Journal of ICT, Design, Engineering and Technological Science (JITDETS)
VOL. 10, NO. 1, pp. 45-56, 2026
DOI: https://doi.org/10.33150/JITDETS-10.1.5

ORIGINAL CONTRIBUTION

BERT-LGBM Model for Error and Union Attacks Detection in Web Application

Imdad Ali Shah[1]*, Noor Zaman Jhanjhi[2]

[1]FEST, Iqra University, Karachi, Pakistan

[2]School of Computer Science, Malaysia

Abstract— The rapid growth of web-based applications has mostly increased the threats of SQL injection (SQLi) attacks, which remain the most critical risks to data security and system integrity. SQLi is one of the severe and persistent threats to data confidentiality. SQLi attacks exploit vulnerabilities in web apps input fields, permitting adversaries to manipulate queries and obtain unauthorized access to sensitive data. In the modern era, SQL injection attack types have increased, with error-based attacks being the most critical security concerns for web app firewalls. Several industries are vulnerable, such as online banking e-commerce, healthcare, financial institutions and government services. With the growing trust in digital infrastructures, attackers use advanced techniques to exploit vulnerabilities in database queries to obtain illegal access to personal information. Traditional detection systems, such as Static, Dynamic, and Manual Analysis, are insufficient for detecting new methods and SQLi attacks due to their static nature and limited adaptability in web apps traffic. The purpose of this article is to build an AI-based model for detecting accurate and robust SQLi (error-based) attacks. This research aims to give intelligent solutions the ability to secure NLP applications against the complicating and changing attack vectors. Our study contributes to advancing web apps security by giving an effective and scalable AI-based solution for SQLi (error-based) attacks detection. Our proposed model has achieved results, accuracy 0.99, precision 0.98, recall 0.97 and F1 0.99. Outperforms existing approaches in SQL injection (error-based) detection, demonstrating superior performance compared to the RF models. While BERT-LSTM achieved slightly lower performance, accuracy: 0.97, precision: 0.963, recall: 0.962, F1-score: 0.958. The RF model matched the proposed model in accuracy 0.99 and F1-score 0.98 while achieving the highest recall 0.997, indicating a strong detection model. These results highlight the robustness and reliability of the proposed model in balancing precision and recall, making it more effective for real-world SQL injection (error-based) detection tasks.

Index Terms— Error and Union attacks detection, BERT, AI-based, LGBM, and Web apps

Received: 3 February 2026; Accepted: 26 March 2026; Published: 10 June 2026

I. INTRODUCTION

SQL Injection (SQLi) attacks remain one of the most persistent and dangerous threats to web apps, allowing attackers to gain unauthorized access to sensitive databases. Among the most common forms, error-based SQLi attack techniques are widely used to exploit vulnerabilities and retrieve confidential information. Traditional detection mechanisms, such as signature-based and rule-based systems, often fall short in identifying obfuscated or novel attack patterns. The benefits of this method for extracting SQL injection statements and detecting SQL injections are well demonstrated by the testing findings. The model's detection of payloads that have undergone several encodings to get around restrictions is not accurate enough.

The figure below presents the OWASP to 10 vulnerabilities, command injection, XML and communication, and risk between the web server and database. Figure 1 process of SQL injection attacks. It provides the foundation of web apps, organising and storing data, including user digital information, content, and other crucial elements that are directly necessary for their operation [1, 2]. Databases hold both organised and unstructured data, including session data, user profiles, content, and transactions [3, 4]. To satisfy user requests and application functionalities, databases retrieve data using queries, data filters, and sorting algorithms [5, 6]. Digital information for unauthorised access to sensitive data, data theft, or data destruction. An attacker uses this approach, which involves injecting SQL commands into an input field. Attackers employed prepared statements that differentiate between code and data in parameterised queries.

The contribution of this research is as follows:

- This proposed AL-based mode integrates BERT-LGBM to improve SQLi detection.
- This research provides an AI-based model to detect SQL injection (error-based) attacks.
- This AI-based model supports the web app firewalls in detecting error-based attacks.
- The proposed model has been compared with existing models. This proposed model has improved results such as accuracy, precision, recall and F1 score in comparison.

*Corresponding author: Imdad Ali Shah
†Email: imdad.ali@iqra.edu.pk

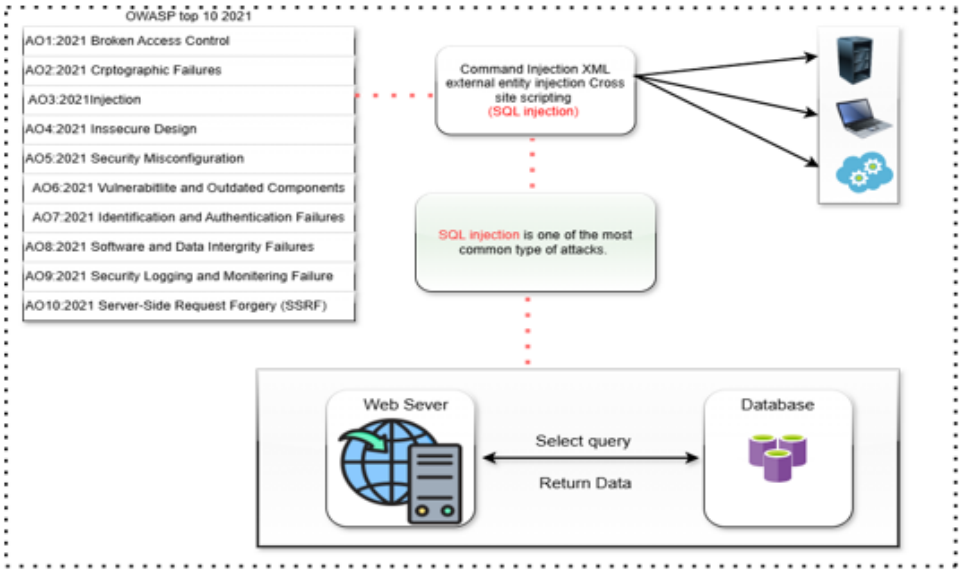

Fig. 1. process of SQL injection attack

This paper is organised as follows:

Section 2 provides a deep examination of the current research in SQL injection (error-based) attacks, focusing on significant studies and existing strategies for detecting attacks within the web apps firewall environment. Section 3 presents the results and gives an empirical evaluation of the proposed AI-based detection model. Section 4 provides a comprehensive discussion, comparing the results with existing models in SQL injection detection attacks. Section 5 provides summarizes of findings for future research.

II. LITERATURE REVIEW

SQL injection attacks have significantly increased in the digital era and have been observed in their attacks, error-based attacks in different industries such as health, logistics, finance, banking, E-commerce, and retail. In line with trends in SQL injection attacks throughout time, 42% of global cyber-attack attempts on systems that are visible to the public are SQL injection-based. More than 21% of businesses, sectors, and institutions continue to be susceptible to SQL injection attacks. The greatest SQL injection attack in history resulted in the theft of over 1 billion credentials, including passwords and user IDs. On the contrary, attackers were able to obtain 130 million user card details through these types of attacks. Furthermore, 200,000 users' credit details have been stolen when a significant SQL injection was targeted at guess.com, an online fashion web apps store [7, 8]. A special dynamic SQL injection detection tool based on the ability to distinguish between HTTP requests sent by clients or users. Its shortcoming, however [9, 10]. A system that integrates tools for both static and dynamic analysis.

TABLE I
MACHINE LEARNING MODELS

Citation	Problem	Proposed approached	Vulnerabilities	Dataset
[11]	Security Holes	PM	External scripting	Projects
[12]	Manual analysis	Hybrid analysis	OB	lighted, Wsmp3d, Kristen
[13]	The current situation necessitates human assistance for classification	Supervised learning	SQL injection	CVSS
[14]	Dynamic	SL	SQLi	NVD
[15]	Current vulnerability prediction models	SL	Unavailable vulnerability data	Bugzilla
[16]	Vulnerability scanner	SL	Unavailable vulnerability data	288 DARPA
[17]	Offline analysis in the modern environment	SL	OS command injection	PHPMyAdmin
[18]	Conventional methods	DL	Overflowing buffers (OB)	NVD, SARD
[19]	No possible danger was discovered	DL	Injection of SQL	SARD, NVD
[20]	Traditional techniques for security analysis	DL	OB	NVD, SARD
[21]	Finding and taking advantage of security flaws	DL	OB	21 Binary
[22]	Conventional methods	DL	OB	NVD, SARD
[23]	Software Vulnerabilities	DL	Injection of SQL	SARD, NVD
[24]	Traditional techniques for security analysis	DL	OB	NVD, SARD
[25]	Finding and taking advantage of security flaws	DL	OB	21 Binary
[26]	Automated vulnerability detection for PHP	DL	SQLI	SARD

The significant increase in web application attacks has prompted experts to step up their efforts to combat them and lessen their impact. They have focused heavily on safeguarding user privacy and securing data that is sent between different parties via the Internet. According to statistics, SQL Injection has been the most common attack on online applications in recent years, accounting for the largest percentage of all attacks. According to some studies, they are responsible for 27% of all attacks. Presented a detection technique that preserves the structure of the original SQL queries by converting them into token sequences [27, 28]. After that, these sequences are represented as a graph with weighted edges representing the interactions between tokens, which serve as nodes. A system for static analysis intended to find SQL injection flaws during compilation. In addition to describing and comparing their suggested CNN model with other experimental machine learning models, their research examines several sorts of SQL injection attacks. They provided model performance indicators in their analysis, including the ROC curve, recall, accuracy, and precision [29, 30]. An RNN model for SQL injection attack detection that is built on a deep learning model [31]. To classify injection attacks, they suggested a unique technique that uses recurrent neural networks (RNNs) to capture the syntactic and semantic feature maps of SQL queries. They presented models in their study that can be used for query analysis and other natural language processing tasks [32]. Identifying SQL injection attacks using a CNN-based technique that extracts pertinent payloads from network flow data. To conduct SQL injection classification, their CNN-based model integrates high-dimensional aspects of SQL behaviour. One crucial area for further study is the incorporation of deep learning and machine learning into network security [33]. These strategies successfully overcome the drawbacks of traditional detection techniques, especially when dealing with large amounts of data. Machine Learning support helps reduce SQLi attacks. Table I presents machine learning models.

This table shows different machine learning and deep learning models' descriptions of technical methods, approaches, types of vulnerabilities and datasets.

In order to control SQL injection on web platforms, they discuss the advantages and uses of ML models and artificial intelligence in their study. When using detection approaches to control SQL injection, they have demonstrated encouraging results. There have been some suggested reviews regarding SQL injection detection and identification. A system for static analysis intended to find SQL injection flaws during compilation. While web applications streamline several tasks, particularly those which require assurance in the digital realm, these platforms are still susceptible to other cybersecurity-related issues. As a result of application network data flows, user databases are maintained with a wide array of sensitive information such as financial details, personal identification data, user passwords and a lot of confidential material. Through taking labelled samples from the internet and adding a SQL-injection-payload list, the study created a dataset. Techniques for classification, detection and mitigation of software vulnerabilities are crucial for locating and reducing security threats in software systems. These approaches can be grouped according to what they employ, the stage of software lifecycles at which they are used, the kind of vulnerabilities they target, and the detection methodology. Curiosity, personal interest, and the desire to find a solution are some of the factors that may drive researchers to investigate a given subject. Deep learning (DL) models are increasingly applied to detect SQL injection (SQLi) attacks in web applications. These models automatically learn complex patterns in user inputs that may indicate malicious behavior. DL approaches, such as recurrent and convolutional neural networks, can identify both known and zero-day SQLi attempts. Unlike traditional detection methods, DL reduces reliance on manual feature engineering.

A. Overview model BERT

BERT is a powerful language model developed, and it plays a significant role in utilising contextual interpretation information from previous and next directions. This predicts the lexical vocabulary sense of a statement within a phrase [34, 35, 36]. The BERT model has a multi-headed self-attention mechanism [37, 38]. BERT requires that the input data follow a particular format, specifically [CLS]sentence [SEP]. [39] BERT creates word vectors with contextual information, builds several Transformer sections and applies the self-attention process to evaluate the connections between every word. The representation for the complete input sentence is simultaneously embodied in the vector linked to the [CLS] marker. Their similarity is smaller, suggesting that they are closer in vector space [40, 41, 42]. These four terms have comparable three-dimensional vector representations and are frequently seen together in malicious SQL queries inside network requests. Figure 2 BERT preparation and Fine-tuning.

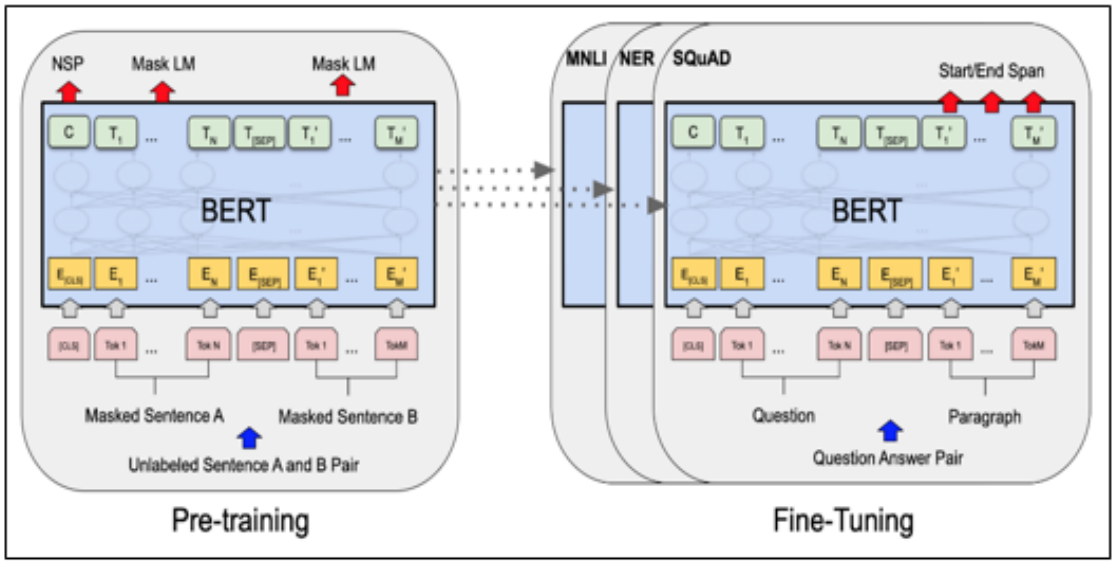

Fig. 2. BERT preparation and Fine-tuning [43]

Every parameter is adjusted during the fine-tuning process. Every input sample now has the special symbol [CLS] preceding it, and [SEP] serves as a special separator token. The transformer model advances the field of natural language processing (NLP) research at an astounding rate by offering a significant solution to the persistent issues with sequential manipulations [44]. The embeddings are sent to the encoder blocks, which consist of a feed-forward network and multi-head attention. One by one, the attention vectors are sent to the decoder block via a feedforward network. It is important to note that the multi-head attention layer achieves parallelisation due to the independence of the attention networks. Figure 3 BERT model architecture.

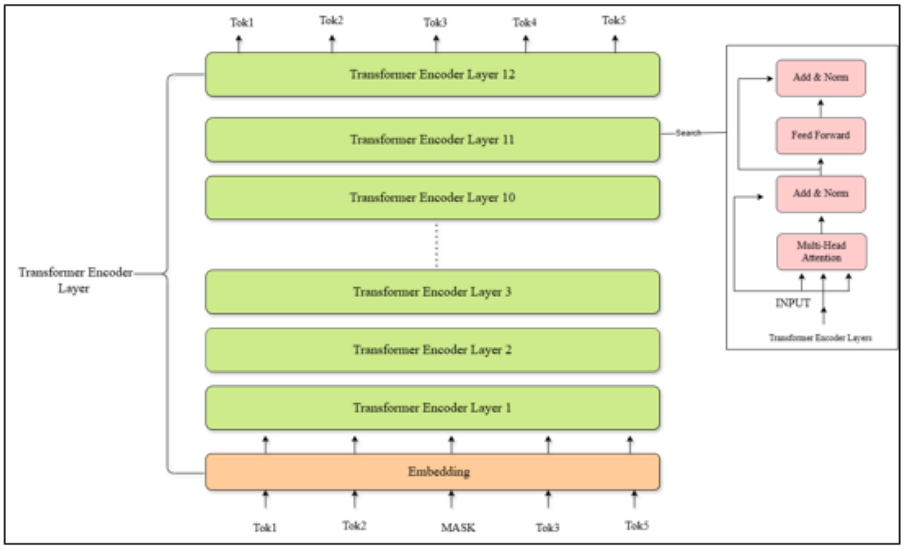

Fig. 3. BERT model architecture

Provides empirical evaluations of transformers and convolutional models on a range of natural language processing tasks, highlighting transformers' competitive edge [45]. Fixed-length segments result in a limited dependency length, and text splitting into fixed-length parts results in context breakdown and an ineffective assessment process [1]. The contextual meaning of the input words is extracted by embedding them and passing them through positional encoders, which give word vectors according to where they appear in a phrase [26]. The embeddings are sent to the encoder blocks, which consist of a feed-forward network and multi-head attention.

This research study was conducted through an extensive review of various interconnected domains, including SQL injection attacks, such as error-based attacks, In-band SQLi, Inferential SQLi, and Out-of-band SQLi. We have selected two error-based attacks. A thorough analysis of present defence mechanisms and their areas of solution follows. This helped in classifying proposed attack detection, such as machine learning-based solutions.

III. METHODOLOGY

Data collection is the process of collecting information from multiple sources to evaluate and make it defensible. Depending on the goals and data type being collected, it can be broken down into several processes and utilize a variety of techniques and instruments. We have collected more than two hundred articles, international conferences, and book chapters from different databases; the details are in Table II. The phases involved in the data collection process can be better understood and communicated by using illustrations. Figure 4 presents the data collection for the proposed model.

TABLE II
DETAILS OF THE DATA COLLECTION

Data Source	2020	2021	2022	2023	2024	Total
Articles						
IEEE	21	28	25	25	18	117
Elsevier	18	16	22	19	20	95
ScienceDirect	10	9	12	11	8	50
Google Scholar	6	5	8	7	4	30
International Conference & Book Chapters						
IEEE	3	4	6	5	4	30
Elsevier	2	5	4	3	2	16
ScienceDirect	2	2	5	2	3	14
Google Scholar	1	3	2	3	2	11

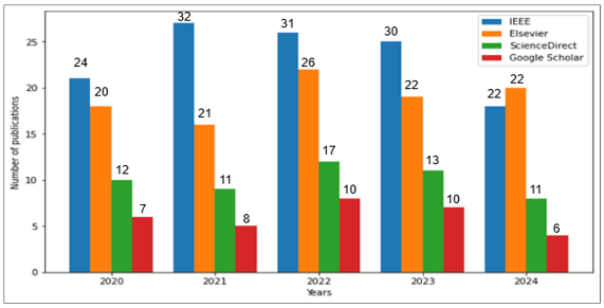

Fig. 4. Data collection for the proposed model

This figure presents the collection and investigates previous research on categorisation and software vulnerability identification across a range of issues and source codes from 2020 to 2024. Since this study intends to look more closely at the application of ML techniques in software flaw detection and classification, we also examine articles that use these techniques to identify software vulnerabilities and web application firewalls. Figure 5. Flowchart of the data collection process for the proposed model.

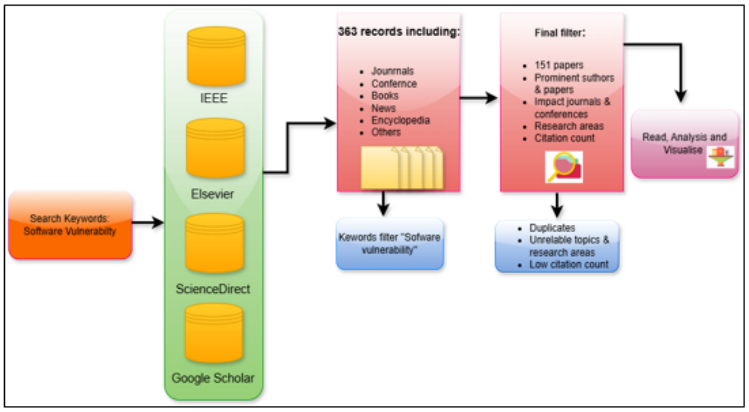

Fig. 5. Flowchart for data collection for the proposed model

Using "software vulnerability" as our primary keyword, we searched for three databases ScienceDirect, IEEE Xplore, and Google Scholar, for pertinent research papers to find the ones used for this study. These databases were selected because they provide a wealth of research articles on a variety of subjects, which raises the visibility of research papers in vulnerability identification. However, many of these records originate from several study fields that are somewhat connected to identifying vulnerabilities. The basis of this research is an analysis of 151 excellent and highly relevant experimental studies on software vulnerability identification. After locating all pertinent articles from the sources, we carefully read and examined each one, taking note of specifics such as the problem statement and the detection strategy employed.

IV. RESEARCH METHODOLOGY

For conducting this research study, the Design Science Research (DSR) methodology is employed, as outlined in Figure 6. It is an objective-centred paradigm in the research methodology domain, and it is applied in this article for aligning methods with the research study and proposed model objectives and making the findings credible and replicable for further research.

.

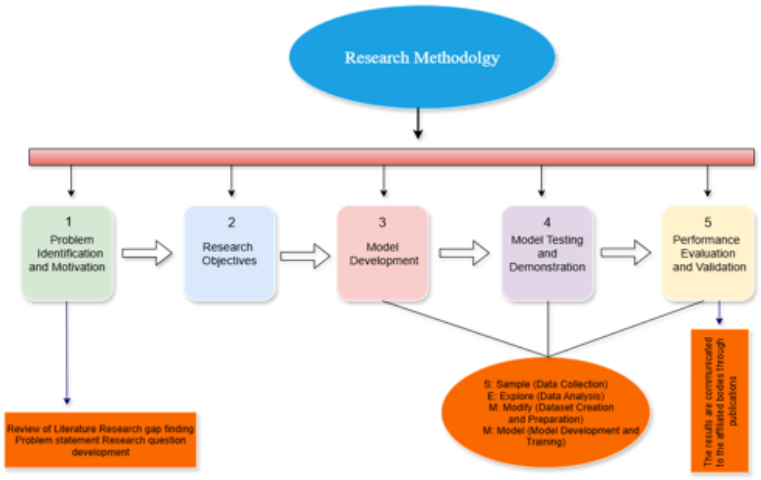

Fig. 6. Design science research methodology

A. Research design

In the research design, we have made a strategy to conduct research to find a specific solution for SQL injection detection (error-based attacks) and got improved results in comparison with existing models and data analysis aimed for reliability, validity, and generalization

The figure 6 presents the five steps: First, problem identification, in which we collected related data for the identification of the problem; second, research objectives; third, model development; fourth, model testing and fifth, performance evaluation

B. Dataset

In this research, we have collected a Httpparams dataset from GitHub, which is designed for detecting malicious web requests and contains 20713. This dataset provides a balanced and comprehensive foundation for training and evaluating AI-based models in web apps security research, making it ideal for academic and practical studies in cyber threat detection. There are several SQL injections available in the dataset as malicious entry types, error-based.

C. Normal SQL queries

Non-malicious in nature queries have been observed in the database retrieval from the table and used on account of rename, delete, exec, drop, select, and database use characters are III ,#,/*, ”,
, =„ /**/, @@,.

D. SQL injection attack queries

TABLE III
DESCRIPTION OF THE SQLI QUERY AND THE NORMAL SQL

SQL Injection Attack Queries	Normal SQL Queries
+, -, ?, %, :, \, /	/**/, @@, .
*, ;, _, -, (,), =	:–, #, /*
{, }, @, &, [,]	”, \\, =, ,

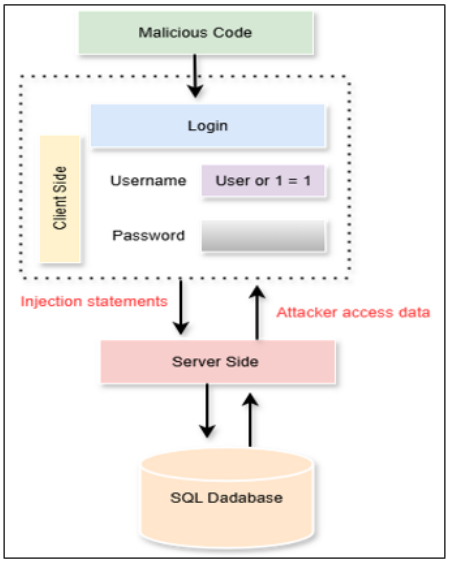

Fig. 7. SQL injection attacks

This figure 7 illustrates the SQLi attack process, showing how attackers insert malicious queries through the client-side input method and gain access to sensitive data on the server-side SQL database, while the web app's firewall receives the malicious request.

E. System model design

The model has been evaluated through the standard parameters, including accuracy, precision, recall, and F1 score. Confusion matrix analysis was performed for validation against true, false positive, and negative rates. The evaluation metrics were selected based on benchmark research and trends adopted in similar research works. The LGBM algorithm was applied to evaluate classification parameters for an extensive performance evaluation and validation task using the cross-entropy function. Figure 8. Overview of the proposed System Model.

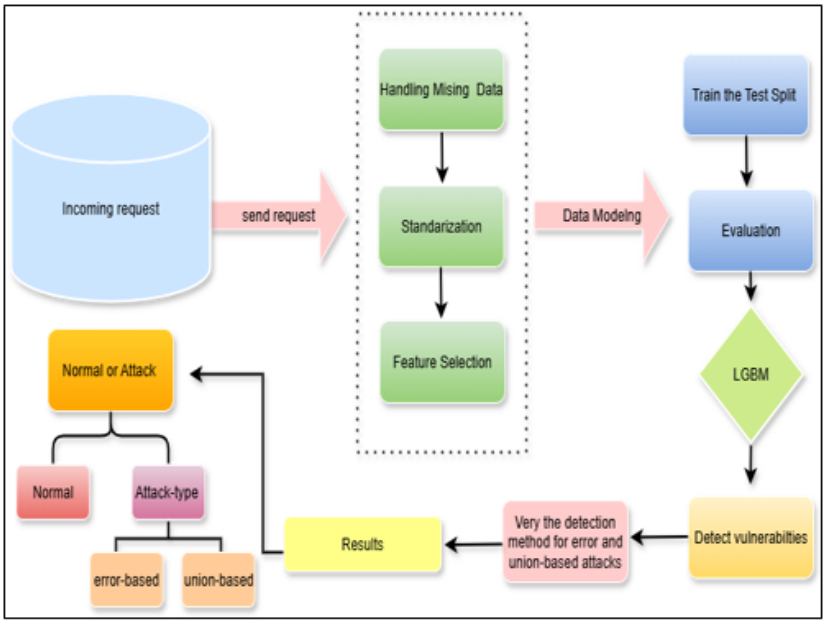

Fig. 8. Overview of the system model adopted for this study

The proposed model consists of a several-layer architecture designed to process incoming HTTP requests, extract relevant features, and perform error-based attack detection in real time.

TABLE IV
OVERVIEW EMBEDDED WORD VECTOR

Words	Vector 1	Vector 2	Vector 3
SELECT	0.52	0.14	-0.16
Select	0.51	0.13	-0.15
FROM	1.36	0.93	-0.05
WHERE	1.57	1.24	-0.8

Tables show the embedded word and the conversion process of the word into a number for machine understanding. Table IV Example of embedded word vector and Figure 9 presents the algorithm results of the proposed algorithm.

$$\text{similarity} = \cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|} \quad (1)$$

Algorithm 1: BERT-LGBM Preparation for Proposed Model Preparation

Input: Set all datasets [x1,x2,x3,.....xn] in Httpparams-LGBM data pool
Output: prepared, pre-processed, feature-engineered Httpparams dataset, error and union
1: BE GIN for x belonging to HttpparamsLGBM, do
2: ***Import:*** panadas as 'pd', && numpy as 'as',&& cvs
3: ***Load:*** Httpparams-LGBM [x1,x2,x3,.....xn]
4. END
5. BEGIN for all *x* belonging to HttpparamsLGBM, do
6. Data = pd.read_csv('payload_rpl.csv',r)
7. Print (dataFrame.head(10))
8. (dp.data = = empty|| empty||undefiened
9. Dp.data = = NaaN|| dp.data = = 1
10. **if** (f1.data = = f_n.data)
11. F_n = drop
12. Else
13. Dp.data = value && f.data = intact
14. **If** (data.dtpes ≠ numerical)
15. Data.dtypes = oneHotEncode
16. **Else**
17. Data.dtypes = num
18. Check data.isnull()
19. **True if** dp.data = = True
20. For (data.data.isnull() = = True
21. Data = oneHotEncode
22. **Until**
23. Return preprocessed, prepared, Httpparams dataset, error and union
24. EN

Fig. 9. Algorithm 1: BERT-LGBM Preparation for Proposed Model Preparation

Fig. 10. Flowchart for data collection for the proposed model Organizations

F. Process flow

This section describes the process flow of the proposed model in detail. These datasets receive a basic descriptive analysis before being merged and sent for preprocessing. In the pre processing step, to increase the model's classification and detection accuracy, categorical features were handled to enhance the model's detection and classification accuracy. A training set and a testing set are then created from the dataset. As is usual in the relevant literature, an 80:20 ratio is employed. The testing set is saved for model validation using performance evaluation metrics, while the training set is used to build the model. Figure 12. Flow Diagram adopted for this study.

Fig. 11. Flow Diagram adopted for this study

In the first step, we perform SQLi initiation, and in the second step, the flow chart shows the benign SQLi model construction. In the third step, the malicious request is implemented in the attack models, and the fourth step model is activated for the attack detection, such as error-based (analysis). In the fifth step, fine-tune the model (preprocessing data consolidation). In the sixth step, the flow chart classifies the target classes with the required accuracy; if not, go to the fifth step or yes, end the program. Table V Description of confusion matrices.

TABLE V
DESCRIPTION OF CONFUSION MATRICES

Malicious	Predicted (Malicious)	Predicted (Normal)
Malicious	TP	FN
Normal	FP	TN

$$\text{Accuracy Error} = \frac{TP_{\text{Error}} + TN}{TP_{\text{Error}} + FN + TN + FP_{\text{Error}}} \quad (1)$$

$$\text{Accuracy Union} = \frac{TP_{\text{Union}} + TN}{TP_{\text{Union}} + FN + TN + FP_{\text{Union}}} \quad (2)$$

$$\text{Precision Union} = \frac{TP_{\text{Union}}}{TP_{\text{Union}} + FP} \quad (3)$$

$$\text{Precision Error} = \frac{TP_{\text{Error}}}{TP_{\text{Error}} + FP} \quad (4)$$

$$\text{Recall Union} = \frac{TP_{\text{Union}}}{TP_{\text{Union}} + FN} \quad (5)$$

$$\text{Recall Error} = \frac{TP_{\text{Error}}}{TP_{\text{Error}} + FN} \quad (6)$$

$$\text{F1 Score Error} = \frac{2\,TP_{\text{Error}}}{2\,TP_{\text{Error}} + FP + FP_{\text{Union}}} \quad (7)$$

$$\text{F1 Score Union} = \frac{2\,TP_{\text{Union}}}{2\,TP_{\text{Union}} + FP + FP_{\text{Union}}} \quad (8)$$

V. RESULTS AND DISCUSSION

We have obtained the BERT model from Google's GitHub open-source code and the LGBM algorithm implemented through Keras. The dataset has been divided into two sets in a CSV file, 80% and 20%. We develop specific word segmentation rules, as presented in Table III, to handle special cases. These rules include function bodies that combine characters with brackets, equations, equal signs, and special symbols. While the specific characters were handled as words through regular expressions and attached in single quotes, this creates a basic word list. We used BERT's tokenization method to ensure they meet the input requirements of the BERT models. Figure 12 shows the tokenize process, Figure 13 presents the Training and Testing Accuracy of the error-based attack, Figure 14 presents the training and testing Precision of the error-based attack, Figure 15 presents the training and testing Recall of the error-based attack, Figure 16 presents the confusion matrices of the F1-score error-based, Figure 17 presents the comparative analysis of the proposed model and Figure 18 Comparison of data processing techniques.

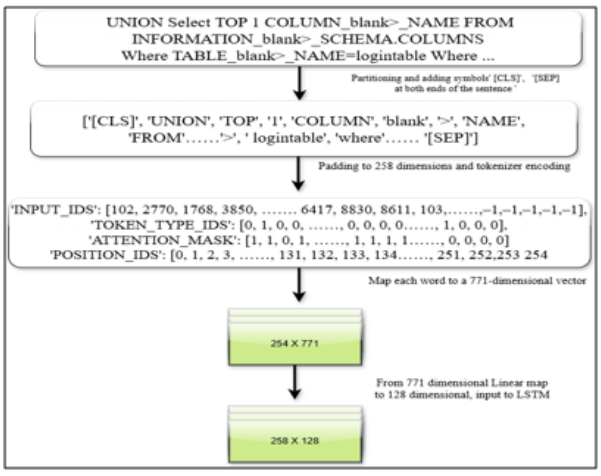

Fig. 12. Tokenize process Organizations

A. Proposed model

We presented our proposed model, which works to convert the text into numbers for machine understanding and captures semantic and syntactic relations between words.

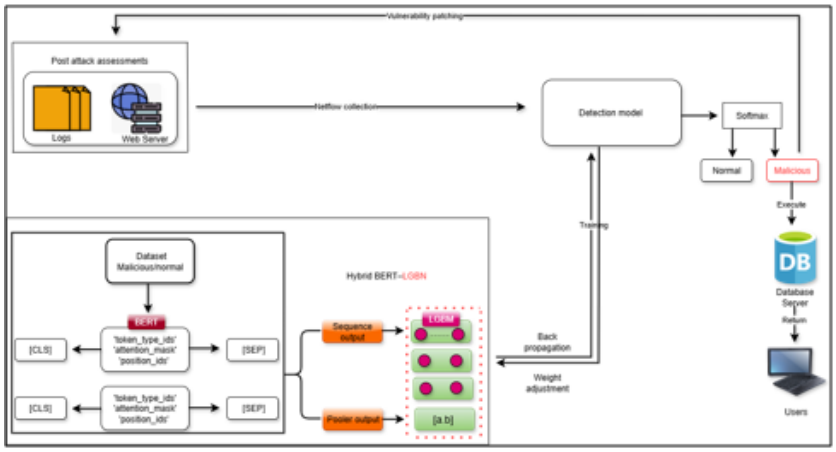

Fig. 13. Flowchart for data collection for the proposed model Organizations

Our proposed model converts text into numbers for machine understanding, captures the syntactic relationships between words, and detects the SQLi (error-based attack) after the BERT pooled hidden state. Figure 12 Proposed Model Description

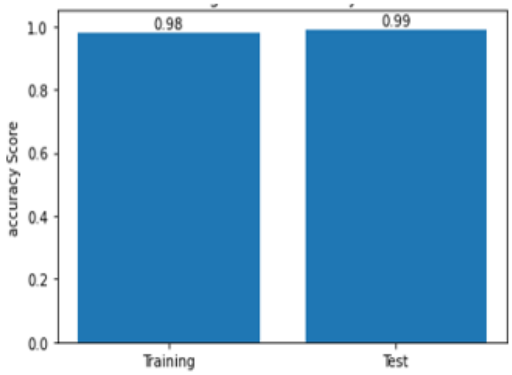

Fig. 14. Training and testing Accuracy of the error-based attack

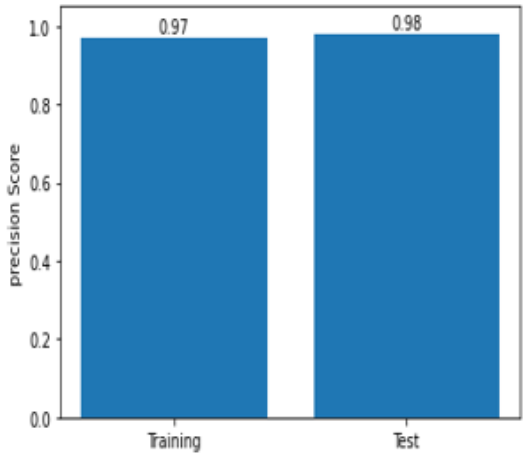

Fig. 15. Training and testing Precision of error-based attack

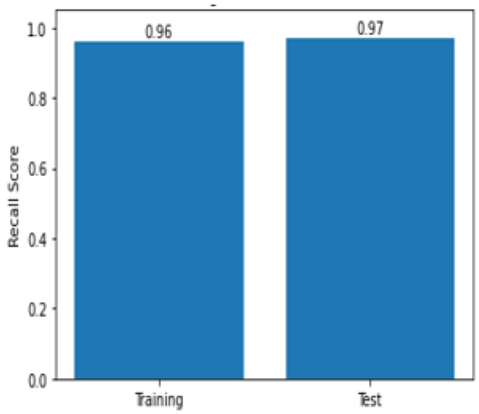

Fig. 16. Training and testing Recall of the error-based attack

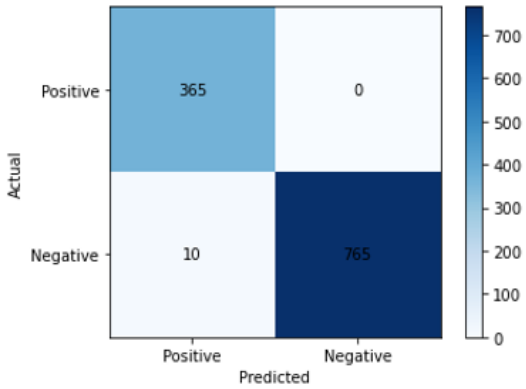

Fig. 17. Confusion matrices of the F1-score error-based

TABLE VI
PROPOSED MODEL RESULTS

Proposed model	Results
Accuracy	0.99
Precision	0.98
Recall	0.97
F1 Score	0.99

B. Comparison and Discussion

We have conducted two comparative experiments on SQLi detection capabilities, and computational analysis was conducted. Using the SQLi attack detection based on BERT and the LGBM algorithm in this article. Our proposed model has been compared to BERT-LSTM and Random Forest. The evaluation of the comparison was conducted with our proposed model to evaluate the NLP feature extraction of the BERT model, TF-IDE, Word2vec model and one-hot methods.

C. Benefit Demonstrated through Comparative Studies

We have used the same dataset for training on the server and analysis their accuracy, precision, recall and F1 score on the same test and the performance of different prediction models for comparison. The results of the evaluation of the comparison are presented in Figure 15. Comparison of data processing techniques.

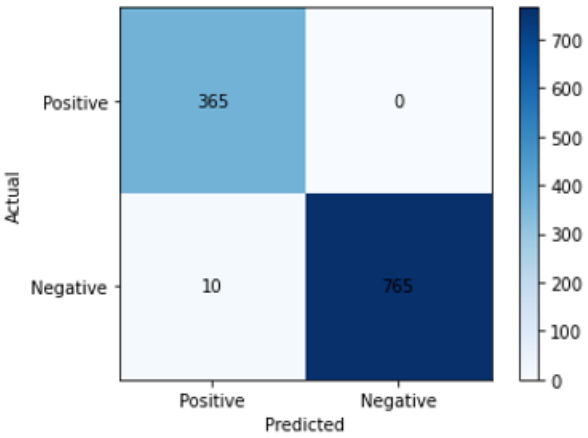

Fig. 18. Comparative analysis of the proposed model

This figure shows Details of the comparative analysis of the proposed model, which received improved results compared to the existing model.

TABLE VII
COMPARATIVE ANALYSIS OF EXISTING APPROACHES

Comparison Approach	Accuracy	Precision	Recall	F1
Proposed	0.99	0.98	0.97	0.99
BERT-LSTN	0.97	0.96	0.95	0.962
RF	0.99	—-	0.97	0.998

This table compares the proposed model to the existing model, including BERT-LST and RF. The results show our model received improved results compared to the existing model.

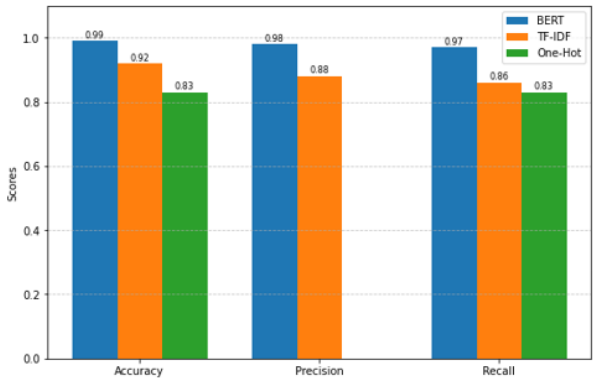

Fig. 19. Comparison of data processing techniques

VI. CONCLUSION

In conclusion, our research presents an AI-based model for the detection of SQL injection (error-based) attacks, using the BERT model and the LGBM algorithm. The proposed model showcases superior accuracy and F1 score, and identifies malicious queries characterised by a unique attacker and rare lexicon. Our study contributes to advancing web apps firewall security by giving an effective and scalable AI-based solution for SQLi (error-based) attacks detection. The results of the proposed model highlight the robustness and reliability of the proposed model in balancing precision and recall, making it more effective for real-world SQL injection (error-based) detection tasks. For future work, we need to refine this model, enhancing its capabilities using deep learning for the detection of SQL injection attacks.

References

[1] A. Salam, F. Ullah, F. Amin, and M. Abrar, "Deep learning techniques for web-based attack detection in industry 5.0: A novel approach," *Technologies*, vol. 11, no. 4, p. 107, 2023.

[2] T. Muhammad and H. Ghafory, "Sql injection attack detection using machine learning algorithm," *Mesopotamian Journal of Cybersecurity*, vol. 2022, pp. 5-17, 2022.

[3] R. A. Katole, S. S. Sherekar, and V. M. Thakare, "Detection of SQL injection attacks by removing the parameter values of SQL query," in *2018 2nd International conference on inventive systems and control (ICISC)*. IEEE, 2018, pp. 736-741.

[4] I. A. Shah, N. Z. Jhanjhi, and S. Rajper, "Use of deep learning applications for drone technology," in *Cybersecurity Issues and Challenges in the Drone Industry*. IGI Global Scientific Publishing, 2024, pp. 128-147.

[5] Y.-C. Wang, G.-L. Zhang, and Y.-L. Zhang, "Analysis of sql injection based on petri net in wireless network." *Journal of Information Science & Engineering*, vol. 39, no. 1, 2023.

[6] D. GABI, "Countermeasure to structured query language injection attack for web applications using hybrid logistic regression technique," *Journal of Nigerian Society of Physical Sciences*, 2022.

[7] U. Farooq, "Real time password strength analysis on a web application using multiple machine learning approaches," *Int J Eng Res Technol (IJERT)*, vol. 9, no. 12, pp. 359-364, 2020.

[8] M. Moh, S. Pininti, S. Doddapaneni, and T.-S. Moh, "Detecting web attacks using multi-stage log analysis," in *2016 IEEE 6th international conference on advanced computing (IACC)*. IEEE, 2016, pp. 733-738.

[9] I. A. Shah, N. Jhanjhi, and S. N. Brohi, "Iot smart healthcare security challenges and solutions," in *Advances in Computational Intelligence for the Healthcare Industry 4.0*. IGI Global Scientific Publishing, 2024, pp. 234-247.

[10] C. Basta, S. Darwish *et al.*, "Detection of sql injection using a genetic fuzzy classifier system," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 6, 2016.

[11] R. Pandey and J. P. Singh, "BERT-LSTM model for sarcasm detection in code-mixed social media post," *Journal of Intelligent Information Systems*, vol. 60, no. 1, pp. 235-254, 2023.

[12] A. Ezen-Can, "A comparison of lstm and bert for small corpus," *arXiv preprint arXiv:2009.05451*, 2020.

[13] N. Rai, D. Kumar, N. Kaushik, C. Raj, and A. Ali, "Fake News Classification using transformer based enhanced LSTM and BERT," *International Journal of Cognitive Computing in Engineering*, vol. 3, pp. 98-105, 2022.

[14] K. Kaur and P. Kaur, "Improving BERT model for requirements classification by bidirectional LSTM-CNN deep model," *Computers and Electrical Engineering*, vol. 108, p. 108699, 2023.

[15] C. Sriharsha, S. Rithwik, K. P. Prahlad, and L. S. Nair, "Intelligent learning assistant using bert and lstm," in *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE, 2021, pp. 1-6.

[16] A. Kumar, J. P. Singh, and A. K. Singh, "Explainable BERT-LSTM stacking for sentiment analysis of COVID-19 vaccination," *IEEE Transactions on Computational Social Systems*, vol. 12, no. 3, pp. 1296-1306, 2023.

[17] A. Noorian, A. Harounabadi, and M. Hazratifard, "A sequential neural recommendation system exploiting BERT and LSTM on social media posts," *Complex & Intelligent Systems*, vol. 10, no. 1, pp. 721-744, 2024.

[18] N. Mandela and F. Etyang, "Comparative analysis of deep learning models for effective denial of service (DoS) attack detection in network security," *Journal of Electrical Systems and Information Technology*, vol. 12, no. 1, p. 73, 2025.

[19] Y. Liu and Y. Dai, "Deep learning in cybersecurity: a hybrid BERT–LSTM network for SQL injection attack detection," *IET Information Security*, vol. 2024, no. 1, p. 5565950, 2024.

[20] Y. E. Seyyar, A. G. Yavuz, and H. M. Ünver, "An attack detection framework based on bert and deep learning," *IEEE Access*, vol. 10, pp. 68 633-68 644, 2022.

[21] A. Gupta and D. C. Misra, "Hybrid IoT security model with integration of LSTM, BERT, ROBERTA and transform learning for attack classification," *International Journal of Information Technology*, vol. 17, no. 8, pp. 4505-4522, 2025.

[22] M. N. Swileh and S. Zhang, "Unseen attack detection in software-defined networking using a BERT-based large language model," *AI*, vol. 6, no. 7, p. 154, 2025.

[23] M. Marali, R. Dhanalakshmi, and N. Rajagopalan, "A hybrid transformer-based BERT and LSTM approach for vulnerability classification problems," *International Journal of Mathematics in Operational Research*, vol. 28, no. 3, pp. 275-295, 2024.

[24] B. G. Bokolo, L. Chen, and Q. Liu, "Detection of web-attack using distilbert, rnn, and lstm," in *2023 11th International Symposium on Digital Forensics and Security (ISDFS)*. IEEE, 2023, pp. 1-6.

[25] Y. Yang and X. Peng, "Bert-based network for intrusion detection system," *EURASIP Journal on Information Security*, vol. 2025, no. 1, p. 11, 2025.

[26] A. Kachavimath, S. Vaishyar, A. Chugani, P. Singh, and S. A. More, "Ddos attacks detection in sdn using multi-feature selection approach and llms," in *International Conference on Inventive Communication and Computational Technologies*. Springer, 2025, pp. 65-74.

[27] A. A. Ashlam, A. Badii, and F. Stahl, "A novel approach exploiting machine learning to detect sqli attacks," in *2022 5th International Conference on Advanced Systems and Emergent Technologies (IC_ASET)*. IEEE, 2022, pp. 513-517.

[28] H. Fu, C. Guo, C. Jiang, Y. Ping, and X. Lv, "SDSIOT: An SQL injection attack detection and stage identification method based on outbound traffic," *Electronics*, vol. 12, no. 11, p. 2472, 2023.

[29] C. Zhao, S. Si, T. Tu, Y. Shi, and S. Qin, "Deep-learning based injection attacks detection method for http," *Mathematics*, vol. 10, no. 16, p. 2914, 2022.

[30] V. Kecman, "Support vector machines- an introduction," in *Support vector machines: Theory and applications*. Springer, 2005, pp. 1-47.

[31] S. Suthaharan, "Decision tree learning," in *Machine learning models and algorithms for big data classification: Thinking with examples for effective learning*. Springer, 2016, pp. 237-269.

[32] C. Wang, F. Jiang, and H. Yang, "A hybrid framework for text modeling with convolutional rnn," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 2061-2069.

[33] M. E. Basiri, S. Nemati, M. Abdar, E. Cambria, and U. R. Acharya, "ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis," *Future Generation Computer Systems*, vol. 115, pp. 279-294, 2021.

[34] G. Saon, Z. Tüske, D. Bolanos, and B. Kingsbury, "Advancing rnn transducer technology for speech recognition," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5654-5658.

[35] A. Bano, S. H. Hamzah, and E. B. Hafiz, "Interplay between gender & and influencing factors to determine physical activity of school children," *Journal of Management Practices, Humanities and Social Sciences*, vol. 9, no. 2, pp. 120-135, 2025.

[36] F. A. Jam, I. Ali, N. Albishri, A. Mammadov, and A. K. Mohapatra, "How does the adoption of digital technologies in supply chain management enhance supply chain performance? A mediated and moderated model," *Technological Forecasting and Social Change*, vol. 219, p. 124225, 2025.

[37] K. Jothi, N. Pandey, P. Beriwal, A. Amarajan *et al.*, "An efficient sql injection detection system using deep learning," in *2021 International conference on computational intelligence and knowledge economy (ICCIKE)*. IEEE, 2021, pp. 442-445.

[38] G. Moschogianni, "Pathways to team performance and reduction of silence behaviour: Exploring the role of benevolent leadership through moderating impact of corporate social responsibility," *Journal of Management Practices, Humanities and Social Sciences*, vol. 9, no. 2, pp. 93-106, 2025.

[39] W. Zhang, Y. Li, X. Li, M. Shao, Y. Mi, H. Zhang, and G. Zhi, "Deep neural network-based SQL injection detection method," *Security and Communication Networks*, vol. 2022, no. 1, p. 4836289, 2022.

[40] M. Zivkovic, M. Tair, N. Bacanin, Š. Hubálovskỳ, P. Trojovskỳ *et al.*, "Novel hybrid firefly algorithm: An application to enhance XGBoost tuning for intrusion detection classification," *PeerJ Computer Science*, vol. 8, p. e956, 2022.

[41] H. Gulzar, "Motivation as the link: Mediating effects of intrinsic and extrinsic drives on student success," *Journal of Advanced Research in Social Sciences and Humanities*, pp. 94-106, 2025.

[42] E. Ağdemir, M. Kuyucu, M. Yücedağ, and K. K. Ağdemir, "Assessment of xenopsin related peptide-1 levels in pregnant women with gestational diabetes mellitus," *Perinatal Journal*, vol. 33, no. 1, pp. 5-10, 2025.

[43] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171-4186.

[44] A. Salam, M. Abrar, F. Amin, F. Ullah, I. A. Khan, B. F. Alkhamees, and H. AlSalman, "Securing smart manufacturing by integrating anomaly detection with zero-knowledge proofs," *IEEE Access*, vol. 12, pp. 36 346-36 360, 2024.

[45] J.-B. Cordonnier, A. Loukas, and M. Jaggi, "On the relationship between self-attention and convolutional layers," *arXiv preprint arXiv:1911.03584*, 2019.