Arabian Journal for Science and Engineering
https://doi.org/10.1007/s13369-024-09916-4

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

UniEmbed: A Novel Approach to Detect XSS and SQL Injection Attacks Leveraging Multiple Feature Fusion with Machine Learning Techniques

Rezan Bakır[1]

Received: 24 July 2024 / Accepted: 22 December 2024

Abstract
Web applications are essential in the digital age, but their security vulnerabilities expose sensitive data and organizational integrity to sophisticated attacks. Among the most prevalent and damaging vulnerabilities in web applications are cross-site scripting (XSS) and SQL injection attacks. In this paper, we introduce UniEmbed, a unified approach for detecting XSS and SQL injection attacks using machine learning classifiers. This novel approach leverages natural language processing techniques, combining features from Word2Vec, the Universal Sentence Encoder (USE), and FastText to extract meaningful data from web applications. Extensive experiments were conducted using various machine learning classifiers on three benchmark datasets to evaluate the performance of the unified detection approach, demonstrating exceptional results. Experimental results demonstrate the superior performance of the MLP classifier. For the XSS attack dataset, the MLP classifier achieved an accuracy of 0.9982 and an F1-score of 0.9983, with minimal false positives and false negatives. Similarly, the hard voting classifier yielded the same outstanding results. For SQL injection attacks, the MLP classifier maintained exceptional performance, achieving an F1-score of 0.9980 and accuracy rates exceeding 0.9980 across two datasets. The classifier effectively minimized false positives and false negatives. The ROC curves further corroborate the effectiveness of the proposed method, indicating high true positive rates and low false positive rates. Furthermore, comparative analysis showed that the UniEmbed method consistently outperformed individual feature extraction methods across all classifiers. These findings indicate that the proposed UniEmbed method, particularly when combined with the MLP classifier, is highly effective in detecting both XSS and SQL injection attacks, making it a promising approach for enhancing web application security.

Keywords Word2vec · FastText · Universal sentence encoder · NLP · XSS attack · SQL injection attacks · Machine learning

1 Introduction

As our dependence on web applications continues to grow for a wide array of online tasks, ensuring their security becomes increasingly crucial. Among the most common and dangerous of these vulnerabilities are cross-site scripting (XSS) attacks and SQL injection attacks [1]. The Securelist report on the top web application vulnerabilities from 2021 to 2023 identifies SQL injection and cross-site scripting (XSS) as significant threats. SQL injection remains prevalent, affecting 43% of analyzed applications, and can lead to sensitive data breaches. XSS vulnerabilities were found in 61% of applications, often posing medium risk levels, but can be exploited for malicious actions like credential theft. Both vulnerabilities emphasize the need for robust security measures in web applications [2]. These alarming statistics highlight the pressing need for robust security measures to protect sensitive data and maintain user trust. Detecting and mitigating these vulnerabilities in real-time are critical to safeguarding user data and maintaining the integrity of web-based services. XSS attacks transpire when malicious scripts are inserted into web pages and subsequently executed by users' browsers without their awareness [3]. These attacks exploit the trust that users have in a website, allowing hackers to steal sensitive data, hijack sessions, and even distribute malware. As XSS attacks can directly target users, they pose a significant risk to personal information and can have severe consequences for both individuals and businesses. On the other hand, SQL injection attacks take advantage of vulnerabilities in web

✉ Rezan Bakır
rezan.bakir@sivas.edu.tr

[1] Department of Computer Engineering, Faculty of Engineering and Natural Sciences, Sivas University of Science and Technology, Sivas, Turkey

Published online: 12 January 2025

application input fields to directly manipulate the application's database. By injecting malicious SQL code, attackers can gain unauthorized access to the database, retrieve sensitive information, modify data, or even delete the entire database [4]. SQL injection attacks have been responsible for some of the most devastating data breaches in recent history. As web application vulnerabilities continue to evolve and become more sophisticated, traditional rule-based and signature-based security measures are often inadequate to protect against these dynamic threats [5, 6]. In tackling these obstacles, scholars have looked into machine learning (ML) methods to autonomously detect XSS and SQL injection assaults with intelligence [5–13].

This paper focuses on the unified detection of XSS and SQL injection attacks using machine learning classifiers. State-of-the-art feature extraction methods, including Word2Vec [14], the Universal Sentence Encoder (USE) [15], and FastText [16], are leveraged to enhance the effectiveness of the approach. By combining these advanced natural language processing techniques, the accuracy and robustness of the detection system are aimed to be improved, providing a comprehensive defense against web application vulnerabilities.

1.1 Motivation

The motivation behind using the combined features from the Word2Vec, the USE, and FastText in detecting XSS and SQL injection attacks lies in their ability to effectively capture semantic meaning and context from textual data. Web application vulnerabilities, such as XSS and SQL injection attacks, often involve manipulating text inputs to exploit weaknesses in the application's security. Traditional feature extraction methods may struggle to capture the intricate relationships and semantics present in natural language, making it challenging to detect sophisticated attacks. Word2Vec, USE, and FastText are advanced natural language processing techniques that excel at generating dense vector representations for words and sentences, preserving semantic similarities and context. By leveraging these powerful feature extraction methods, the proposed approach aims to enhance the ability of ML classifiers to distinguish between benign and malicious inputs accurately. Word2Vec captures semantic relationships between words, while USE provides a contextually aware representation of entire sentences or phrases. FastText, on the other hand, handles out-of-vocabulary words effectively and can handle subword information, making it suitable for handling unseen data. The fusion of Word2Vec, USE, and FastText features allows suggested approach to capture the intricacies of language and capture patterns that may be indicative of XSS and SQL injection attacks. This combination enhances the detection capabilities of ML classifiers, leading to more robust and reliable protection against evolving web application threats.

1.2 Contribution

This research contributes significantly to the realm of web application security and the identification of XSS and SQL injection attacks in several key ways:

1- An innovative unified method that merges Word2Vec, the USE, and FastText as feature extractors to detect both XSS and SQL injection attacks is introduced. By harnessing the capabilities of these advanced natural language processing techniques, the accuracy and efficiency of the detection system are elevated.
2- Extensive experiments using a wide range of ML classifiers were conducted to evaluate the performance of the unified detection approach. The comprehensive evaluation ensures the reliability and effectiveness of the system in detecting various attack patterns and scenarios.
3- The fusion of multiple feature extraction methods leads to improved detection accuracy, enabling our system to identify malicious code and suspicious patterns with higher precision, reducing false positives and false negatives.
4- This research focuses on real-world applicability, considering the importance of web application security in today's digital landscape. The proposed unified detection approach can be easily integrated into existing web application security systems, ensuring practical and scalable implementation.
5- In comparison with transformer-based models such as BERT, UniEmbed offers a significant advantage in terms of computational efficiency. While transformer models require substantial computational resources and longer training times, UniEmbed achieves similar detection accuracy with far fewer resources, making it more practical for real-time web application security scenarios.

Overall, this research presents an innovative and effective solution to address the challenges of detecting XSS and SQL injection attacks. The unified approach, coupled with the comprehensive evaluation and practical applicability, strengthens web application security and aids in mitigating potential cyber threats, making a valuable contribution to the field of cybersecurity.

2 Background

2.1 XSS and SQL Injection Vulnerabilities

Cross-site scripting (XSS) and SQL injection (SQLi) are among the most prevalent security threats to web applications [2]. These vulnerabilities allow attackers to exploit weaknesses in the way input data are processed and can lead to severe consequences, such as unauthorized access, data theft, and site defacement. XSS attacks occur when malicious scripts are injected into trusted websites [11]. These scripts are executed in the victim's browser, allowing attackers to steal cookies, hijack user sessions, or redirect users to malicious websites. For example, a basic XSS attack might involve inserting a < script > tag into a comment field on a blog, which would then execute malicious code in every user's browser who visits that page.

SQL injection attacks exploit vulnerabilities in web applications that do not properly sanitize input fields, allowing attackers to manipulate SQL queries [17]. By injecting malicious SQL code, an attacker can bypass authentication mechanisms, retrieve sensitive information, or even delete entire databases. For instance, a typical SQL injection (SQLi) attack might target a login form where a user inputs a malicious string, such as ' OR 1=1 --. This string manipulates the underlying SQL query, bypassing authentication checks and granting unauthorized access to the system. Such attacks, along with others, remain significant concerns for web developers and security experts due to their ease of execution, even with minimal technical expertise, and their potential to cause widespread damage.

2.2 Feature Extraction Methods

Effective detection of XSS and SQLi attacks depends heavily on the ability to accurately represent and analyze the input data. To achieve this, advanced feature extraction techniques like Word2Vec [14], the Universal Sentence Encoder (USE) [15], and FastText [16] have been employed. Each method captures different levels of semantic meaning and structural patterns from textual data, making them suitable for identifying malicious code.

Word2Vec is a popular technique for generating dense vector representations of words based on their surrounding context. Developed by Mikolov et al. [14], this method allows words with similar meanings to be represented close to each other in the vector space. It operates in two main models: Continuous Bag of Words (CBOW) and Skip-Gram, both of which capture relationships between words that are critical for understanding patterns in attack data, such as common SQLi phrases. Word2Vec was selected for its ability to generate dense vector representations of words based on their contextual usage in a corpus. This method effectively captures semantic relationships, which are essential for understanding the nuanced language patterns often exploited in web attacks. Moreover, its computational efficiency allows for quick processing of large datasets, making it suitable for real-time applications where timely threat detection is vital.

Universal Sentence Encoder (USE) [15] is a more advanced technique that generates embeddings at the sentence level, rather than focusing on individual words. This method captures both the meaning and context of entire sentences, making it highly effective for analyzing the structure of web inputs. USE is particularly beneficial in detecting complex attack patterns, where the malicious intent may be hidden in longer, more contextually nuanced inputs. The robustness of USE against variability in syntax and structure enhances its effectiveness in diverse web application scenarios.

To further enrich the feature extraction approach, FastText was utilized. FastText**,** developed by Facebook's AI Research lab, enhances Word2Vec by considering subword information, which allows the model to handle out-of-vocabulary words and capture finer nuances in text. FastText breaks words into smaller n-grams [16], making it highly effective for detecting obfuscated or encoded attacks, where traditional methods might fail to recognize slightly altered or misspelled malicious content. These three methods, when combined, provide a comprehensive feature set that captures word-level, sentence-level, and character-level patterns, allowing for more accurate and robust attack detection. The integration of these three methods provides a comprehensive representation of input data, combining word-level, sentence-level, and character-level embeddings. This multi-dimensional feature set enhances our model's ability to discern subtle differences between benign and malicious inputs, ultimately leading to improved detection accuracy of XSS and SQLi attacks while minimizing false positives and negatives.

However, while other NLP methods, such as transformer-based models like BERT, offer strong contextual understanding, their computational demands may limit their practicality for real-time web application security. In contrast, the proposed integrated approach leveraging Word2Vec, USE, and FastText strikes a balance between performance and efficiency, making it well-suited for immediate threat detection in web applications. The lightweight nature of UniEmbed leads to faster inference times, which is critical in scenarios where immediate detection of threats is necessary. For example, in real-time web application monitoring, delays in detecting XSS or SQLi attacks can lead to severe consequences, including data breaches or service disruptions. UniEmbed's efficient processing allows for rapid analysis of incoming requests, enabling quicker responses to potential threats.

3 Related Work

Web application security is of paramount importance due to the increasing prevalence of cyber threats that exploit vulnerabilities in web-based systems. XSS and SQL injection attacks, in particular, have risen as prevalent and perilous forms of assaults directed at web applications.

The literature related to XSS attack detection techniques is reviewed initially. Various approaches, including static analysis, dynamic analysis, and ML-based methods, have been explored by researchers to identify and prevent XSS attacks [11]. Static analysis entails examining the source code of the application to pinpoint possible vulnerabilities, while dynamic analysis observes the behavior of the application during runtime [18]. ML techniques, on the other hand, leverage the power of data-driven models to detect malicious patterns and identify XSS attack vectors. Numerous research papers have explored various approaches to address the critical issue of web application security, specifically focusing on detecting XSS attacks. Client-side and server-side detection methods have been proposed [18], each employing different techniques to counteract these threats effectively. For example, the proposed Noxes, as detailed in [19], functioned as a web proxy on the client side, utilizing a blend of manual and automated rules to thwart XSS attacks. Another study [20] introduced an approach based on the Knuth–Morris–Pratt (KMP) string matching algorithm to identify malicious code, enhancing defense against such threats. On the server side, SWAP (Secure Web Application Proxy), outlined in [21] research study, provided an effective approach for both identifying and thwarting XSS attacks. The proposed method involved a reverse proxy that intercepts HTML responses and a modified web browser capable of detecting script content, resulting in successful identification of exploits in real-world web applications. Furthermore, unsupervised ML techniques, showcased in [22] study, have also been utilized for XSS attack detection, ensuring a balanced load between clients and servers while employing divergence measures to pinpoint vulnerabilities. In addition, reinforcement learning (RL) has exhibited potential in addressing XSS detection challenges, as seen in the RLXSS approach proposed in [23] study, which leveraged adversarial and retraining models to fortify the detection system against adversarial threats.

Researchers have explored a diverse set of methods, including genetic algorithms, swarm-based intelligence, deep learning, and hybrid methodologies, to enhance the accuracy and speed of XSS attack detection [24, 25], and [26]. Notably, feature extraction has emerged as a crucial element in handling cybersecurity threats, enabling accurate and efficient detection of various cyber threats [27]. As a result, researchers have been diligently focusing on enhancing feature extraction techniques in their studies. A wide array of methods has been employed to represent data effectively, ensuring optimal performance. For instance, in the study [28], a novel feature extraction approach utilizing the autoencoder structure was introduced for classifying Android malware applications. Similarly, [29] and [30] presented image-based feature extraction methods to tackle cybersecurity challenges.

Recent investigations have refined feature extraction methods, utilizing transformer-based techniques like BERT, RoBERTa, AlBERT and Elmo embeddings, as demonstrated in [31–33] and [34], to effectively analyze and detect cyber threats across different data types and scenarios. These diverse approaches empower cybersecurity analysts to effectively analyze and detect cyber threats across various data types and scenarios.

Similarly, in the context of SQL injection attacks, numerous studies have been conducted to develop effective detection and prevention mechanisms. These attacks aim to manipulate SQL queries to gain unauthorized access to a database, leading to potential data breaches and security compromises. Researchers have explored techniques like parameterized queries, stored procedures, and ML algorithms to thwart SQL injection attempts and enhance web application security [35–38]. Notably, three categories of approaches have been explored: ML-based, deep learning-based, and hybrid methods. Leveraging techniques such as support vector machines (SVM), random forests, and decision trees, ML-based approach has shown promise in detecting SQL injection attacks. Extracting features from query structures and contextual information, ML models are trained to classify queries as benign or malicious. Studies [39, 40], and [41] are examples of this approach, where various ML classifiers like logistic regression, AdaBoost, and SVM were employed for detection. Relying on convolutional neural networks (CNNs), recurrent neural networks (RNNs), and hybrid models, deep learning methods effectively capture complex patterns in SQL queries. Noteworthy works, such as [42, 43], and [44], applied LSTM, feed-forward networks, and MLP for detecting SQL injection attacks, demonstrating the potential of deep learning in this context. Combining ML, rule-based analysis, and anomaly detection, hybrid methods aim to enhance detection accuracy while minimizing false positives. Study [45], for instance, introduced a hybrid approach utilizing CNN, MLP, and word embedding to prevent SQL injection attacks effectively. Several related studies have also explored different techniques for SQL injection detection and prevention. These include genetic algorithms, swarm-based intelligence, and program trace techniques [46–48].

While existing works have made significant contributions to the field of web application security, there is still room for improvement. Many studies focus on individual attack types, neglecting the potential benefits of a unified

approach that can detect multiple types of attacks efficiently. Moreover, the use of advanced natural language processing techniques, such as Word2Vec [14], FastText [16], and the USE [15], has been limited in the context of web application security.

In light of these gaps in the literature, our research aims to bridge the existing knowledge by presenting a unified approach that leverages Word2Vec, FastText, and the USE as feature extractors to detect both XSS and SQL injection attacks. By combining these advanced NLP techniques with ML classifiers, this study strives to achieve improved detection accuracy and resilience against evolving attack vectors.

In the subsequent sections, a detailed explanation of the proposed unified detection approach, the methodology employed, the experimental setup, and the evaluation of results were provided.

4 Methodology

4.1 Used Dataset

In this research, three distinct datasets were employed for the experiments. The first dataset, titled "Cross-Site Scripting (XSS) dataset for Deep Learning," was shared by SYED SAQLAIN HUSSAIN SHAH and obtained from the Kaggle repository. It consisted of 13,685 entries sourced from PortSwigger and OWASP Cheat Sheets, providing a diverse collection of web content containing both benign samples and instances of XSS attacks. This dataset served as the foundation for evaluating the XSS attack detection methods, allowing for the assessment of their effectiveness across various attack scenarios and the drawing of meaningful conclusions from rigorous evaluations.

Additionally, the SQL injection dataset shared by Syed Saqlain Hussain Shah on Kaggle, which consists of 33,726 unique SQL queries along with their corresponding labels were utilized. The dataset includes both SQL injection attacks and benign traffic, collected from various websites and subsequently cleaned. Furthermore, the third dataset utilized in this study was the SQL injection dataset shared by SAJID576 on Kaggle, comprising 30,919 unique SQL queries.

To ensure robust evaluations, all three datasets were divided into an 80% training set and a 20% testing set. This partitioning enabled to effectively train the proposed models on a significant portion of the data and validate their performance on unseen instances, providing reliable assessments of the proposed detection techniques.

4.2 Proposed Method

4.2.1 UniEmbed: A Unified Embedding Model

Feature extraction plays a crucial role in detecting XSS and SQL injection attacks due to the unique characteristics of these web application vulnerabilities. XSS and SQL injection attacks involve the injection of malicious code or scripts into web applications, exploiting vulnerabilities in the underlying code and potentially leading to unauthorized access, data theft, and other cyber threats.

Traditional detection methods rely on static and dynamic analysis, which may not effectively capture the intricate patterns and variations present in malicious code. Feature extraction techniques transform raw data, such as text and code, into meaningful representations that can be fed into ML algorithms for analysis. Advanced feature extraction methods like Word2Vec, the USE, and FastText capture semantic relationships, contextual information, and syntactic structures present in the data. This enables the proposed models to discern subtle differences between benign and malicious content, even when the attacks are disguised with various obfuscation techniques. The extracted features act as essential discriminative factors, enhancing the accuracy and robustness of the proposed detection models. Additionally, feature extraction techniques help to reduce the dimensionality of the data, improving computational efficiency and reducing training time for ML classifiers. To enhance the effectiveness of XSS and SQL injection attack detection, a novel hybrid unified method called UniEmbed was devised. This approach leverages the unique strengths of three distinct feature extraction techniques: the sentence-level embedding of the USE, the word-level embedding of Word2Vec, and the character n-gram level embedding of FastText. By combining these complementary representations, a comprehensive view of the input data is captured, including semantic meaning, syntactic structures, and fine-grained character-level information. This feature fusion process creates more robust and expressive feature vectors, effectively representing the intricacies of both benign and malicious content in web applications. The sentence-level representation provided by the Universal Sentence Encoder [15] allows for capturing contextual information and semantic relationships within the data. Word2Vec [14, 14], on the other hand, focuses on word-level representations, encoding the meaning and relationships between individual words. Additionally, the character n-gram level representation from FastText [16] further enhances the ability to capture character-level patterns, useful in identifying potential obfuscation and evasion

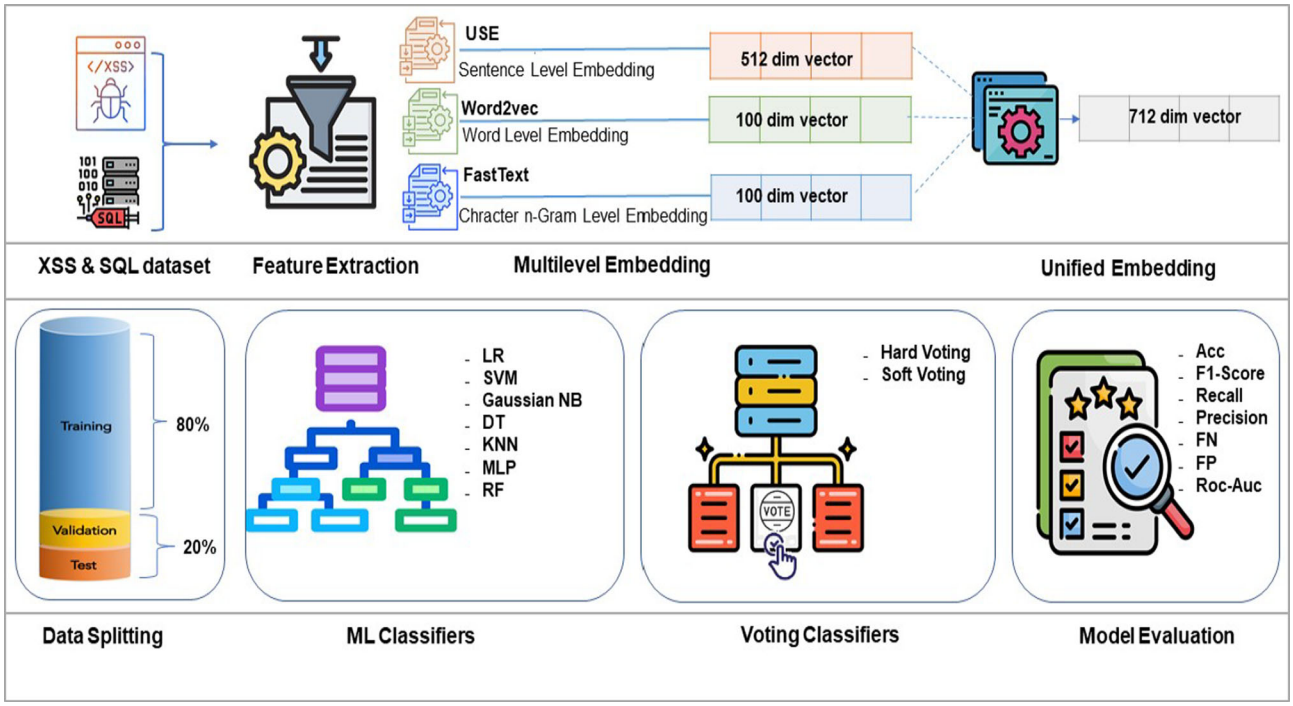

Fig. 1 Block diagram of the proposed method

techniques employed by attackers. By fusing these diverse feature representations, the hybrid unified method takes advantage of the unique strengths of each technique, resulting in a more comprehensive and discriminative feature space. This approach empowers the proposed ML classifiers to make more informed and accurate decisions when distinguishing between normal and malicious content. Figure 1 shows the block diagram of the proposed method. Moreover, the process of extracting features and evaluating machine learning classifiers using the UniEmbed method is outlined in Algorithm 1, which details the steps from data preprocessing and feature extraction to model training, evaluation, and the implementation of hard and soft voting classifiers.

4.3 Classification Approach

In this study, a wide range of ML classifiers were employed, including logistic regression (LR), SVM, Gaussian Naïve Bayes (GNB), decision tree (DT), k-nearest neighbor (KNN), multilayer perceptron (MLP), and random forest (RF). Additionally, both hard voting classifier and soft voting classifier were utilized for comprehensive evaluation and comparison of their performances in detecting XSS and SQL injection attacks.

5 Experimental results

5.1 Experimental Setup and Evaluation Metrics

The experiments took place on a computational setup equipped with an 11th Gen Intel(R) Core(TM) i7-11,700 processor clocked at 2.50GHz and 32 GB of RAM. Python was used for implementing the experiments.

Algorithm 1 The Algorithm for the Proposed Method

Input:
Datasets for XSS and SQL Injection attacks
Output:
- Feature vectors extracted using the UniEmbed method.
- Performance metrics of individual classifiers, hard voting classifier, and soft voting classifier.

Steps:
Step1. Load Pretrained Models:
- Load pretrained Word2Vec, FastText, and USE models.

Step 2. Preprocess Data
For each dataset:
i. Clean the text data (remove HTML tags, special characters, etc.).
ii. Tokenize the cleaned data.

Step 3. Feature Extraction
For each dataset:
→ **3.1. Extract Word2Vec embeddings**:
words = Tokenize(cleaned_data)
word_embeddings = []
For each word in words:
→ If the word exists in the Word2Vec vocabulary:
→ embedding = Word2Vec[word]
→ word_embeddings.append(embedding)
word2vec_representation = Concatenate(word_embeddings) (dimension: 100)
→ **3.2. Extract FastText embeddings**:
fasttext_representation = FastText[cleaned_data] (dimension: 100)
→ **3.3. Extract USE embeddings**:
use_representation = USE [cleaned_data] (dimension: 512)
→ **3.4. Concatenate Word2Vec, FastText, and USE embeddings**:
uniembed_representation = Concatenate (word2vec_representation, fasttext_representation, use_representation)(dimension: 712)
Append uniembed_representation to the feature_vectors list.

Step 4. Split Data
For each combined feature set:
→ Split the dataset into training (80%) and testing (20%) sets.

Step 5. Define Machine Learning Models
Select a set of machine learning classifiers to evaluate.

Step 6. Train and Evaluate Models
For each model in the selected classifiers:
i. Train the model using the training set of the XSS dataset.
ii. Make predictions on the testing set of the XSS dataset.
iii. Evaluate model performance using metrics such as accuracy, precision, recall, and F1-score.
iv. Repeat steps (i) to (iii) for the SQL Injection datasets (both SQLi datasets).

Step 7. Voting Classifiers Implementation
a. **Define Hard and Soft Voting Classifiers**:
i. **Hard Voting Classifier**:
For each instance in the testing set, take the majority vote from all individual classifiers.
ii. **Soft Voting Classifier**:
For each instance in the testing set, average the predicted probabilities from all individual classifiers.
b. **Train the Voting Classifiers**:
i. Train both the hard voting and soft voting classifiers on the training set.
c. **Evaluate the Voting Classifiers**:
i. Evaluate the hard-voting classifier on the testing set (for XSS and SQLi datasets) using accuracy, precision, recall, and F1-score.
ii. Evaluate the soft voting classifier on the testing set (for XSS and SQLi datasets) using accuracy, precision, recall, and F1-score.

To gauge the model's performance and effectiveness, a comprehensive evaluation using various metrics were conducted. These metrics, such as accuracy, precision, recall, and F1-score, offered a thorough understanding of the models' performance. The confusion matrix visually represented true positive, true negative, false positive, and false negative predictions, aiding in identifying biases or imbalances. Moreover, receiver operating characteristic (ROC) curves and area under the curve (AUC) values were utilized to assess classification performance across different thresholds, providing insights into the models' overall performance and their ability to balance true positive and false positive rates.

5.2 Results and Discussion

The results of the ML experiments conducted using the proposed UniEmbed method on the three datasets introduced in this study are presented in Tables 1, 2, and 3. Moreover, Figs. 2, 3, and 4 showcase the ROC curves of ML classifiers utilizing the proposed UniEmbed method across different introduced datasets. On the other hand, Table 4 presents a comparison of accuracy results achieved on the XSS attack dataset using different feature extraction methods.

As evident from the results presented in Tables 1, 2, and 3, for the XSS attack dataset, the MLP classifier exhibits superior performance compared to other ML classifiers across all evaluation metrics. The MLP classifier achieved an impressive accuracy of 99.82% and an AUC of 0.9999, showcasing its ability to effectively discriminate between benign and malicious instances. Furthermore, the MLP classifier demonstrated exceptional precision and recall, with only 3 false positives (FP) and 2 false negatives (FN). Notably, the hard voting classifier also yielded the same outstanding results. This exceptional performance indicates that the proposed UniEmbed method combined with the MLP classifier is highly effective in detecting XSS attacks, making it a promising approach for web application security. Likewise, when dealing with SQL injection attacks (Tables 2 and 3), the MLP classifier demonstrated outstanding performance in both datasets. It achieved an impressive AUC of 0.999 in both cases, coupled with accuracy rates exceeding 99.8%. In the first dataset, there were only 9 false positives (FP) and 3 false negatives (FN), while in the second dataset, the numbers were further reduced to 8 FP and 1 FN. These findings underscore the effectiveness of the MLP classifier in precisely identifying SQL injection attacks while also reducing occurrences of both false positives and false negatives. The ROC curves depicted in Figs. 2, 3, and 4 further corroborate the superior performance of the proposed method in combination with all classifiers across the three datasets. The curves clearly illustrate the ability of the proposed approach to effectively balance the true positive rate and false positive rate for both XSS and SQL injection attack detection. Figure 2 illustrates the ROC curve for the XSS attack dataset, demonstrating the outstanding discrimination capability of the proposed method with all classifiers. The curves are consistently close to the top-left corner, indicating high true positive rates and low false positive rates, which are indicative of strong performance in distinguishing between malicious and benign instances.

Similarly, Fig. 3 displays the ROC curves for the first SQL injection attack dataset. Once again, the curves demonstrate the effectiveness of the proposed method with all classifiers in achieving high true positive rates and low false positive rates. The curves' proximity to the top-left corner validates the approach's ability to accurately detect SQL injection attacks. In Fig. 4, the ROC curves for the second SQL injection attack dataset further reinforce the robustness of the proposed method in combination with all classifiers. The curves exhibit similar characteristics as in the previous figures, reaffirming the consistent and reliable performance of our approach in detecting SQL injection attacks.

In the context of Table 4, it is evident that the UniEmbed method, which incorporates a combination of Word2Vec, Universal Sentence Encoder (USE), and FastText as feature extraction techniques, demonstrated the most remarkable outcome achieved the highest accuracy results, outperforming individual methods and their pairwise combinations. This demonstrates the effectiveness of leveraging multiple feature extraction techniques to enhance the performance of XSS attack detection models. The overall results obtained using all classifiers were exceedingly impressive and promising, demonstrating the effectiveness of the proposed feature extraction method in detecting both XSS and SQL injection attacks. The robustness of the proposed approach was evident across various ML classifiers, showcasing its ability to consistently detect and mitigate both types of attacks with high accuracy and reliability. These compelling results underscore the significance of the chosen feature extraction technique in enhancing the performance of the classifiers for both XSS and SQL injection attacks.

6 Conclusion

This study presented an innovative and unified approach for detecting XSS and SQL injection attacks in web applications. By leveraging the strengths of sentence-level representation from USE, word-level representation from Word2Vec, and character n-gram level representation from FastText, the proposed UniEmbed feature extraction method significantly improved the performance of various ML classifiers. The evaluation results demonstrated outstanding performance across different datasets and classifiers. Specifically, the MLP classifier consistently outperformed other ML classifiers in both XSS and SQL injection attack detection, achieving

Table 1 Performance comparison of ML classifiers on XSS attack dataset

ML Classifier	Accuracy	F1-score	Recall	Precision	FP	FN	ROC-AUC score
LR	0.9938	0.9943	0.9926	0.9959	6	11	0.9999
SVM	0.9890	0.9898	0.9939	0.9858	21	9	0.9997
Gaussian NB	0.9795	0.9812	0.9759	0.9865	20	36	0.9843
DT	0.9967	0.9970	0.9966	0.9973	4	5	0.9966
KNN	0.9974	0.9976	0.9973	0.9980	3	4	0.9992
MLP	0.9982	0.9983	0.9986	0.9980	3	2	0.9999
RF	0.9978	0.9980	0.9986	0.9973	4	2	0.9999
Hard voting classifier	0.9982	0.9983	0.9986	0.9980	3	2	–
Soft voting classifier	0.9978	0.9980	0.9986	0.9973	4	2	–

Table 2 Performance comparison of ML classifiers on the first SQL injection attack dataset

ML Classifier	Accuracy	F1-score	Recall	Precision	FP	FN	ROC-AUC score
LR	0.9970	0.9956	0.9991	0.9921	18	2	0.9994
SVM	0.9963	0.9945	1.0000	0.9890	25	0	0.9985
Gaussian NB	0.9913	0.9871	0.9843	0.9899	23	36	0.9929
DT	0.9950	0.9926	0.9908	0.9943	13	21	0.9949
KNN	0.9970	0.9956	0.9996	0.9917	19	1	0.9973
MLP	0.9982	0.9974	0.9987	0.9961	9	3	0.9990
RF	0.9972	0.9958	0.9991	0.9926	17	2	0.9992
Hard voting classifier	0.9972	0.9958	1.0000	0.9917	19	0	–
Soft voting classifier	0.9970	0.9956	0.9996	0.9917	19	1	–

Table 3 Performance comparison of ML classifiers on the second SQL injection attack dataset

ML Classifier	Accuracy	F1-score	Recall	Precision	FP	FN	ROC-AUC score
LR	0.9966	0.9954	0.9987	0.9921	18	3	0.9992
SVM	0.9960	0.9945	0.9996	0.9895	24	1	0.9995
Gaussian NB	0.9908	0.9875	0.9956	0.9795	47	10	0.9908
DT	0.9937	0.9915	0.9887	0.9943	13	26	0.9938
KNN	0.9969	0.9958	0.9991	0.9926	17	2	0.9985
MLP	0.9985	0.9980	0.9996	0.9965	8	1	0.9990
RF	0.9976	0.9967	0.9987	0.9948	12	3	0.9991
Hard voting classifier	0.9973	0.9963	0.9991	0.9935	15	2	–
Soft voting classifier	0.9977	0.9969	0.9996	0.9943	13	1	–

impressive accuracy and ROC-AUC scores. The low false positive and false negative rates further highlight the reliability of this approach in discriminating between malicious and benign instances. Moreover, the ROC curves showcased the effectiveness of the suggested method with all classifiers, consistently exhibiting high true positive rates and low false positive rates. This demonstrates the approach's ability to effectively balance sensitivity and specificity for both attack types.

The robustness and versatility of this approach make it a valuable tool for real-world application scenarios where swift and accurate attack detection is of paramount importance. Future work should focus on applying transfer learning and domain adaptation to enable the UniEmbed method to generalize across diverse web application environments, integrating with real-time monitoring systems for immediate threat detection and response, and expanding evaluations to include a broader range of datasets and attack scenarios to further assess its performance and adaptability.

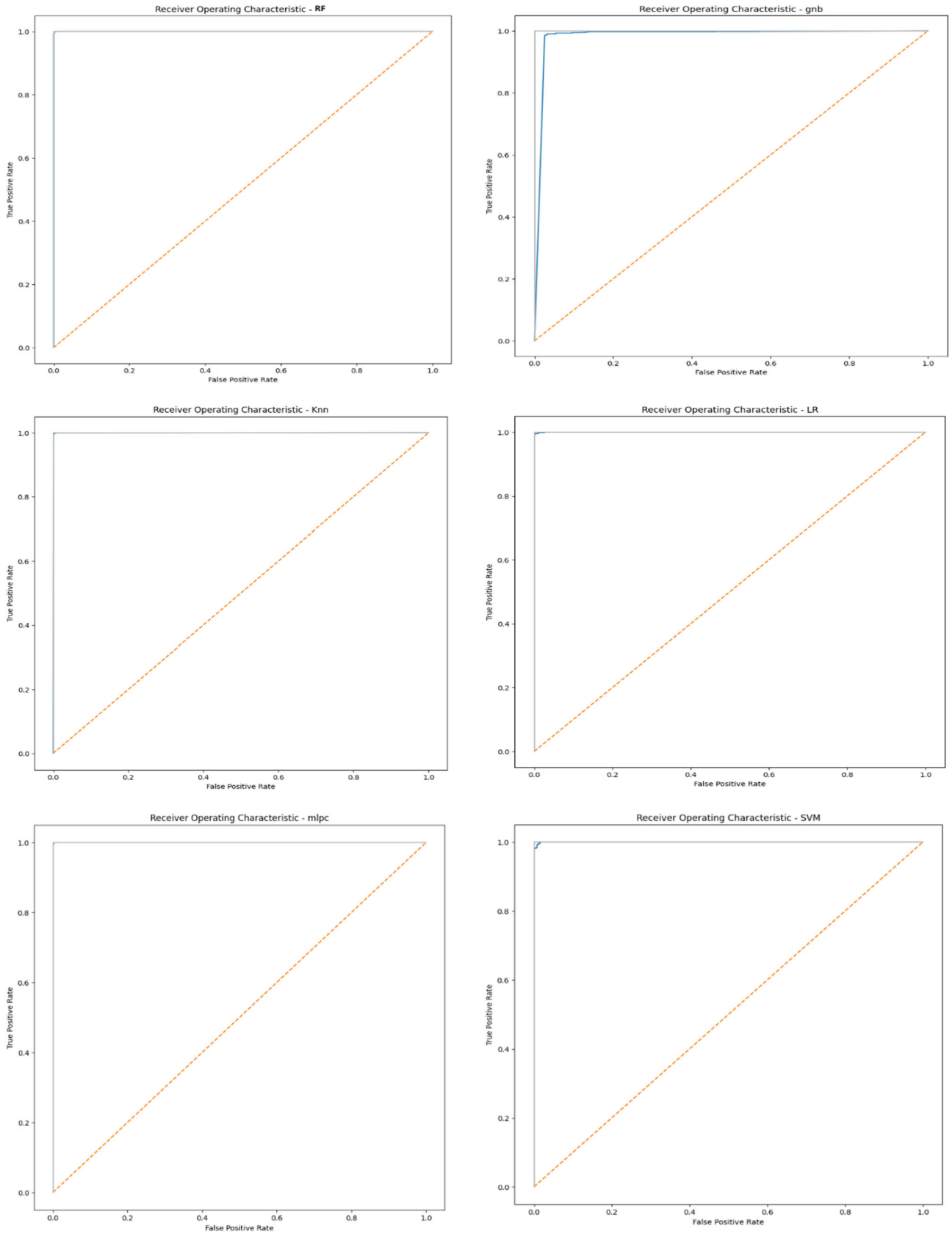

Fig. 2 ROC curves of ML classifiers using UniEmbed method for XSS attack detection

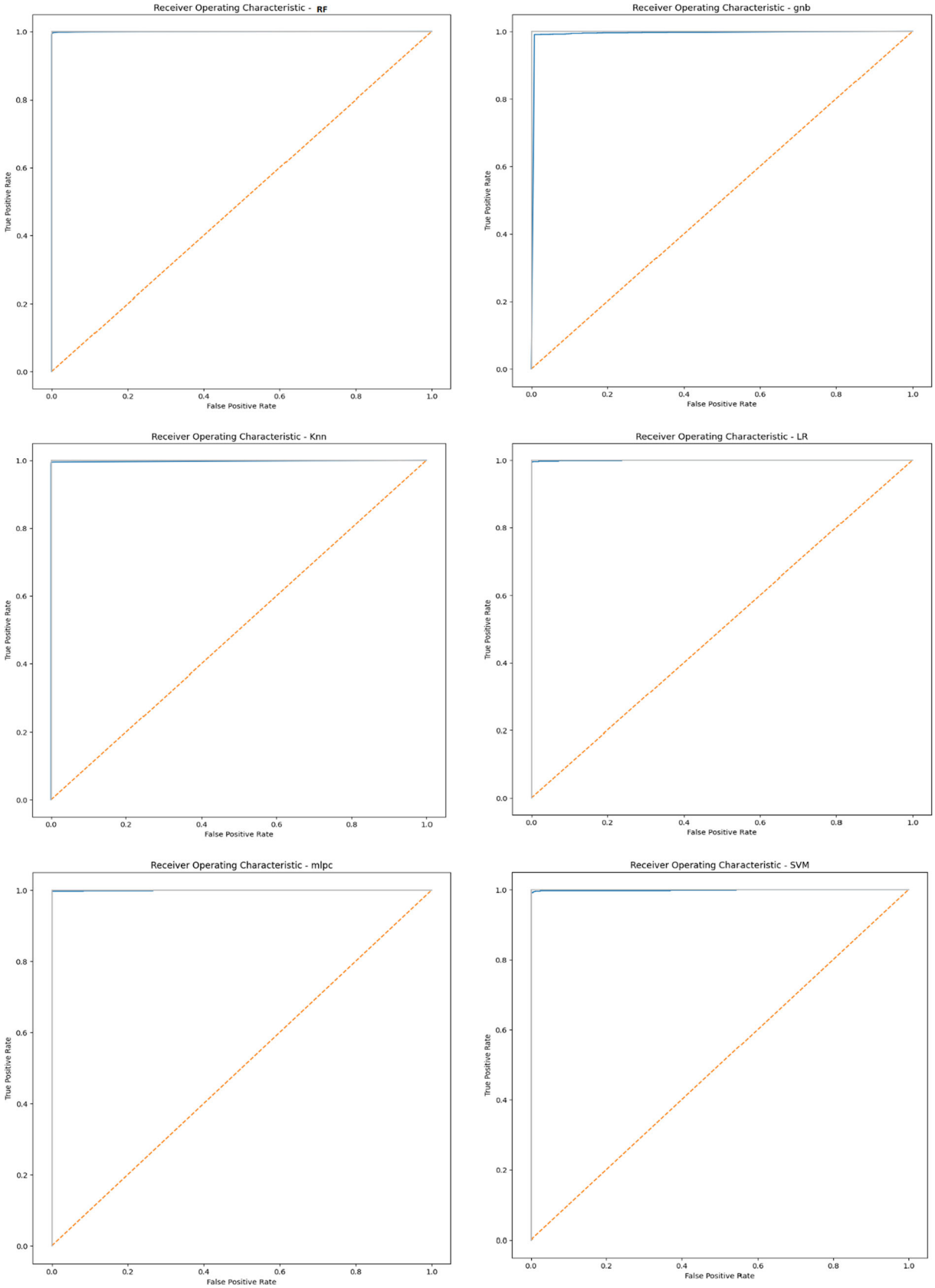

Fig. 3 ROC curves of ML classifiers using UniEmbed method for first SQL injection dataset

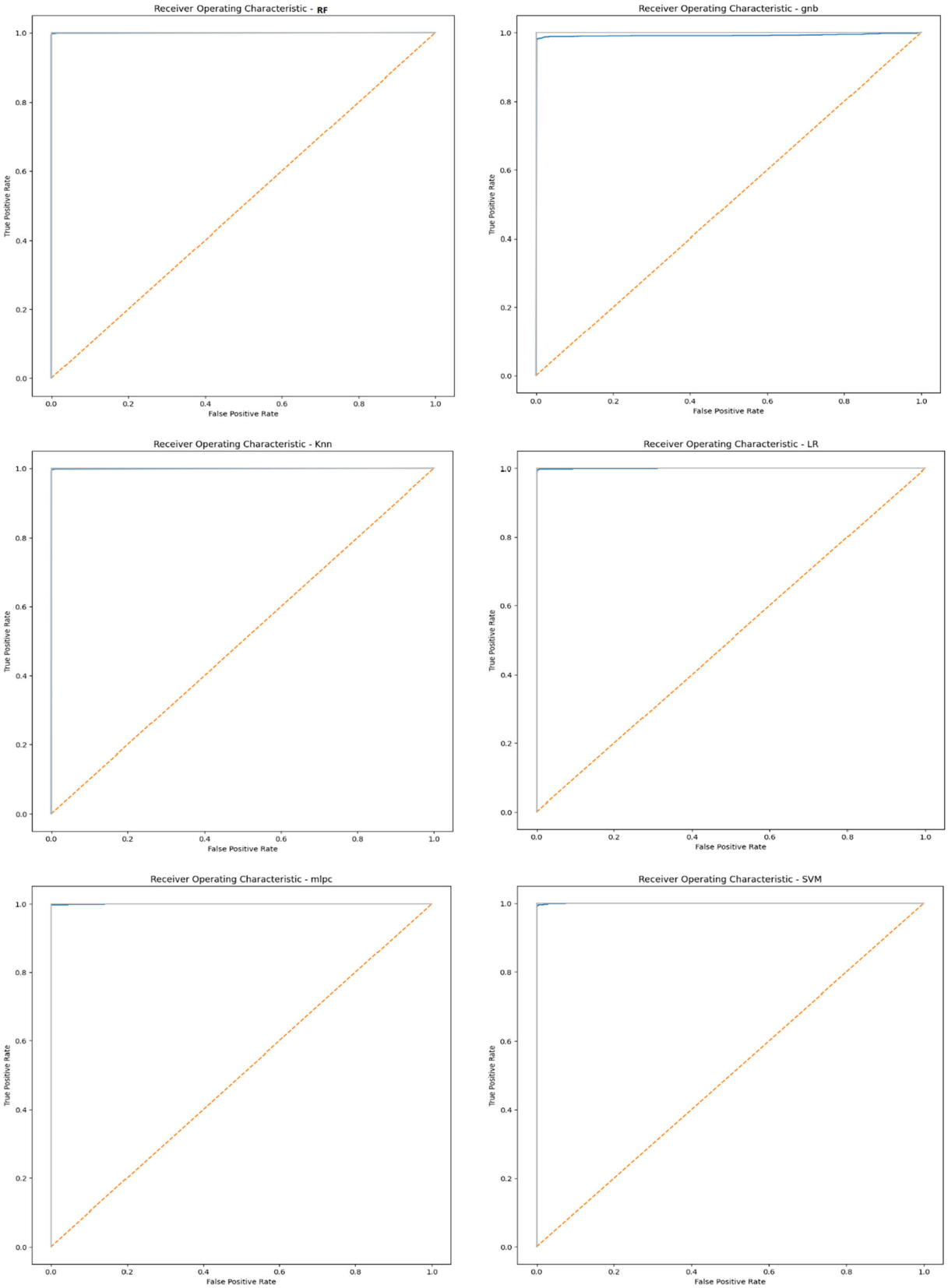

Fig. 4 ROC curves of ML classifiers using UniEmbed method for second SQL injection dataset

Table 4 Comparing accuracy results of various feature extraction methods on XSS attack dataset

ML Classifier	Accuracy (%)			
	USE	Word2vec	USE-word2vec	UniEmbed
LR	96.06	95.11	97.48	99.38
SVM	98.39	95.00	97.99	98.90
GNB	90.17	91.75	93.54	97.95
DT	94.81	97.48	98.25	99.67
KNN	97.92	97.26	98.58	99.74
MLP	98.61	97.04	99.10	99.82
RF	98.10	97.55	99.45	99.78

Acknowledgements The author would like to thank SYED SAQLAIN HUSSAIN SHAH and SAJID576 for sharing their datasets on Kaggle. The datasets used in this research are as follows: Cross-Site Scripting (XSS) Dataset: Link, SQL Injection Dataset by SYED SAQLAIN HUSSAIN SHAH: Link, SQL Injection Dataset by SAJID576: Link

Funding Open access funding provided by the Scientific and Technological Research Council of Türkiye (TÜBİTAK). This research did not receive any specific grants from funding agencies in the public, commercial, or not-for-profit sectors.

Data Availability The data supporting the findings of this study are available upon reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethics Approval and Consent to Participate Not applicable.

Consent for Publication Not applicable.

References

1. Nair, S.S.: Securing against advanced cyber threats: a comprehensive guide to phishing, XSS, and SQL injection defense. J Computer Sci Technol Stud **6**(1), 76–93 (2024)
2. Oxana A.: Top 10 web application vulnerabilities in 2021–2023. Securelist, [Online]. Available: https://securelist.com/top-10-web-app-vulnerabilities/112144/ (2023) Accessed: 11 Oct 2024
3. Odeh, A.; Taleb, A.A.: XSSer: hybrid deep learning for enhanced cross-site scripting detection. Bull Electrical Eng Inf **13**(5), 3317–3325 (2024)
4. Gudipati, VK.; Venna, T.; Subburaj, S.: Abuzaghleh, O.: Advanced automated SQL injection attacks and defensive mechanisms. In: 2016 Annual Connecticut Conference on Industrial Electronics, Technology & Automation (CT-IETA), IEEE, 1–6 2016
5. Okoli, U.I.; Obi, O.C.; Adewusi, A.O.; Abrahams, T.O.: Machine learning in cybersecurity: A review of threat detection and defense mechanisms. World J Adv Res Rev **21**(1), 2286–2295 (2024)
6. Ismail, M.; Alrabaee, S.; Choo, K.-K.R.; Ali, L.; Harous, S.: A comprehensive evaluation of machine learning algorithms for web application attack detection with knowledge graph integration. Mobile Netw Appl (2024). https://doi.org/10.1007/s11036-024-02367-z
7. Venkatramulu, S.; Waseem, M.S.; Taneem, A.; Thoutam, S.Y.; Apuri, S.: Research on SQL injection attacks using word embedding techniques and machine learning. J Sensors, IoT Health Sci **2**(01), 55–66 (2024)
8. Mustapha, AA.; Udeh, AS.; Ashi, TA.; Sobowale, OS.; Akinwande, MJ.; Oteniara, AO.: Comprehensive review of machine learning models for sql injection detection in e-commerce (2024).
9. Arasteh, B.; Aghaei, B.; Farzad, B.; Arasteh, K.; Kiani, F.; Torkamanian-Afshar, M.: Detecting SQL injection attacks by binary gray wolf optimizer and machine learning algorithms. Neural Comput. Appl. **36**(12), 6771–6792 (2024)
10. Taborda Echeverri, S.: Evaluation of SQL injection (SQLi) attack detection strategies in web applications using machine learning. Industry semester (2024).
11. Alhamyani, R.; Alshammari, M.: Machine learning-driven detection of cross-site scripting attacks. Information **15**(7), 420 (2024)
12. Njie, B.; Gabriouet, L.: Machine learning for cross-site scripting (XSS) Detection: A comparative analysis of machine learning models for enhanced XSS detection (2024).
13. Kshetri, N.; Kumar, D.; Hutson, J.; Kaur, N.; Osama, OF.: algo-XSSF: Detection and analysis of cross-site request forgery (XSRF) and cross-site scripting (XSS) attacks via Machine learning algorithms. In: 2024 12th International Symposium on Digital Forensics and Security (ISDFS), IEEE, 1–8 (2024)
14. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.
15. Cer, D. et al. Universal sentence encoder. *arXiv preprint* arXiv:1803.11175, (2018).
16. Joulin, A.; Grave, E.; Bojanowski, P.; Mikolov, T.: Bag of tricks for efficient text classification. *arXiv preprint* arXiv:1607.01759, (2016).

17. Kumar, A.; Dutta, S.; Pranav, P.: Analysis of SQL injection attacks in the cloud and in WEB applications. Security Privacy **7**(3), e370 (2024)
18. Sarmah, U.; Bhattacharyya, D.K.; Kalita, J.K.: A survey of detection methods for XSS attacks. J. Netw. Comput. Appl. **118**, 113–143 (2018)
19. Kirda, E.; Kruegel, C.; Vigna, G.; Jovanovic, N.: Noxes: a client-side solution for mitigating cross-site scripting attacks. In Proceedings of the 2006 ACM symposium on Applied computing, 2006, pp. 330–337.
20. Abikoye, O.C.; Abubakar, A.; Dokoro, A.H.; Akande, O.N.; Kayode, A.A.: A novel technique to prevent SQL injection and cross-site scripting attacks using Knuth-Morris-Pratt string match algorithm. EURASIP J. Inf. Secur. **2020**, 1–14 (2020)
21. Wurzinger, P.; Platzer, C.; Ludl, C.; Kirda, E.; Kruegel, C.: SWAP: Mitigating XSS attacks using a reverse proxy. In 2009 ICSE Workshop on Software Engineering for Secure Systems, IEEE, 33–39 (2009)
22. Goswami, S.; Hoque, N.; Bhattacharyya, D.K.; Kalita, J.: An Unsupervised method for detection of XSS attack. Int. J. Netw. Secur. **19**(5), 761–775 (2017)
23. Fang, Y.; Huang, C.; Xu, Y.; Li, Y.: RLXSS: Optimizing XSS detection model to defend against adversarial attacks based on reinforcement learning. Future Internet **11**(8), 177 (2019)
24. Alqarni, A.A.; Alsharif, N.; Khan, N.A.; Georgieva, L.; Pardade, E.; Alzahrani, M.Y.: "MNN-XSS: modular neural network based approach for XSS attack detection. Computers, Mater Continua **70**(2), 4075 (2022)
25. Bakour, K.; Daş, GS.; Ünver, HM.: An intrusion detection system based on a hybrid Tabu-genetic algorithm. In: 2017 International Conference on Computer Science and Engineering (UBMK), Ieee, 215–220 (2017)
26. Kumar, P.P.; Jaya, T.; Rajendran, V.: SI-BBA–A novel phishing website detection based on Swarm intelligence with deep learning. Mater Today Proc **80**, 3129–3139 (2023)
27. Doğan, E.; BAKIR, H.: Hiperparemetreleri Ayarlanmış Makine Öğrenmesi Yöntemleri Kullanılarak Ağdaki Saldırıların Tespiti. In: International Conference on Pioneer and Innovative Studies, 274–286 (2023)
28. Bakır, H.; Bakır, R.: DroidEncoder: Malware detection using autoencoder based feature extractor and machine learning algorithms. Comput. Electr. Eng. **110**, 108804 (2023)
29. Ünver, H.M.; Bakour, K.: Android malware detection based on image-based features and machine learning techniques. SN Appl Sci **2**, 1–15 (2020)
30. Bakour, K.; Ünver, H.M.: DeepVisDroid: android malware detection by hybridizing image-based features with deep learning techniques. Neural Comput. Appl. **33**, 11499–11516 (2021)
31. Ghanem, R.; Erbay, H.; Bakour, K.: Contents-Based Spam Detection on Social Networks Using RoBERTa Embedding and Stacked BLSTM. SN Comput Sci **4**(4), 380 (2023)
32. Ghanem, R.; Erbay, H.: Spam detection on social networks using deep contextualized word representation. Multimed Tools Appl **82**(3), 3697–3712 (2023)
33. Bakır, R.; Erbay, H.; Bakır, H.: ALBERT4Spam: a novel approach for spam detection on social networks. Bilişim Teknolojileri Dergisi **17**(2), 81–94 (2024)
34. Ghanem, R.; Erbay, H.: Context-dependent model for spam detection on social networks. SN Appl Sci **2**, 1–8 (2020)
35. Joshi, A.; Geetha, V.: SQL Injection detection using machine learning. In: 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), IEEE, 1111–1115 (2014)
36. Ahmad, K.; Karim, M.: A method to prevent SQL injection attack using an improved parameterized stored procedure. Int J Adv Comput Sci Appl (2021). https://doi.org/10.14569/IJACSA.2021.0120636
37. Jemal, I.; Cheikhrouhou, O.; Hamam, H.; Mahfoudhi, A.: Sql injection attack detection and prevention techniques using machine learning. Int. J. Appl. Eng. Res. **15**(6), 569–580 (2020)
38. Katole, RA.; Sherekar, SS.; Thakare, VM.: 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, 736–741 (2018)
39. Hasan, M.; Balbahaith, Z.; Tarique, M.: Detection of SQL injection attacks: a machine learning approach. In: 2019 International Conference on Electrical and Computing Technologies and Applications, ICECTA 2019, (2019) https://doi.org/10.1109/ICECTA48151.2019.8959617.
40. Roy, P.; Kumar, R.; Rani, P.: SQL injection attack detection by machine learning classifier. In: Proceedings—International Conference on Applied Artificial Intelligence and Computing, ICAAIC 2022, Institute of Electrical and Electronics Engineers Inc., 394–400. (2022) https://doi.org/10.1109/ICAAIC53929.2022.9792964.
41. Uwagbole, SO.; Buchanan, WJ.; Fan, L.: Applied Machine Learning predictive analytics to SQL Injection Attack detection and prevention. In: Proceedings of the IM 2017 - 2017 IFIP/IEEE International Symposium on Integrated Network and Service Management. https://doi.org/10.23919/INM.2017.7987433 (2017)
42. Li, Q.; Wang, F.; Wang, J.; Li, W.: LSTM-based SQL injection detection method for intelligent transportation system. IEEE Trans Veh Technol (2019). https://doi.org/10.1109/TVT.2019.2893675
43. Hassan, M.M.; Badlishah Ahmad, R.; Ghosh, T.: SQL injection vulnerability detection using deep learning: a feature-based approach. Indonesian J Electrical Eng Inf (IJEEI) **9**(3), 702–718 (2021)
44. Tang, P.; Qiu, W.; Huang, Z.; Lian, H.; Liu, G.: Detection of SQL injection based on artificial neural network. 190,105528, (2020)
45. Chen, D.; Yan, Q.; Wu, C.; Zhao, J.: SQL injection attack detection and prevention techniques using deep learning. J Phys: Conf Series IOP Publishing Ltd (2021). https://doi.org/10.1088/1742-6596/1757/1/012055
46. Basta, C.; Darwish, S.: Detection of SQL injection using a genetic fuzzy classifier system. Int J Adv Comput Sci Appl (2016). https://doi.org/10.14569/IJACSA.2016.070616
47. Baptista, K.; Bernardino, E.; Bernardino, A.: Swarm Intelligence applied to SQL Injection. In: 2022 17th Iberian Conference on Information Systems and Technologies (CISTI), IEEE, 1–6 (2022)
48. Bahruddin, H.; Suryani, V.; Wardana, AA.: Adversary Simulation of Structured Query Language (SQL) Injection Attack Using Genetic Algorithm for Web Application Firewalls (WAF) Bypass. In: Proceedings of SAI Intelligent Systems Conference, Springer 656–669 (2023)