malicious query language and normal language into numerical array and reshaped into 64 × 64 × 1 as a first input shape. In our proposed model SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2), the first convolutional layer-1 fixed with output shape size 62 × 62 × 32 in both with filter size 32 and kernel size of 3 × 3. After first convolutional layer a maxpooling layer-1 added with 2 × 2 strides size. In SIDNet-2 (SQL Injection-attack Detection Network-2) after first convolutional layer and maxpooling a dropout layer-1 set with value.25. After first convolution layer-1 in both model second convolutional layer-2 added with the size of output shape 29 × 29 × 64 with filter size 64 and kernel size of 3 × 3. Maxpooling layer-2 set with strides 2 × 2 in both model. In SIDNet-2 (SQL Injection-attack Detection Network-2) the second dropout layer-2 fixed with.25. We sequentially done the process up to convolutional layer-4 where convolutional layer-3 set with output shape 12 × 12 × 128 with filter size 128 and kernel size 3 × 3 and convolutional layer-4 fixed with output shape $4 \times 4 \times 256$ with filter size of 256 and kernel size of $3 \times 3$. Maxpooling layer-3 added with output shape 6 × 6 × 128 and maxpooling layer-4 with output shape 2 × 2 × 256 with strides size 2 × 2 in both model. But SIDNet-2 (SQL Injection-attack Detection Network-2) has extended with dropout layer with after each convolutional layer and maxpooling layer where dropout layer-3 and layer-4 fixed with value 0.4. The flatten layer is used to transition from the convolution layer and pooling layers to the dense layer or fully connect layer. Finally our proposed model add fully connected layer with dense layer-1 to dense layer-5 with output shape 256, 128, 64, 32, 1 respectively.

### D. LIMITATIONS & STRENGTHS

While our customized CNN model, SIDNet, effectively detects SQL injection attacks by classifying inputs as normal or malicious based on established patterns, it does have limitations. One key issue is its vulnerability to novel attack vectors that deviate from known patterns, potentially leading to missed detections. As cyber threats evolve, new scripting techniques may not be recognized by the model, affecting its overall reliability. However, a significant strength of SIDNet lies in its ability to identify recurring sequences within the malicious patterns. Even as new datasets emerge, many attacks may still exhibit recognizable characteristics, allowing the model to adapt and remain effective. By leveraging these consistent sequences, SIDNet can maintain a strong detection rate, making it a robust tool in the ongoing fight against SQL injection attacks.

**Algorithm 1**

0: **1**. Collect query data.
0: **2**. Convert the SQL (Structured Query Language) query texts into numerical format.
0: **3**. Convert to a Numpy array and reshape to 3D.
0: **4**. Create the SIDNet-1 (SQL Injection-attack Detection Network-1) model with the specified layers configuration in Table-1.
0: **5**. Compile the model with adam optimizer and binary crossentropy loss function.
0: **6**. Fit the model using the training dataset.
0: **7**. Assess the model's effectiveness using the test dataset.
0: **8**. Use the trained model to predict and classify new SQL (Structured Query Language) queries
0: **9**. Plot accuracy and loos curve.
0: **10**. Display confusion metrics
=0

## IV. RESULTS

### A. DATASET

In this study we have used SQL (Structured Query Language) injection detection dataset prepared by capturing both normal and malicious HTTP (Hypertext Transfer Protocol) requests. We have extracted features from the requests to effectively train our proposed SIDNet model. The employed datasets, namely SQLI (Structured Query Language Injection) and SQLIV2 (Structured Query Language Injection Version-2), encompass a diverse range of query patterns, including both typical and anomalous ones. These datasets play a pivotal role in bolstering web application security by precisely detecting and flagging instances of SQL injection attacks. For an example of sample of SQL (Structured Query Language) Injection Attack Using "o=o" – 1. An attacker can exploit the above code by injecting malicious input like this way which follows as Username: o=o && Password: – 1 where the input is concatenated into the query, resulting in SQL (Structured Query Language) as SELECT * FROM users WHERE username = 'o=o' AND password = '– 1'; In

**Algorithm 2**

0: **1**. Collect query data.
0: **2**. Convert the SQL (Structured Query Language) query texts into numerical format.
0: **3**. Convert to a Numpy array and reshape to 3D.
0: **4**. Create the SIDNet-2 (SQL Injection-attack Detection Network-2) model with the specified layers configuration in Table-2.
0: **5**. Compile the model with adam optimizer and binary crossentropy loss function.
0: **6**. Fit the model using the training dataset.
0: **7**. Assess the model's effectiveness using the test dataset.
0: **8**. Use the trained model to predict and classify new SQL (Structured Query Language) queries
0: **9**. Plot accuracy and loos curve.
0: **10**. Display confusion metrics
=0

**TABLE 4. Number of sample dataset distribution with train test split.**

| Dataset | SQL Inj. Set | NoSQL Inj. Set | Total Sample | Train Set | Test Set |
|---|---|---|---|---|---|
| SQLI | 1128 | 3071 | 4200 | 3360 | 40 |
| SQLIV2 | 11456 | 22305 | 33761 | 27009 | 6752 |

the context of Query Execution, the – sequence in SQL (Structured Query Language) is used to comment out the rest of the query. The query effectively becomes as SELECT * FROM users WHERE username = 'o=o'; – 1;. This changes the logic of the query. Instead of checking the username and password, it now checks if o=o (which is always true) and ignores the rest of the query. The database returns all rows where the condition is true, potentially allowing the attacker to bypass authentication. The multiple dataset we have used that are publicly available in kaggle platform https://www.kaggle.com/datasets/syedsaqlainhussain/SQL (Structured Query Language)-injection-dataset?select=SQL (Structured Query Language)iv2.csv.

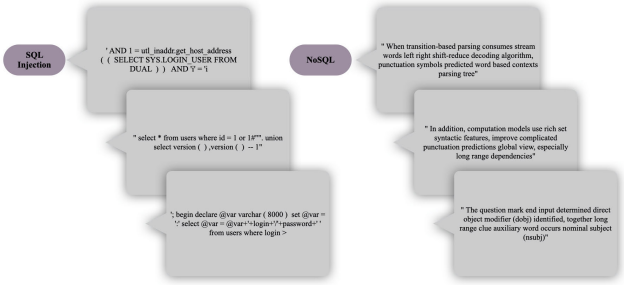



**FIGURE 4. Data Sample of SQLI dataset.**

### B. PROPOSED MODEL EVALUATION

We assess the effectiveness of our custom CNN models, SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2),

in detecting SQL injection attacks. To achieve this, we utilize the confusion matrix, which presents a comprehensive overview of the model's performance by detailing the counts of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). In Figure-5(a) and 5(b) for **SQLI** dataset and 6(a) and 6(b) for **SQLIV2** dataset shows the performance of SQL (Structured Query Language) injection detection with calssification values of True label and Predicted label of our proposed model SIDNet-1 (SQL Injection-attack Detection Network-1) & SIDNet-2 (SQL Injection-attack Detection Network-2).

Using the values from the confusion matrix, we can derive several key performance metrics:

Accuracy: The accuracy ratio, which considers both true positives and true negatives, is calculated by dividing the correct predictions by the total number of instances.

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN} \tag{8}$$

Precision: The precision, which represents the proportion of true positives among all predicted positive instances, serves as an indicator of the accuracy of positive predictions.

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

Recall (Sensitivity): The recall, which represents the proportion of true positives among all actual positive instances, reflects the model's capability to detect positive cases.

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

F1 Score: The F1 score, which harmonizes precision and recall, strikes a balance between these two performance metrics.

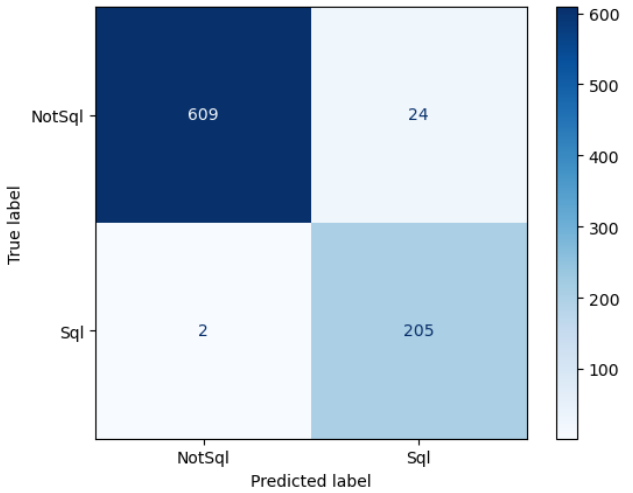
$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{11}$$

FAR:The False Acceptance Rate (FAR) is a critical metric in evaluating the performance of bio-metric systems and classification models.

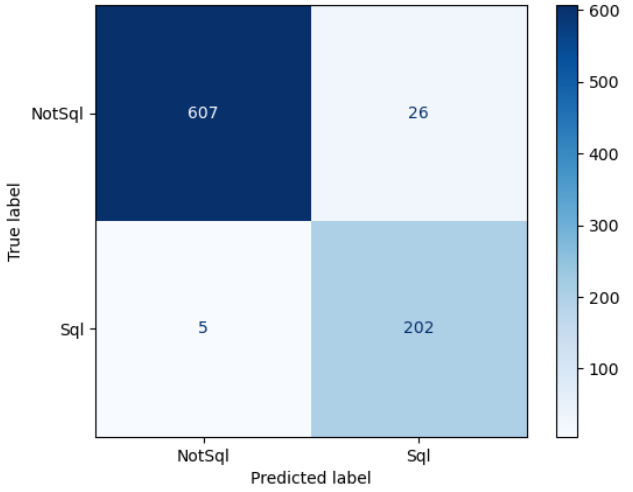
$$FAR = \frac{FA}{N} \tag{12}$$

### C. MODEL PERFORMANCE

We acknowledge potential validity threats to the detection of SQL injection attacks using customized CNNs. To effectively evaluate the performance of our SQL (Structured Query Language) injection detection model, we utilize various graphical representations. Internally, the dataset's representativeness may influence model performance, as it primarily reflects specific attack patterns and may not generalize to all real-world scenarios. Externally, the evolving nature of SQL injection techniques poses a challenge; as attackers adapt, our model may become less effective if not regularly updated. These visual tools help in understanding the model's accuracy, precision, recall, and overall effectiveness in detecting SQL (Structured Query Language) injection



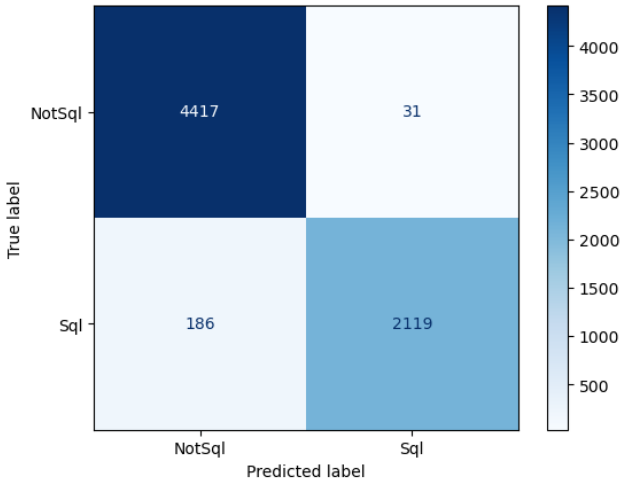

((a)) True and False prediction Confusion metrics of our proposed SIDNet-1 (SQL Injection-attack Detection Network-1)



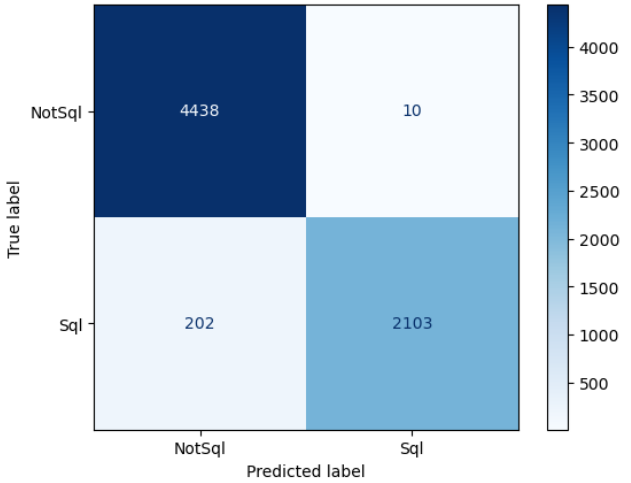

((b)) True and False prediction Confusion metrics of our proposed SIDNet-2 (SQL Injection-attack Detection Network-2)

**FIGURE 5. Confusion metrics model evaluation of our proposed customized SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2) over SQLI dataset.**

attacks. The accuracy and loss curves provide into the model's learning process over time. By plotting these metrics for both training and validation datasets, we have observed and analysed how well the model generalizes to unseen data. In Figure-7(a) (SIDNet-1 (SQL Injection-attack Detection Network-1)) and Figure-8(a) (SIDNet-2 (SQL Injection-attack Detection Network-2)) for **SQLI** dataset and Figure-9(a) (SIDNet-1 (SQL Injection-attack Detection Network-1)) and Figure-10(a) (SIDNet-2 (SQL Injection-attack Detection Network-2)) for **SQLIV2** dataset, the graph shows the proportion of correctly identified SQL (Structured Query Language) injection over the total number of instances. The flows steady increase in accuracy indicates that the model is learning effectively. In Figure-7(b) (SIDNet-1 (SQL Injection-attack Detection Network-1)) and Figure-8(b) (SIDNet-2 (SQL Injection-attack Detection Network-2)) for **SQLI** dataset and Figure-9(b) (SIDNet-1

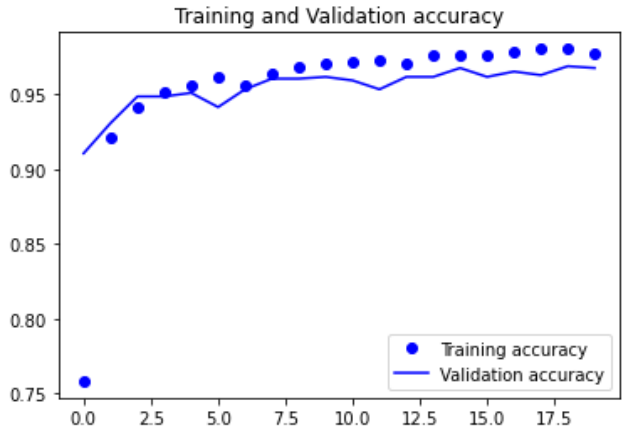



((a)) True and False prediction Confusion metrics of our proposed SIDNet-1 (SQL Injection-attack Detection Network-1)

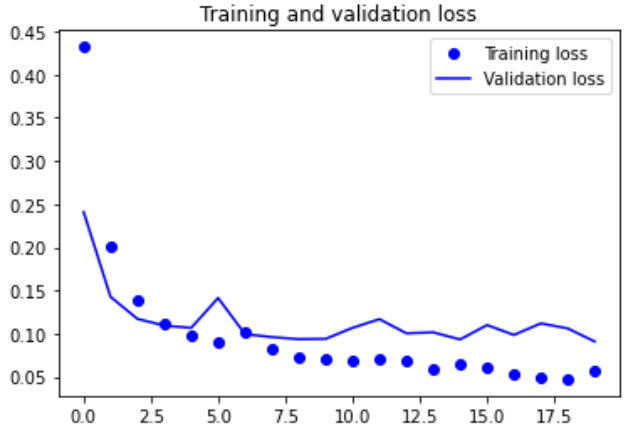



((b)) True and False prediction Confusion metrics of our proposed SIDNet-2 (SQL Injection-attack Detection Network-2)

**FIGURE 6.** Confusion metrics model evaluation of our proposed customized SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2) over SQLIV2 dataset.




((a)) Train and Validation Accuracy




((b)) Train and Validation Accuracy

**FIGURE 7.** Accuracy and loss performance of our proposed customized SIDNet-1 (SQL Injection-attack Detection Network-1) over SQLI dataset where X-axis represent Number of Epocs and Y-axis represent Number of Percentage.

(SQL Injection-attack Detection Network-1)) and Figure-10(b) (SIDNet-2 (SQL Injection-attack Detection Network-2)) for **SQLIV2** dataset graphs represents the error in the model's predictions. The decreasing loss curve suggests that the model is improving its predictions over time.

### D. COMPARATIVE ANALYSIS

In this study, we have shown an comprehensive comparative performance, evaluating accuracy. precision, recall F1-score with other relevant metrics using our appropriate datasets. In Figure-11 and Figure-12 We have demonstrate bar charts of each machine learning models comparatively with our proposed CNN model SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2) individually on classification performance and sentiment analysis. In Table-5 and Table-6 we have shown the comparetive analysis over performance metrics which represent the differences in values between various ML models with our proposed customized CNN models.

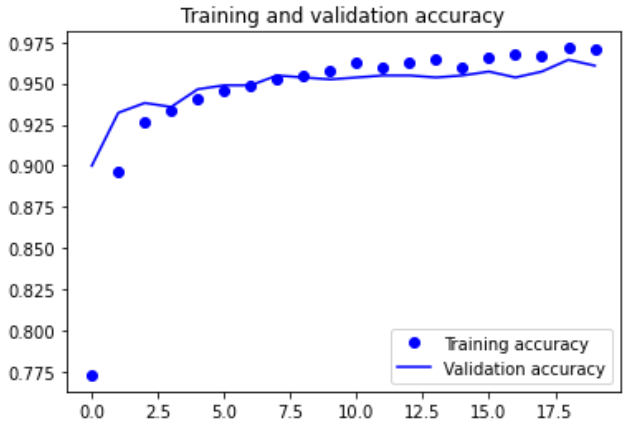
**TABLE 5.** Comparetive analisis between ML models and proposed CNN models over SQLI dataset.

| Classifier Used | Acc(%) | FAR (%) | Rec(%) | Pre(%) | F1-S(%) |
|---|---|---|---|---|---|
| **NB** | 97.61 | 100 | 100 | 91.18 | 95.39 |
| **DT** | 84.64 | 100 | 100 | 61.60 | 76.24 |
| **KNN** | 44.88 | 100 | 100 | 30.89 | 47.20 |
| **SVM** | 80.71 | 0 | 21.72 | 100 | 35.71 |
| **SIDNet-1** | 98.02 | 96.7 | 99.51 | 90.56 | 94.27 |
| **SIDNet-2** | 97.54 | 96.55 | 99.51 | 89.03 | 93.42 |

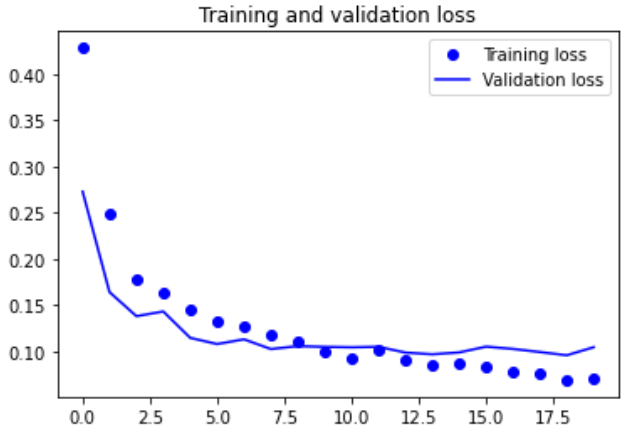
NB: Naïve Bayes; DT: Decision Tree; KNN: K-Nearest Neighbor; SVM: Support Vector Machine; Acc: Accuracy; FAR: False Acceptance Rate; Rec: Recall; Pre: Precision; F1-S: F1-Score

## V. DISCUSSION

The use of convolutional neural network (CNNs) for SQL (Structured Query Language) injection detection and prevention is an innovative approach which represent leverages the capabilities of deep learning to enhance cybersecurity. The discussion explores the design, implementation,

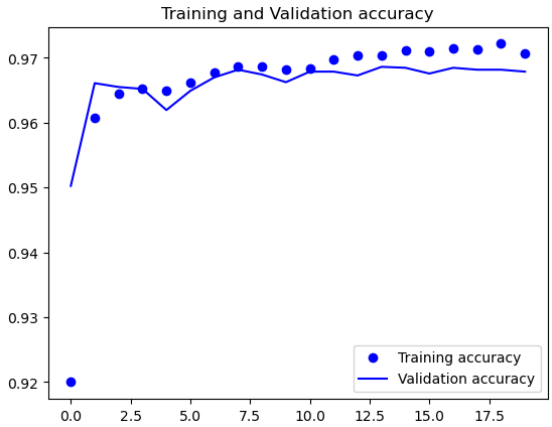



((a)) Train and Validation Accuracy

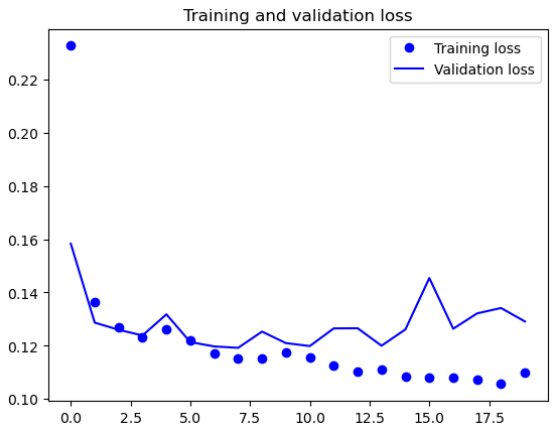



((b)) Train and Validation Accuracy

**FIGURE 8.** Accuracy and loss performance of our proposed customized SIDNet-2 (SQL Injection-attack Detection Network-2) over SQLI dataset where X-axis represent Number of Epocs and Y-axis represent Number of Percentage.




((a)) Train and Validation Accuracy




((b)) Train and Validation Accuracy

**FIGURE 9.** Accuracy and loss performance of our proposed customized SIDNet-1 (SQL Injection-attack Detection Network-1) over SQLIV2 dataset where X-axis represent Number of Epocs and Y-axis represent Number of Percentage.

**TABLE 6.** Comparetive analisis between ML models and proposed CNN models over SQLIV2 dataset.

| Classifier Used | Acc(%) | FAR (%) | Rec(%) | Pre(%) | F1-S(%) |
|---|---|---|---|---|---|
| **NB** | 54.44 | 96.60 | 99.47 | 42.80 | 59.85 |
| **DT** | 95.98 | 14.02 | 89.89 | 98.19 | 93.86 |
| **KNN** | 96.13 | 4.98 | 89.24 | 99.37 | 94.03 |
| **SVM** | 88.83 | 0.0 | 67.28 | 100 | 80.44 |
| **SIDNet-1** | 97.77 | 14.67 | 91.93 | 98.51 | 95.10 |
| **SIDNet-2** | 97.83 | 8.41 | 91.49 | 99.15 | 95.17 |

NB: Naïve Bayes; DT: Decision Tree; KNN: K-Nearest Neighbor; SVM: Support Vector Machine; Acc: Accuracy; FAR: False Acceptance Rate; Rec: Recall; Pre: Precision; F1-S: F1-Score

**TABLE 7.** Performance comparison between existing models and the proposed models.

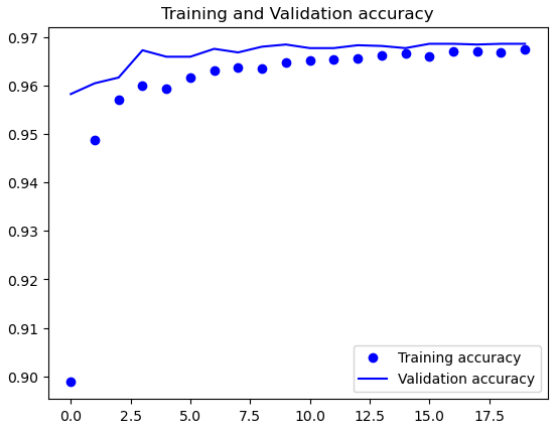
| Existed Model | Classifiers Used | Acc(%) | Rec(%) | Pre(%) | F1-S(%) |
|---|---|---|---|---|---|
| Alghawazi et al. [2] | RNN | 94.40 | 95.9 | 90.6 | 92.58 |
| Poul et al. [29] | CNN-LSTM | 97.23 | 97.66 | 95.11 | 96.30 |
| Shabaz et al. [34] | CNN | 97.41 | 96.50 | 99.00 | 97.00 |
| Yegnidemir et al. [43] | CNN | 96.43 | 98.41 | 91.84 | - |
| **Proposed Model** | **SIDNet-1** | 98.02 | 99.51 | 90.56 | 94.27 |
| **Proposed Model** | **SIDNet-2** | 97.54 | 99.51 | 89.03 | 93.42 |

Acc: Accuracy; FAR: False Acceptance Rate; Rec: Recall; Pre: Precision; F1-S: F1-Score

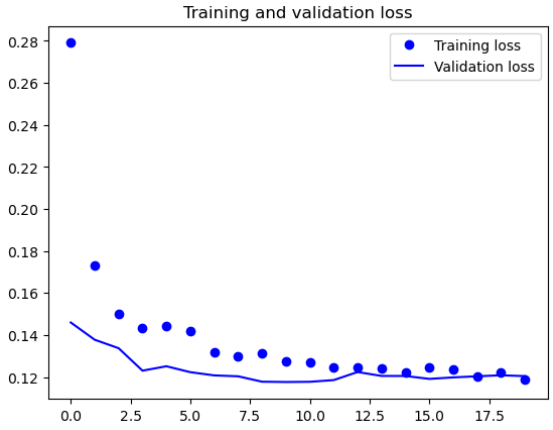
performance graphical representation and model evaluation of two customized CNN models, SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2), tailored for SQL (Structured Query Language) injection detection and prevention.

### A. OVERVIEW OF SQL INJECTION ATTACKS

SQL (Structured Query Language) injection is a prevalent and dangerous type of attack where malicious SQL (Structured Query Language) code is inserted into a query through input fields, allowing attackers to manipulate databases. Traditional detection methods rely on signature-based or heuristic approach, which can be limited by the evolving nature of SQL (Structured Query Language) injection techniques. Therefore, we deployed and completed our innovative experimental reseach on dynamic and adaptive approach, such as deep learning and machine learning.




((a)) Train and Validation Accuracy




((b)) Train and Validation Accuracy

**FIGURE 10.** Accuracy and loss performance of our proposed customized SIDNet-2 (SQL Injection-attack Detection Network-2) over SQLIV2 dataset where X-axis represent Number of Epocs and Y-axis represent Number of Percentage.

### B. DESIGN OF PROPOSED CUSTOMISED CNN MODELS

The SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2) models are designed with the unique characteristics of SQL (Structured Query Language) queries in mind. The SIDNet-1 (SQL Injection-attack Detection Network-1), designed for detecting SQL injection attacks, employed a complex CNN architecture comprising several convolutional layers, followed by max-pooling and dense layers. The focus is on capturing local patterns within the input SQL (Structured Query Language) strings, which helps in identifying anomalous that may indicate an injection attempt. On the contrary SIDNet-2 (SQL Injection-attack Detection Network-2) builds upon SIDNet-1 (SQL Injection-attack Detection Network-1) with a more complex architecture, including additional dropout layer after each convolutonal layer and Maxpooling layer to deactivating a portion of input units during each training updates and also capture deeper and more including patterns. This model aims to improve detection accuracy by considering more extensive contextual information within the SQL (Structured Query Language) queries.

**TABLE 8.** Summary of the research questions.

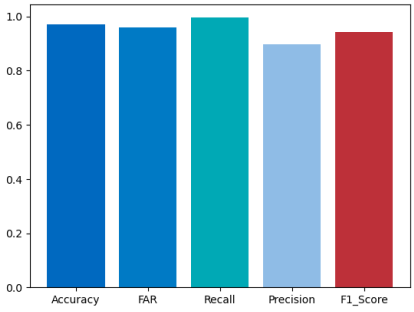
| Research questions | Responses |
|---|---|
| How do the detection accuracies of SIDNet-1 and SIDNet-2 compare to traditional machine learning models in SQL injection scenarios? | The detection accuracies of SIDNet-1 and SIDNet-2 surpass traditional machine learning models in SQL injection scenarios, demonstrating superior performance across various metrics including False Alarm Rate (FAR), recall, precision, and F1 score, thereby enhancing the robustness and reliability of intrusion detection systems. The Table-5 and Table-6 exhibit those superior detection metrics in SQL injection scenarios compared to traditional machine learning models. This substantial enhancement underscores the efficacy of our advanced CNN architecture, surpassing the conventional algorithms by leveraging deep learning's profound capacity for intricate pattern recognition and anomaly detection in complex datasets. |
| How can the architectural features enable SIDNet models to efficiently handle feature extraction and selection compared to handcrafted methods? | The proposed SIDNet architecture, predicated on a Convolutional Neural Network (CNN), diverges from manualized methodologies. The architectural features of SIDNet models enable superior feature extraction and selection by autonomously discerning intricate patterns within numerical arrays of malicious SQL queries. This automated process, facilitated by deep learning's hierarchical structure, surpasses the limitations of handcrafted methods, culminating in an impressive 98.03% accuracy. This efficacy underscores the profound capabilities of SIDNet in managing complex, high-dimensional data with minimal manual intervention |
| Which specific strategies in SIDNet models most effectively reduce overfitting, and how do these impact model performance on unseen data? | SIDNet models mitigate overfitting through strategies such as data augmentation for balaning dataset, dropout regularization and batch normalization. These techniques ensure robust generalization by preventing over-reliance on training data, thereby enhancing model performance on unseen data. The high test accuracies of 98.03%, 97.64%, 97.77%, and 97.83% validate the efficacy of these strategies, demonstrating their pivotal role in maintaining model reliability and accuracy. |

### C. DATA COLLECTION AND TEXT PREPROCESSING

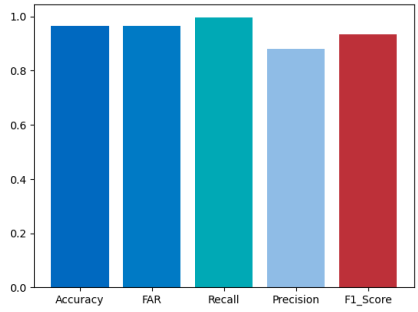
Effective data preparation is crucial for training CNN models where in this study a diverse dataset contain both benign and malicious SQL (Structured Query Language) queries is essential. The dataset should cover various types of SQL (Structured Query Language) injection techniques, including tautologies, union queries, piggybacked queries and more. In the context of text preprocessing steps SQL (Structured Query Language) queries are tokenized into meaning units. Tokenized queries are converted into numerical representations, such as numpy array which are suppored by Tensorflow module and resized the array size with suitable for our proposed customised CNN input.

### D. PREVENTION STRATEGIES

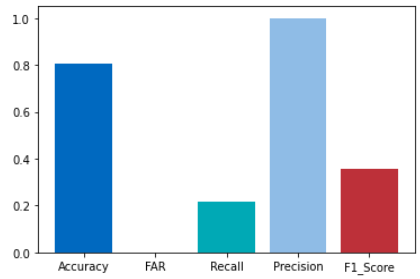
In the context of Integration with Web Applications, our proposed models can be integrated into web application firewalls (WAF (Web Application Firewall)s) or database management systems to provide real-time protection against SQL (Structured Query Language) injections. For Continuous Learning to adapt to evolving threats, the proposed



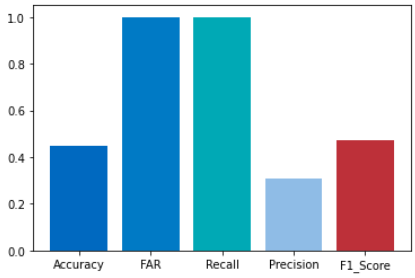

((a)) Bar plot analysis of SIDNet-1 model performance metrics.



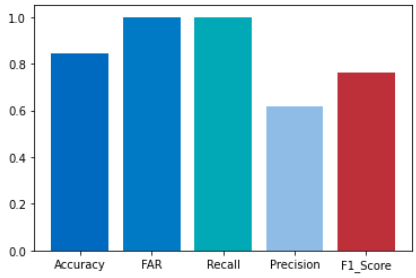

((b)) Bar plot analysis of SIDNet-2 model performance metrics.



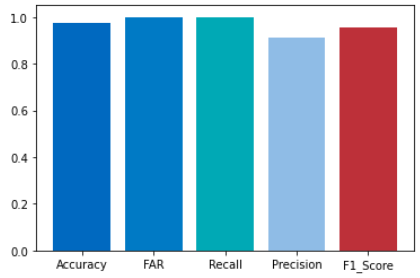

((c)) Bar plot analysis of SVM model performance metrics.




((d)) Bar plot analysis of KNN model performance metrics.

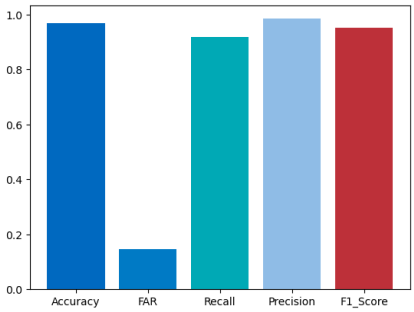



((e)) Bar plot analysis of DT model performance metrics.

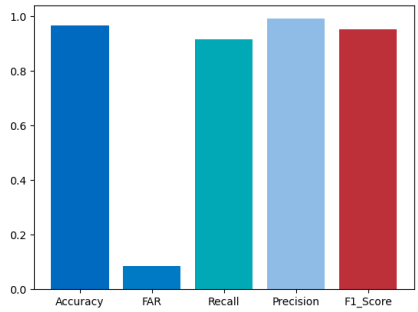



((f)) Bar plot analysis of NB model performance metrics.

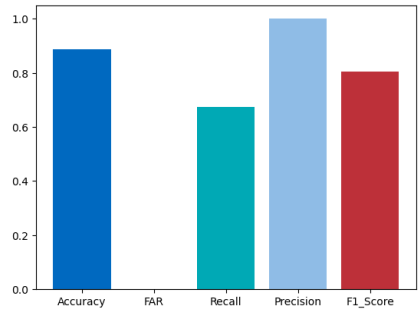
**FIGURE 11.** **Bar plot of hybrid analysis, comparison between our proposed customized CNN model SIDNet-1 (SQL Injection-attack Detection Network-1) 11(a) and SIDNet-2 (SQL Injection-attack Detection Network-2) 11(b) with other machine learning model such as Support Vector Machine (SVM) 11(c), K-Nearest Neighbors (KNN) 11(d), Decision Tree (DT) 11(e), Naive Bayes (NB) 11(f) performance metrics over SQLI dataset.**



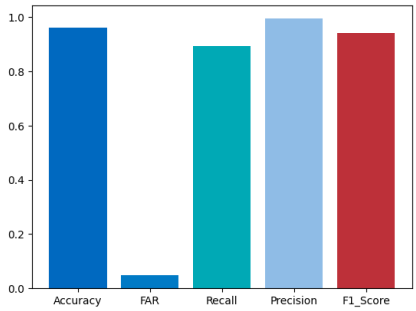

((a)) Bar plot analysis of SIDNet-1 model performance metrics.



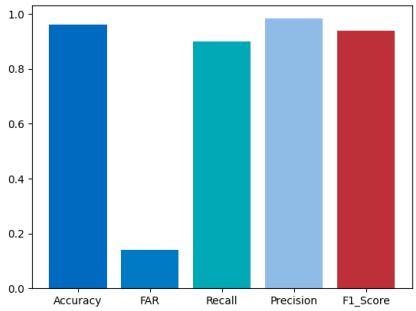

((b)) Bar plot analysis of SIDNet-2 model performance metrics.



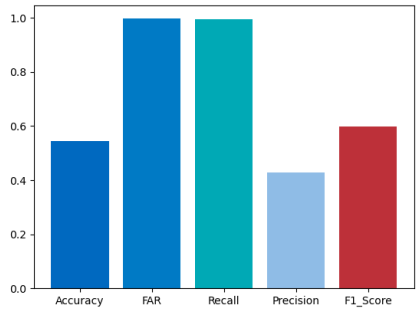

((c)) Bar plot analysis of SVM model performance metrics.




((d)) Bar plot analysis of KNN model performance metrics.




((e)) Bar plot analysis of DT model performance metrics.




((f)) Bar plot analysis of NB model performance metrics.

**FIGURE 12.** **Bar plot of hybrid analysis, comparison between our proposed customized CNN model SIDNet-1 (SQL Injection-attack Detection Network-1) 12(a) and SIDNet-2 (SQL Injection-attack Detection Network-2) 12(b) with other machine learning model such as Support Vector Machine (SVM) 12(c), K-Nearest Neighbors (KNN) 12(d), Decision Tree (DT) 12(e), Naive Bayes (NB) 12(f) performance metrics over SQLIV2 dataset.**

models can be periodically restrained with new data. This continuous learning approach ensures that the models remain effective against emerging SQL (Structured Query Language) injection techniques. The CNN-based models should be part of a broader security strategy that includes input validation, parameterized queries, and regular security audits to provide

comprehensive protection against SQL (Structured Query Language) injection attacks.

### E. COMPARISON WITH EXISTED MODEL

This subsection contains the comparison results of the proposed models, SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2), with existing SQL (Structured Query Language) injection detection models. The comparison includes key performance metrics accuracy, recall, precision and F1-score. Several proposed models based on RNN Autoencoder [2], hybrid CNN-LSTM model [29], Convolutional Neural Networks, one of the deep learning techniques [34], Convolutional Neural Network (CNN) [43] represent their own model with novelty with a well performance of accuracy, where our proposed model SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2) also shows a reputed performance following the top-performing model and the comparison results shown in Table-7.

### F. DISCUSSION & PRACTICAL APPLICATION

The deployment of our customized model, SIDNet, offers a pragmatic solution for enhancing the security posture of web applications against SQL injection attacks. This model can be strategically positioned after cloud servers or web servers as a custom policy layer, thereby providing an additional line of defense that operates in conjunction with existing security measures. Furthermore, SIDNet can be effectively integrated post-firewall and alongside regex-based policies, creating a multi-tiered filtration system. This dual-layered approach allows for the preliminary screening of traffic by the firewall, followed by a more nuanced analysis using SIDNet, which has been trained on comprehensive datasets of SQL injection patterns. By leveraging deep learning techniques, SIDNet not only identifies known attack vectors but also adapts to emerging threats through its advanced pattern recognition capabilities. This integration ensures a robust, adaptive defense mechanism, safeguarding web applications while minimizing false positives and preserving legitimate user interactions. Such deployment considerations underscore the model's versatility and efficacy in real-world applications, providing organizations with a fortified framework against evolving cyber threats.

## VI. CONCLUSION AND FUTURE WORK

The development and implementation of customized convolutional neural networks, SIDNet-1 (SQL Injection-attack Detection Network-1) and SIDNet-2 (SQL Injection-attack Detection Network-2), for SQL (Structured Query Language) injection detection and prevention mark significant advancements in cybersecurity. These models leverage the power of deep learning to identify and mitigate one of the most common and dangerous web application vulnerabilities. SIDNet-1, with its straightforward architecture, provides a solid foundation for detecting SQL injection attacks by capturing local patterns within SQL queries. SIDNet-2, with its more complex design, builds on this foundation to offer improved accuracy and generalization by capturing deeper and more intricate patterns in the data. Future research could focus on further optimizing these models, exploring hybrid approaches, and applying the principles of SIDNet to other types of cybersecurity threats. In conclusion, SIDNet-1 and SIDNet-2 represent promising approaches to SQL injection detection and prevention, leveraging the capabilities of convolutional neural networks to provide a dynamic, adaptive, and effective security solution for web applications. This study contributes to potential future directions that could further optimize the SIDNet models to enhance their efficiency and accuracy, making them more suitable for deployment in various environments. Combining CNNs with other machine learning techniques, such as recurrent neural networks (RNNs) or attention mechanisms, could improve detection capabilities. Exploring the application of these models to other types of cybersecurity threats, such as cross-site scripting (XSS) or command injection, could provide a more holistic security solution.

## ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA for funding this research work through the project number "NBU-FFR-2024-1850-02".

## REFERENCES

[1] V. Abdullayev and D. A. S. Chauhan, "SQL injection attack: Quick view," *Mesopotamian J. Cyber Secur.*, vol. 2, pp. 30–34, Feb. 2023.

[2] A. Alazzawi, "SQL injection detection using RNN deep learning model," *J. Appl. Eng. Technological Sci. (JAETS)*, vol. 5, no. 1, pp. 531–541, Dec. 2023.

[3] M. Alenezi, M. Nadeem, and R. Asif, "SQL injection attacks countermeasures assessments," *Indonesian J. Electr. Eng. Comput. Sci.*, vol. 21, no. 2, p. 1121, Feb. 2021.

[4] M. Alghawazi, D. Alghazzawi, and S. Alarifi, "Detection of SQL injection attack using machine learning techniques: A systematic literature review," *J. Cybersecurity Privacy*, vol. 2, no. 4, pp. 764–777, Sep. 2022.

[5] H. C. Altunay and Z. Albayrak, "Network intrusion detection approach based on convolutional neural network," *Eur. J. Sci. Technol.*, vol. 2, no. 26, pp. 22–29, Jun. 2021.

[6] G. A. Anastassiou, "General sigmoid based Banach space valued neural network approximation," *J. Comput. Anal. Appl*, vol. 31, no. 4, pp. 520–534, 2023.

[7] D. Appelt, C. D. Nguyen, and L. Briand, "Behind an application firewall, are we safe from SQL injection attacks?" in *Proc. IEEE 8th Int. Conf. Softw. Test., Verification Validation (ICST)*, Apr. 2015, pp. 1–10.

[8] J. Bharadiya, "Convolutional neural networks for image classification," *Int. J. Innov. Sci. Res. Technol.*, vol. 8, no. 5, pp. 673–677, 2023.

[9] F. M. Bianchi and Veronica Lachi, "The expressive power of pooling in graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, pp. 1–12.

[10] A. Falor, M. Hirani, H. Vedant, P. Mehta, and D. Krishnan, "A deep learning approach for detection of SQL injection attacks using convolutional neural networks," in *Data Analytics and Management*. Cham, Switzerland: Springer, 2022, pp. 293–304.

[11] E. Frank, A. Luz, and H. Jonathan, "Access control and authentication mechanisms in cloud databases," 2024.

[12] B. Gunjal and M. M. Koganurmath, "Database system: Concepts and design," in *Proc. 24th IASLIC-SIG-2003*, 2003, pp. 1–14.

[13] S. Hajar, A. G. Jaafar, and F. Abdul Rahim, "A review of penetration testing process for SQL injection attack," *Open Int. J. Informat.*, vol. 12, no. 1, pp. 221–236, Jun. 2024.
[14] W. G. J. Halfond, J. Viegas, and A. Orso, "A classification of SQL injection attacks and countermeasures," in *Proc. ISSSE*, 2006, pp. 1–10.
[15] J. L. Harrington, *Relational Database Design and Implementation*. San Mateo, CA, USA: Morgan Kaufmann, 2016.
[16] M. Kravchik and A. Shabtai, "Detecting cyber attacks in industrial control systems using convolutional neural networks," in *Proc. Workshop Cyber-Phys. Syst. Secur. Privacy*, Jan. 2018, pp. 72–83.
[17] R. Lu, S. Wang, and Y. Li, "Research on SQL injection detection model based on CNN," in *Proc. Int. Conf. Intell. Comput., Autom. Appl. (ICAA)*, Jun. 2021, pp. 111–114.
[18] A. Luo, W. Huang, and W. Fan, "A CNN-based approach to the detection of SQL injection attacks," in *Proc. IEEE/ACIS 18th Int. Conf. Comput. Inf. Sci. (ICIS)*, Jun. 2019, pp. 320–324.
[19] F. Makhrus, "The effect of amplitude modification in S-shaped activation functions on neural network regression," *Neural Netw. World*, vol. 33, no. 4, pp. 245–269, 2023.
[20] D. Muduli, R. Dash, and B. Majhi, "Automated breast cancer detection in digital mammograms: A moth flame optimization based ELM approach," *Biomed. Signal Process. Control*, vol. 59, May 2020, Art. no. 101912.
[21] D. Muduli, R. Dash, and B. Majhi, "Enhancement of deep learning in image classification performance using VGG16 with swish activation function for breast cancer detection," in *Proc. Int. Conf. Comput. Vis. Image Process.*, 2021, pp. 191–199.
[22] D. Muduli, R. Dash, and B. Majhi, "Fast discrete curvelet transform and modified PSO based improved evolutionary extreme learning machine for breast cancer detection," *Biomed. Signal Process. Control*, vol. 70, Sep. 2021, Art. no. 102919.
[23] D. Muduli, R. Dash, and B. Majhi, "Automated diagnosis of breast cancer using multi-modal datasets: A deep convolution neural network based approach," *Biomed. Signal Process. Control*, vol. 71, Jan. 2022, Art. no. 102825.
[24] M. Nasereddin, A. ALKhamaiseh, M. Qasaimeh, and R. Al-Qassas, "A systematic review of detection and prevention techniques of SQL injection attacks," *Inf. Secur. J., A Global Perspective*, vol. 32, no. 4, pp. 252–265, Jul. 2023.
[25] M. Olalere, R. A. Egigogo, R. Umar, and S. M. Abdulhamid, "A systematic literature review on detection, prevention and classification with machine learning approach," Tech. Rep., 2018.
[26] H. Omotunde and M. Ahmed, "A comprehensive review of security measures in database systems: Assessing authentication, access control, and beyond," *Mesopotamian J. Cyber Secur.*, vol. 2, pp. 115–133, Aug. 2023.
[27] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015, *arXiv:1511.08458*.
[28] A. Osmani, *Developing Backbone. Js Applications: Building Better JavaScript Application*. Sebastopol, CA, USA: O'Reilly Media, Inc, 2013.
[29] A. Paul, V. Sharma, and O. Olukoya, "SQL injection attack: Detection, prioritization & prevention," *J. Inf. Secur. Appl.*, vol. 85, Sep. 2024, Art. no. 103871.
[30] J. A. Recio-Garcia, M. G. Orozco-del Castillo, and J. A. Soladrero, "Case-based explanation of classification models for the detection of SQL injection attacks," in *Proc. XCBR*, 2023, pp. 1–23.
[31] F. Ren, X. Wang, Y. Li, and Z. Zeng, "Fully connected neural network-based fixed-time adaptive sliding mode control for fuzzy semi-Markov system," *IEEE Trans. Ind. Informat.*, vol. 20, no. 10, pp. 12317–12327, Oct. 2024.
[32] A. Sadeghian, M. Zamani, and S. M. Abdullah, "A taxonomy of SQL injection attacks," in *Proc. Int. Conf. Informat. Creative Multimedia*, Sep. 2013, pp. 269–273.
[33] L. F. S. Scabini and O. M. Bruno, "Structure and performance of fully connected neural networks: Emerging complex network properties," *Phys. A, Stat. Mech. Appl.*, vol. 615, Apr. 2023, Art. no. 128585.
[34] M. Shahbaz, G. Mumtaz, S. Zubair, and M. Rehman, "Evaluating CNN effectiveness in SQL injection attack detection," *J. Comput. Biomed. Informat.*, vol. 7, no. 2, pp. 1–33, 2024.
[35] S. K. Sharma, D. Muduli, R. Priyadarshini, R. R. Kumar, A. Kumar, and J. Pradhan, "An evolutionary supply chain management service model based on deep learning features for automated glaucoma detection using fundus images," *Eng. Appl. Artif. Intell.*, vol. 128, Feb. 2024, Art. no. 107449.
[36] V. Sharma and S. Kumar, "Comparative study of machine learning algorithms for prediction of SQL injections," in *Algorithms for Intelligent Systems*. Cham, Switzerland: Springer, 2023, pp. 455–466.
[37] R. F. Sidik, S. N. Yutia, and R. Z. Fathiyana, "The effectiveness of parameterized queries in preventing SQL injection attacks at go," in *Proc. Int. Conf. Enterprise Ind. Syst.*, 2023, p. 204.
[38] H. Sun, Y. Du, and Q. Li, "Deep learning-based detection technology for SQL injection research and implementation," *Appl. Sci.*, vol. 13, no. 16, p. 9466, Aug. 2023.
[39] S. T. Echeverri, "Evaluation of sql injection (SQLi) attack detection strategies in web applications using machine learning," Universidad de Antioquia, 2024.
[40] M. M. Taye, "Theoretical understanding of convolutional neural network: Concepts, architectures, applications, future directions," *Computation*, vol. 11, no. 3, p. 52, Mar. 2023.
[41] V. P. Thoai, V.-T. Pham, G. Grassi, and S. Momani, "Assessing sigmoidal function on memristive maps," *Heliyon*, vol. 10, no. 6, Mar. 2024, Art. no. e27781.
[42] H. E. Williams and D. Lane, *Web Database Application With PHP MySQL: Building Effective Database-Driven Web Sites*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2004.
[43] E. Yegnidemir and R. Khondoker, "SPSDN: A security platform for SDN networks with an AI-based SQL injection attack detection and mitigation as an example service," in *Proc. IEEE Int. Conf. Mach. Learn. Commun. Netw. (ICMLCN)*, May 2024, pp. 1–2.

**DEBENDRA MUDULI** (Member, IEEE) received the M.Tech. degree in computer science and engineering and the Ph.D. degree in computer science from the National Institute of Technology, Rourkela, in 2016 and 2022, respectively. He is currently an Associate Professor with the Department of Computer Science and Engineering, C. V. Raman Global University, Bhubaneswar, India. His research interests include pattern recognition, medical image analysis, software defect prediction, and cloud computing. His work has been published in numerous journals and international conferences. His a Life Member of the Indian Society for Technical Education (ISTE). He is actively involved in the academic community, serving as a reviewer for various reputable publications.

**SHANTANU SHOOKDEB** received the B.Tech. degree in computer science and engineering from C. V. Raman Global University, in July 2024. He is currently an Expert in various types of machine learning and deep learning models. His research interests include image, mask, and video processing, including the modification in different neural network architecture to expand its performance.

**ABU TAHA ZAMANI** (Member, IEEE) has been a Lecturer with the Department of Computer Science, Faculty of Science, Northern Border University, Arar, Saudi Arabia. He has several research articles in reputed International journals. His research interests include cloud computing, ad-hoc network, cyber security, AI, the IoT, machine learning, and data science. He is also a member of various international journals like ACM. He is an editorial board member of some reputed international journals in computer sciences. He serves as a reviewer for various journals.

**SURABHI SAXENA** received the Ph.D. degree from the Department of Computer Application, Babu Banarasi Das University, Lucknow, Uttar Pradesh, India, in 2021. She is currently an Assistant Professor with the Department of Computer Science and Engineering, Christ University, Central Campus, Bengaluru, India. She has more than five years of teaching experience and six years of research experience. She is having one national patents. Her research and publication interests include artificial intelligence, machine learning, security software quality software, software engineering, and soft computing. She is also working in the area of E-Commerce, E-Governance, hybrid data security system, Voronoi partitioning, deep learning, data science, and the IoT. Her research has been recorded in over 20 journal publications and international conferences and five international conference reviewer. She is a life-time member of IAENG and IACSIT and the Editor-in-Chief and an Editor of Blue Eyes Publications and Soft Computing Research Society Technical Program Committee Member of Christ University.

**ANURADHA SHANTANU KANADE** received the M.C.A. and B.Sc. degrees in physics from Shivaji University, the M.B.A. (B.A.) degree from SPPU, the M.Phil. degree from BVDU, and the Ph.D. degree from Savitribai Phule Pune University. She is currently the Program Director of the Department of Computer Science and Applications, Dr. Vishwanath Karad-MIT-World Peace University, Pune. She is enthusiastic and passionate for teaching and research and works with dedication and positive approach. She has a vast experience of about 23 years, including 21 years of teaching and about two years of industry experience. She is approved as a Ph.D. supervisor with Dr. Vishwanath Karad-MIT World Peace University. Her research interests include databases, data science, blockchain technology, cloud computing, and algorithmics. She is a member of the professional bodies, including ACM, Informing Science Institute, a BOS Member of the Department of Computer Science and Applications, Dr. Vishwanath Karad-MIT World Peace University, Pune, Maharashtra, a BOS Member of the Vivekanand Autonomous College, Kolhapur, Maharashtra, Pune, a TPC member, a reviewer for many conferences and journals, and associated with state and private universities for research activities and collaborations.

**NIKHAT PARVEEN** received the B.Sc. degree in computer science and the M.C.A. degree from Andhra University, Andhra Pradesh, India, in 2000 and 2003, respectively, and the Ph.D. degree from the Department of Computer Application, Integral University, Lucknow, Uttar Pradesh, India. She is currently an Associate Professor with the Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Guntur, India. She has more than 12 years of teaching experience and six years of research experience. She is having six national patents. Her research and publication interests include artificial intelligence, machine learning,, security software, security testing, software engineering, and requirement engineering. She is also working in the area of soft computing, image analysis, big data analytics, and the IoT. Her research has been chronicled in over 30 journal publications and international conferences. She is a life-time member of CSI, ACM, IAENG, and IACSIT.

**MOHAMMAD SHAMEEM** received the Ph.D. degree from Indian Institute of Technology (Indian School of Mines), Dhanbad. Currently, he has more than five years of teaching experience and more than ten years of research experience. He is also a Postdoctoral Researcher with IRS-ISS, KFUPM, Saudi Arabia. His research interests include empirical software engineering, cloud computing, machine learning, global software engineering, and systematic literature review. He has published various papers in well-reputed SCI and Scopus journals, i.e., *Information and Software Technology*, *Journal of Software Evolution and Process* (Wiley), *Applied Soft Computing* (Elsevier), and *Arabian Journal of Science and Engineering* (Springer). Moreover, he has presented his research in various reputed conferences.