International Journal of Safety and Security Engineering
Vol. 15, No. 9, September, 2025, pp. 1961-1970
Journal homepage: http://iieta.org/journals/ijsse

Hybrid Deep Learning Framework via Early Feature Fusion for XSS Attacks Detection

Ali Nafea Yousif[1,2*], Ziyad Tariq Mustafa Al-Ta'i[1]

[1] Department of Computer Science, College of Science, University of Diyala, Baqubah 32001, Iraq
[2] University of Information Technology and Communications, Baghdad 10001, Iraq

Corresponding Author Email: scicomphd232404@uodiyala.edu.iq

https://doi.org/10.18280/ijsse.150919

Received: 24 August 2025
Revised: 24 September 2025
Accepted: 27 September 2025
Available online: 30 September 2025

Keywords:
Early Feature Fusion, recurrent neural networks, web application security, cross-site scripting detection

ABSTRACT

Cross-site scripting (XSS) is still a significant security risk for online applications because it frequently avoids detection by using inventive payload formats and obfuscation. Most earlier studies either process statistical or sequential data alone or combine them at a later stage, which makes it difficult to capture their complementary interactions. This work differs from earlier approaches by integrating statistical and sequential representations at input stage, allowing both feature types to be learned jointly rather than fused later. Aiming to overcome this gap, this study suggests a hybrid deep learning framework that combines statistical features based on Term Frequency-Inverse Document Frequency (TF-IDF) with sequential dependencies learned through Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU) networks using Early Feature Fusion. Principal Component Analysis (PCA) is used to reduce dimensionality in the TF-IDF dimension to enhance generalization. The proposed models have been evaluated on two well-known XSS datasets depending on eight key parameters. Both datasets, GRU and TF-IDF, performed well, with accuracy and F1-score exceeding 99%. The results indicate that early statistical and sequential feature fusion enhances the model's effectiveness in detecting malicious XSS payloads across the evaluated datasets.

1. INTRODUCTION

The extensive use of the Internet, especially web applications and services, has become an integral part of daily life due to the development of dynamic websites that have simplified users' interaction processes. Unfortunately, these websites have become a threat or vulnerable, which have a significant impact on online security and allows attackers to target internet-connected devices [1].

With these potential threats, cybersecurity has become vital on a wide scale in social, industrial and economic contexts, as developers and cybersecurity professionals recognize the possibility of trained attackers launching complex and mysterious attacks [2, 3].

Cross-site scripting (XSS) attacks are one of the most methods leveraged by attackers to inject malicious scripts into legitimate websites to steal information, gain unauthorized access, or perform other illegal activities [4].

Although there are comprehensive strategies for mitigating XSS attacks (which have included integrating robust security controls, regular security audits, and classic detection methods such as signature-based detection), skilled or sophisticated attackers often evade detection, compromising the security of online applications [5]; where payloads containing specialized code or deceptive HTML tags can bypass traditional filters.

Consequently, the evolution in artificial intelligence and machine learning enhances online application security that can promptly identify dynamically evolving attack patterns [6, 7]. Based on vast amounts of data, these technologies offer a strong and proactive security system that can recognize hidden patterns and criminal strategies [8].

Unlike traditional detection frameworks, ML- and DL-based systems can learn from the evolving behavior of dynamic threats and continuously improve their detection performance [9].

The methods of deep learning of the semantic and structural representation of XSS payloads have demonstrated efficient performance through the essential metric results [10]; the detection rate of XSS attacks has significantly increased due to the application of these techniques [11]. Combining machine learning and deep learning techniques, which use artificial intelligence to scan web content, find patterns, and search for potentially harmful code [12], such as malicious JavaScript code or input validation issues, has been shown to achieve better detection of XSS attacks [13].

Through constant learning of the emerging attack trends and evolving threat patterns, AI-powered XSS detection systems can enhance proactive defense mechanisms [14], thereby strengthening defenses against the common and emerging threats [15].

Despite significant progress in applying deep learning and machine learning techniques for XSS attack detection, many methods proposed so far continue to process symbolic features, such as Term Frequency-Inverse Document Frequency (TF-IDF) and sequential forms, such as token embeddings, separately. Separate processing of these prevents models from fully benefiting from the promised synergies of combining frequency-based statistical features with sequential contextual patterns. Previous works have rarely addressed early fusion methods that merge the two complementary perspectives, leaving a significant research gap and limiting the development of more robust and general detection systems.

In addition to these limitations, several XSS payloads used in modern attacks incorporate varying layers of structural manipulation, encoding, and context-dependent behavior, which makes it increasingly difficult for models relying on a single feature perspective to generalize across different obfuscation styles. This highlights the need for detection approaches that can capture both statistical and contextual cues in a unified manner.

Addressing these features separately limits the model's ability to detect payloads that combine structural manipulation with subtle character-level signals, which is a common approach in modern obfuscation techniques. The effectiveness of current fusion techniques to capture joint patterns suffers since they usually combine information after each branch has learned independently. By allowing statistical and sequential representations to be trained simultaneously from the input stage, Early Feature Fusion directly closes this gap and enables the model to recognize complementing inputs that each representation can capture on its own.

This research paper proposes an early fusion strategy, that constantly combines contextual and statistical features to make models to comprehend the syntactic and semantic nature of XSS attacks and categorize them more precisely and reliably.

The primary contributions of this research toward XSS attack detection are as follows:

• Feature Fusion Architecture: A preliminary fusion method is presented that combines statistical features obtained through TF-IDF with temporal representations that are learned by deep recurrent networks (LSTM or GRU); the fusions obtain both syntactic frequency-based patterns and contextual dependencies of XSS payloads.

• Reducing Dimensionality Using Principal Component Analysis (PCA): In high-dimensional statistical representations, for example, those based on TF-IDF, the multiplicity of dimensions can affect model generalization and potentially increase computational costs. Therefore, employing PCA as an approach to reducing the dimensions for common and redundant features enables the model to reduce computational costs, increase learning efficiency, and decrease the likelihood of overfitting, especially in fusion-based architectures.

• Performance Assessment: Multiple variants of the suggested architecture are trained and tested. Performance is assessed using a variety of classification metrics, including accuracy, precision, recall, and F1-score.

The rest of the research comes under Section 2, is dependent upon the investigation of comparative literature. Section 3 describes our proposed research approach. Findings from the method used in Section 4. Discussion in section 5. Finally, the conclusion in Section 6.

2. RELATED WORKS

This part presents an analytical summary of the earlier related studies. To present the existing research more clearly, the reviewed studies can be grouped into three methodological categories. The first group includes classical machine-learning models that rely mainly on statistical or handcrafted features such as TF-IDF, n-grams, or similarity patterns. The second group consists of deep learning approaches, including LSTM, Gated Recurrent Unit (GRU), CNN, and attention-based architectures that capture sequential or semantic structures. The third group includes hybrid and fusion-based approaches that combine multiple feature types, although most of them apply fusion only at a later stage. This categorization helps clarify the methodological direction of each study and highlights where the proposed early-fusion model fits within the broader research landscape. All the selected works are reviewed according to employed detection techniques, assessment datasets, and employed performance measures. The motivation for conducting this analysis is to identify the strengths, weaknesses, and limitations of the existing literature. This also provides a foundation for the models proposed in this study. A comparative summary of this section is presented in Table 1.

Recent developments in XSS vulnerability detection research have demonstrated a variety of models that take advantage of deep learning, statistical techniques, hybrid architecture, and feature integration strategies. These approaches can be categorized based on their basic methodological orientations: evolutionary systems, dimensionality reduction using deep learning, syntactic/semantic analysis, and feature integration.

One direction focuses on the use of evolutionary or adaptive mechanisms to enhance detection robustness. This method is demonstrated by GeneMiner [16], a two-level system in which GeneMiner-C does classification and GeneMiner-E uses a genetic algorithm for feature extraction. It demonstrated 98.5% accuracy in adapting to malicious payloads after training on more than 160,000 obfuscated samples. However, its wider applicability is limited by the lack of an explicit comparison with deep learning techniques or practical assessments.

Other studies focus on the incorporation of dimensionality reduction and deep learning models. The hybrid approach outlined in the study by Stiawan et al. [17] improves with a 97.00% F1-score on the RAMA dataset and 96.85% detection performance accuracy, the detection of SQLi and XSS assaults by combining LSTM and PCA. Similarly, Oshoiribhor and John-Otumu [18] employed PCA over TF-IDF and n-gram features before classification using logistic regression, resulting in a 99.67% F1-score and 99.70% accuracy. Although these efforts show advances in efficiency, they frequently lack comparisons with deep models or assessments against complex adversarial inputs.

A separate class of studies incorporates semantic or syntactic structure into the detection process. In the study by Tan et al. [19], the PATS model transforms JavaScript code into AST paths processed by attention layers. Although it recorded 90.25% accuracy and an 81.62% F1-score, its limited coverage of XSS and long training times hinder deployment. Likewise, Li et al. [20] proposed a character-level BiLSTM with multi-head attention suited for dynamic environments, achieving a 98.71% F1-score and 99.32% precision, but it lacked robustness testing on public datasets. Additionally, Bakır and Bakır [21] suggested semantic hybridization using Word2Vec and Universal Sentence Encoder, which was verified across multiple classifiers. The RF variant achieved a 99.49% F1-score and 99.45% accuracy, although there was no adversarial validation.

Many models that combine CNNs, LSTMs, and attention mechanisms have surfaced in the field of deep feature integration. The work in the study by Luu et al. [22]

(XSShield) was utilizing a dataset of 107,406 samples and integrating hand-engineered features with deep models (CNN+LSTM+Optuna), which produced an impressively low false positive rate of 0.06% and an accuracy of 99.27%. Over a 66,762-sample dataset, another model in the study by Hu et al. [23] merged with DSCNN, BiLSTM, and attention layers, reaching 99.87% accuracy and a 99.92% F1-score, which established a standard in feature fusion despite untested deploy ability. Similarly, Alhamyani and Alshammari [24] evaluated conventional and deep classifiers over 138,569 cases, with MLP achieving 99.76% accuracy and recall, providing an outstanding baseline but missing insight into model generalization or fusion approaches.

Younas et al. [25] produced LSTF, a TF-IDF-LSTM-based extractor that was evaluated on 13,686 Kaggle data; the Random Forest classifier achieved 99.00% across all major criteria, supported by explainability approaches (XAI), but lacked a comprehensive evaluation. At the same time, Farea et al. [26] showed a BiLSTM system that performed multiclass classification (XSS, SQLi, benign) with 99.26% accuracy and a 99.24% F1-score; however, scalability was limited because dimensionality reduction and feature fusion were not used.

Although these studies exhibit outstanding results, most techniques still treat statistical and sequential data individually, which limits their ability to discover complimentary sequences that appear in obfuscated payloads. In particular, by merging information only at the classification stage, late-fusion models avoid early interaction between TF-IDF signals and sequence-level dependencies; this split leads to weaker generality, especially when payloads employ a range of processing strategies. The absence of feature-level integration is a major drawback of the existing literature.

While reviewed methodologies perform well, the majority of approaches evaluate statistical and sequential data independently or wait until the classification stage to integrate them. Furthermore, even though dimensionality reduction techniques such as PCA are effective in lowering training complexity and improving generalization, fusion-based deep learning approaches ignore them.

To overcome these gaps, this study proposes a feature-level early fusion architecture that utilizes TF-IDF representations with LSTM/GRU-based sequence learning. To improve statistical parameters, PCA is used. Eight performance indicators are used to thoroughly evaluate this approach on two benchmark XSS datasets, enabling a fair and comprehensible comparison under various conditions.

A key factor in XSS detection is the time of feature fusion. Late integration strategies limit the ability of models to catch frequent patterns during training since feature branches are created after independent learning.

Frequencies of abnormal terms and structural variations are common components of XSS payloads, requiring the study of both statistical and sequential elements. These interactions can be acquired at the input stage via early integration, allowing the model to integrate sequence behavior and TF-IDF distributions into a single representation.

Previous studies in the field of malicious script detection show that early integration significantly increases anti-obfuscation efficiency and performance consistency across datasets. The early integration technique proposed in this paper is supported by both experimental and theoretical data.

Table 1. The summary of related studies with its datasets, and evaluation metrics

Ref.	Year	Proposed Technique	Dataset	Accuracy (%)	F1-Score (%)	Recall (%)	Precision (%)
[16]	2022	GeneMiner (Genetic Algo. + Classifier)	Generated (160,264 samples)	98.50	98.03	96.25	97.99
[17]	2023	LSTM + PCA (Ensemble)	RAMA Repository	96.85	97.00	97.00	97.00
[18]	2025	Logistic Regression (with TF-IDF + PCA features)	Kaggle XSS (13,686 samples)	99.70	99.67	100.00	99.36
[19]	2023	PATS (AST + Attention)	NIST + GitHub (11,000 samples)	90.25	81.62	-	-
[20]	2023	BiLSTM + Multihead Attention	Cloud-based (105,470 samples)	Not reported	98.71	98.11	99.32
[21]	2025	RF (with USE + Word2Vec features)	13,686 samples (OWASP + PortSwigger)	99.45	99.49	99.26	99.73
[22]	2024	CNN + LSTM	XSSed + Cisco (107,406 samples)	99.27	97.59	95.61	99.65
[23]	2025	DSCNN + BiLSTM + Multihead Attention	XSSed + PortSwigger (66,762 samples)	99.87	99.92	99.89	99.95
[24]	2024	MLP (with RF, SVM, XGBoost, CNN compared)	Real (138,569 samples)	99.76	99.76	99.77	99.74
[25]	2024	Random Forest (with LSTM + TF-IDF features, XAI used)	Kaggle (13,686 samples)	99.00	99.50	99.00	100.00
[26]	2023	BiLSTM	3-class (17,478 samples)	99.26	99.24	99.25	99.26
Our proposed system 2025		**GRU + TFIDF**	Kaggle XSS (13,686 samples) DMOZ and XSSed databases (64,833 samples)	**99.85** **99.12**	**99.86** **99.14**	**99.96** **98.50**	**99.77** **99.80**

3. THE METHODOLOGY OF THE PROPOSED SYSTEM

The following section describes the methodological approach used in the present study to detect XSS attacks via an Early Feature Fusion procedure based on two publicly available datasets. Providing almost all necessary processes, such as data preparation and preprocessing, feature extraction and dimensionality reduction, fusion, model training, and evaluation, to ensure that the findings were reliable and generalizable.

3.1 Proposed feature fusion approach

The present work proposes an efficient technique for Early Feature Fusion that incorporates both syntactic (statistical) and sequential characteristics of payloads at the input level. The framework combines the statistical text representation approach TF-IDF and sequential modeling techniques LSTM/GRU in a unified design, with PCA used as an auxiliary component to strengthen the features and enhance performance. Figure 1 illustrates the architectural structure of the proposed model.

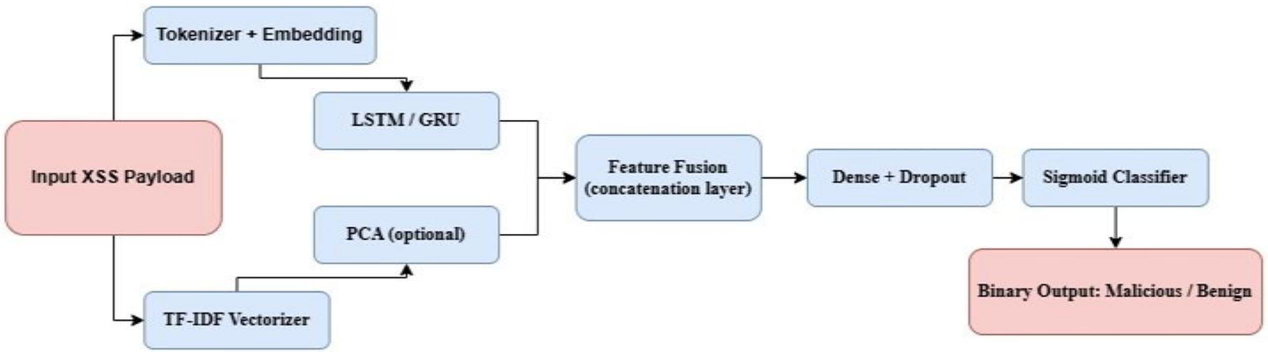

Figure 1. The proposed architecture for XSS attack detection

3.2 Dataset description

Two benchmark datasets were employed; the first dataset, referred to as Dataset_1, is available on Kaggle, as cited in the study by Shah [27]. The second dataset, referred to as Dataset_2, as cited in the study by Fang et al. [28] and is available on GitHub.

Dataset_1 contains 13,686 samples, with 7,373 of the samples being malicious and 6,313 being benign. The Dataset_2 includes 64,833 samples that are distributed with 31,407 benign and 33,426 malicious samples, collected from the DMOZ and XSSed databases. Each entry comprises a single payload string labeled as either benign or malicious. Both datasets underwent stratified splitting (70% training, 30% testing) to maintain class balance.

3.3 Dataset and preprocessing

The experiments are conducted on a labeled dataset of XSS payloads, where each sample is labeled as either malicious or benign. The text samples are preprocessed in the standard manner with tokenization and padding of sequences to a uniform length of 100 tokens. The Keras Tokenizer was employed to convert the text strings into numeric sequences prior to padding. The dataset is classified into training and test sets in the ratio 70:30, and TF-IDF vectors are constructed with a maximum of 1,000 features. The same workflow was performed on both datasets separately to ensure generalizability, with results saved for each one separately.

3.4 Exploratory data analysis (EDA)

The structure of the used Datasets 1 and 2 was visualized by the use of exploratory data analysis (EDA); each contain two classes, which "Label" column for the labeled class and a "Sentence" column for payload, as shown in Figures 2 and 3. The stratified segmentation of the training and testing is presented in Figures 4 and 5.

Malicious scripts had more tokens per sample than benign scripts, according to a token-level analysis, suggesting greater structural complexity. Furthermore, a word cloud of malicious scripts displayed the word frequency plot, as seen in Figures 6 and 7; the most common tokens were HTML elements, script-related phrases, and obfuscation characters that are commonly used in obfuscation.

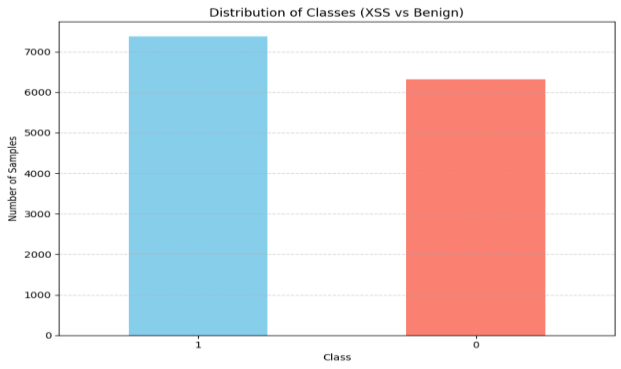

Figure 2. The distribution of Dataset_1 classes

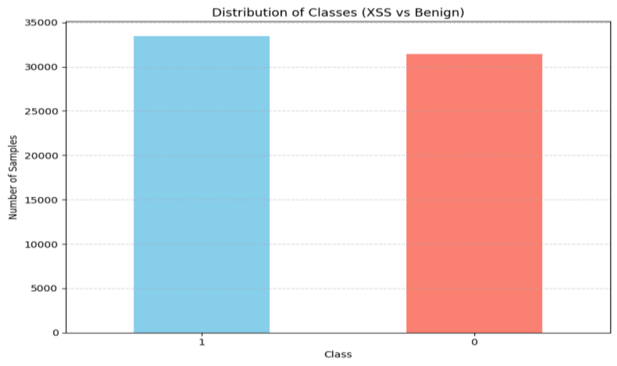

Figure 3. The distribution of Dataset_2 classes

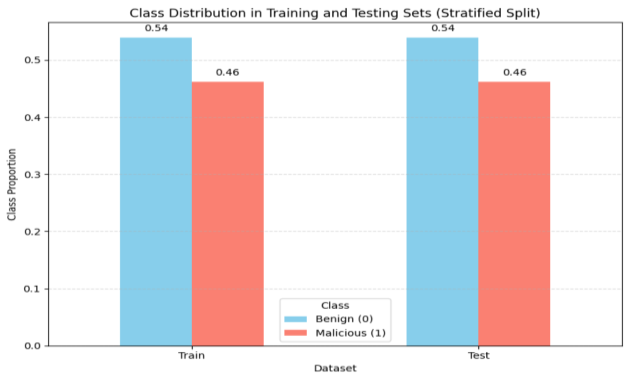

Figure 4. The stratified split Dataset_1

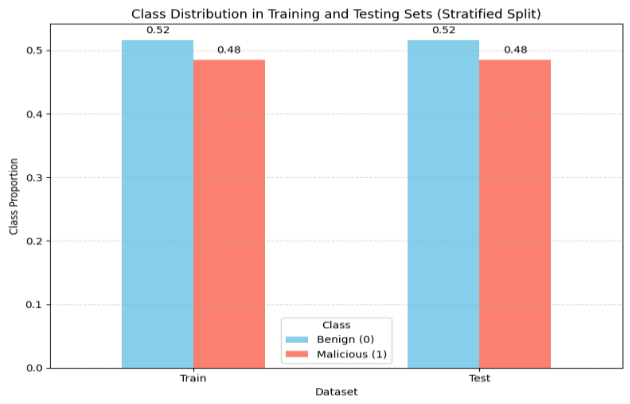

Figure 5. The stratified split Dataset_2

The results validate that in order to accurately model the malware patterns in XSS payloads, syntactic and sequential features must be combined.

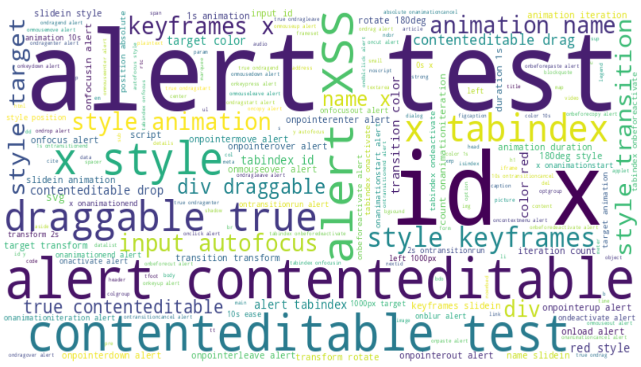

Figure 6. Word frequency of Dataset_1

Figure 7. Word frequency of Dataset_2

3.5 Feature extraction and early fusion

3.5.1 TF-IDF statistical features

TF-IDF was employed to convert payload texts into numerical feature vectors based on word-level statistics. A vocabulary of the top 1000 features was selected, and the resulting sparse matrix served as the statistical representation.

3.5.2 Token-based sequential features

Each payload was tokenized and padded to a fixed sequence length (100 tokens). These sequences were then embedded using an embedding layer and fed into deep learning models (LSTM and GRU) to capture temporal dependencies.

Algorithm 1: Step-by-step process of generating the hybrid feature set through the combination of TF-IDF and RNN-based features based on early fusion.

Algorithm 1: Our Proposed System Algorithm

```
Input: Preprocessed cross-site script data X_d
Output: Hybrid feature set H_features for model training
1: Initialize process
2: F_tfidf ← TFIDF_Vectorizer(X_d)
3: S_token ← Tokenizer(X_d)
4: F_seq ← Embedding(S_token)
5: F_rnn ← GRU or LSTM _Model(F_seq)
6: F_pca ← PCA(F_tfidf)
7: H_features ← Concatenate (F_pca, F_rnn)
8: Return H_features
```

Due to feature-level concatenation can be used to combine disparate feature spaces with significantly different dimensions and statistical behavior, it was chosen for the fusion stage. While RNN-based embeddings are dense and low-dimensional, TF-IDF vectors usually have large dimensionality and sparse distributions. Without imposing alignment constraints that are necessary for additive, multiplicative, or attention-based fusion processes, concatenation maintains the complete representational capacity of both branches. In the same manner, the PCA dimension was set at 100 components to balance the trade-off between computing efficiency and variance retention; larger component sizes added redundancy without consistently improving accuracy, while smaller component sizes eliminated discriminative information.

Based on initial assessments carried out during model building, the PCA dimensionality was set at 100 components. Because discriminative TF-IDF variance was eliminated, lower configurations (especially in the range of 40–60) consistently decreased detection performance, and increasing the PCA size beyond 100 resulted in additional computational overhead without yielding appreciable gains in accuracy or stability. This demonstrates that 100 components offer an acceptable balance between knowledge retention and training efficacy. Although conducting a comprehensive study to eradicate a wide range of major component analysis configurations falls outside the scope of this main work, the observed behavior highlights the statistical sensitivity to component size and suggests that further exclusion may be a natural extension of this methodology.

The TF-IDF vectorization technique was employed to transform XSS payloads into a format that is appropriate for deep learning models. This approach deals with almost the statistical context of the complete information, focusing on statements that indicate harmful activity, and on those that are most frequently used.

Using samples from each dataset, the TF-IDF vectorizer was used to extract up to 1000 unique textual features. Each row is a sample, and each column represents a distinct token that was extracted during preprocessing.

Figures 8 and 9 display the top 20 TF-IDF features according to their cumulative weight across each dataset that will assist better understand the discriminative power of various tokens.

The fact that some letters, such as "script," "alert," and "test," are commonly used in XSS attacks emphasizes their importance in the vector space paradigm.

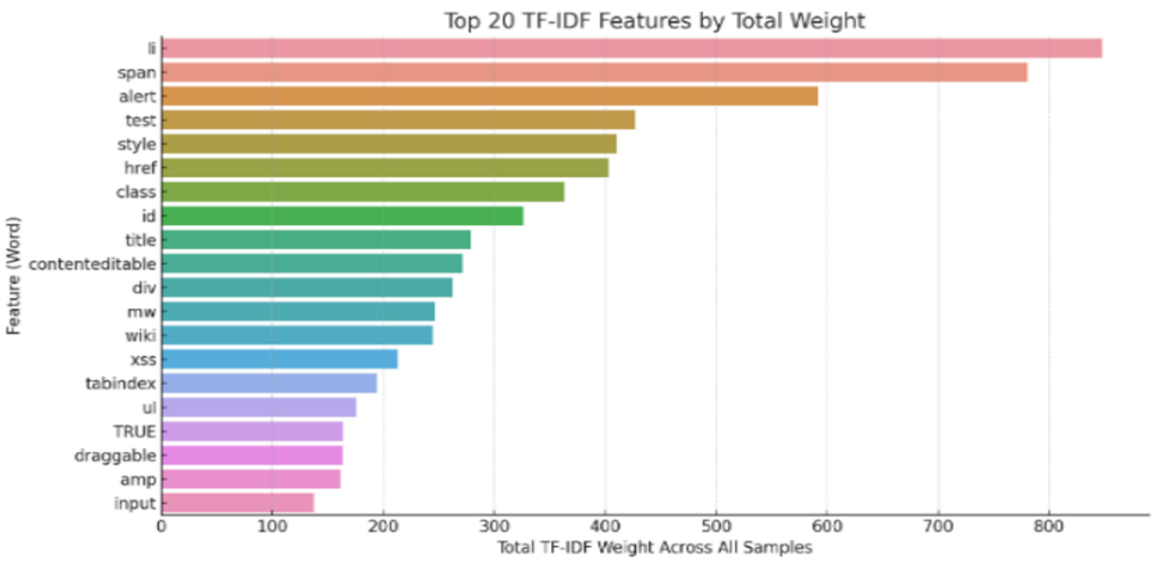

Figure 8. Features weight based on TF-IDF of Dataset_1

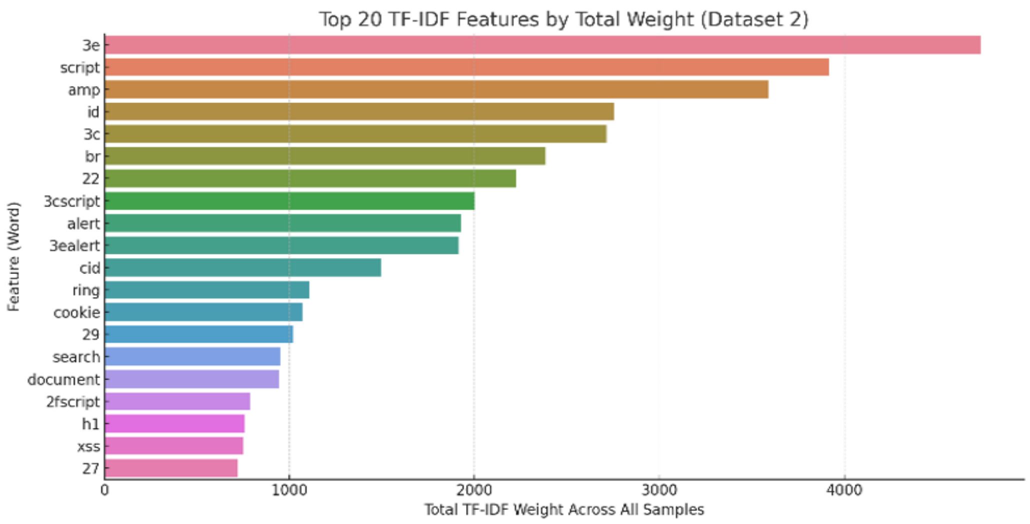

Figure 9. Features weight based on TF-IDF of Dataset_2

The sparsity of the TF-IDF representation and the concentration of weight in a few key features helps focus the learning on discriminative tokens, reducing the computational burden and improving generalization.

3.6 Dimensionality reduction (PCA)

To address the potential high dimensionality of TF-IDF vectors, PCA is applied prior to fusion in two of the models (PCA-LSTM and PCA-GRU). The number of components is empirically set to 100. PCA serves to reduce noise, enhance generalization, and accelerate convergence during training.

3.7 Model architecture and classifier

The proposed models consist of two input branches: one for padded sequences passed through an embedding layer and recurrent block (LSTM or GRU), and another for TF-IDF. The branches are merged and followed by dense layers with dropout regularization. The final output layer uses a sigmoid activation function for binary classification. Table 2 illustrates the hyperparameters configurations of the proposed model.

Table 2. The hyperparameter configurations for the proposed Early Feature Fusion models

Hyperparameter	**LSTM + TF-IDF \ GRU + TF-IDF**	**PCA-LSTM + TF-IDF \ PCA-GRU + TF-IDF**
Sequence encoder	units = 64	units = 64
TF-IDF feature size	1000	100 (after PCA)
Fusion layer	Concatenate ([sequence out, TF-IDF])	Concatenate ([sequence out, PCA_TF-IDF])
Hidden layers (post-fusion)	Dense(32, ReLU) → Dropout(0.3) → Dense(16, ReLU)	Dense(32, ReLU) → Dropout(0.3) → Dense(16, ReLU)
Output layer	Dense(1, Sigmoid)	Dense(1, Sigmoid)
Optimizer / LR	Adam (default LR ≈ 0.001)	Adam (default LR ≈ 0.001)
Loss	Binary Cross-Entropy	Binary Cross-Entropy
Epochs (max)	50 (with Early Stopping)	50 (with Early Stopping)
Batch size	32	32
Validation split	0.2	0.2
Train/Test split	70/30 (stratified)	70/30 (stratified)

3.8 Training and hyperparameter settings

All models were trained for a maximum of 50 epochs with a batch size of 32, using the Adam optimizer and binary cross-entropy loss function. A validation split of 20% was applied to monitor model performance during training. An early stopping mechanism was implemented to prevent overfitting and optimize training efficiency. The embedding layer produced 100-dimensional vectors, and the TF-IDF word vectorizer used a dimensionality of 1,000. Both fully connected layers included a dropout rate of 0.3 to mitigate overfitting. No hyperparameter tuning or optimization search was performed; instead, the same fixed parameters were applied across all model variations to ensure a fair comparison and to isolate the effect of early fusion and dimensionality reduction techniques.

3.9 Evaluation metrics

Performance was evaluated using an eight-point, well-thought-through, well-referenced, and well-established measure set that can be used in the case of binary classification problems. Out of these, Accuracy measures overall correctness in predictions; Precision denotes the value of true positives over all predicted positives; and Recall measures the ability to classify true malware instances rightly. To trade off Precision and Recall into one harmonic value, the term F1-score was utilized. To calculate credibility in predictions and statistical significance, the Matthews Correlation Coefficient (MCC) was employed. To measure the proper classification of benign samples, the terms 'specificity' (True Negative Rate) and (False Negative Rate) were utilized, while ROC–AUC (Receiver Operating Characteristic – Area Under the Curve) is a metric that evaluates the model's ability to distinguish between the positive and negative classes across all possible decision thresholds. The ROC curve is plotted using the True Positive Rate (TPR) against the False Positive Rate (FPR); this makes it particularly useful in evaluating how well the model separates malicious XSS payloads from legitimate requests in a balanced way. The same measure set was employed for experiments to calculate and achieve a multidimensional and interpretable process of assessment. All these measures were calculated based on the equations illustrated below Eqs. (1)-(8), which are the mathematical expressions for Accuracy, Precision, Recall, F1-score, MCC, FNR, TNR, and ROC–AUC, respectively.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

$$MCC = \frac{cov(X,Y)}{\sqrt{(cov(X,X) \times cov(Y,Y))}} \tag{5}$$

$$FNR = \frac{FN}{FN + TP} \tag{6}$$

$$TNR = \frac{TN}{TN + FP} \tag{7}$$

$$ROC - AUC = \sum_{i=1}^{n} \left(\frac{TPR_i + TPR_{i-1}}{2}\right) \times (FPR_i - FPR_{i-1}) \tag{8}$$

4. RESULTS

The experimental results of the proposed research will be presented in this section, after training and testing each model. Four hybrid deep learning-based models, i.e., LSTM + TFIDF, GRU + TFIDF, PCA-LSTM + TFIDF, and PCA-GRU + TFIDF, were compared among themselves by two datasets. They were compared against a large array of metrics, including accuracy, precision, recall, the F1-score, and others.

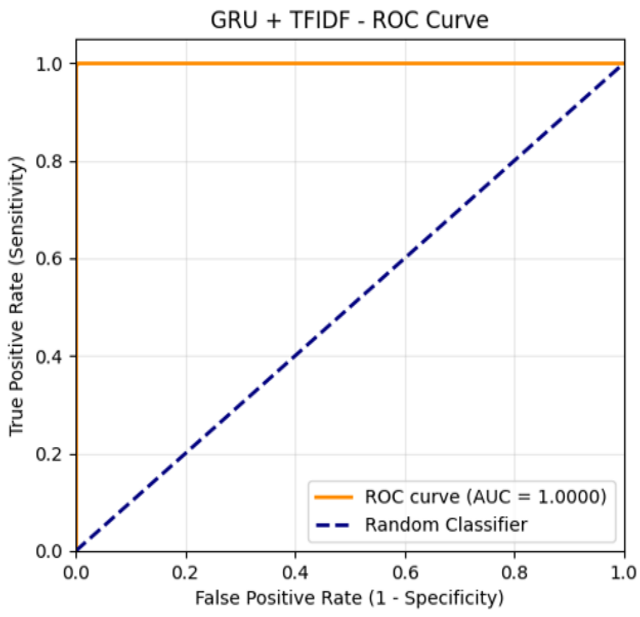

Figure 10. GRU+TFIDF ROC-AUC of Dataset_1

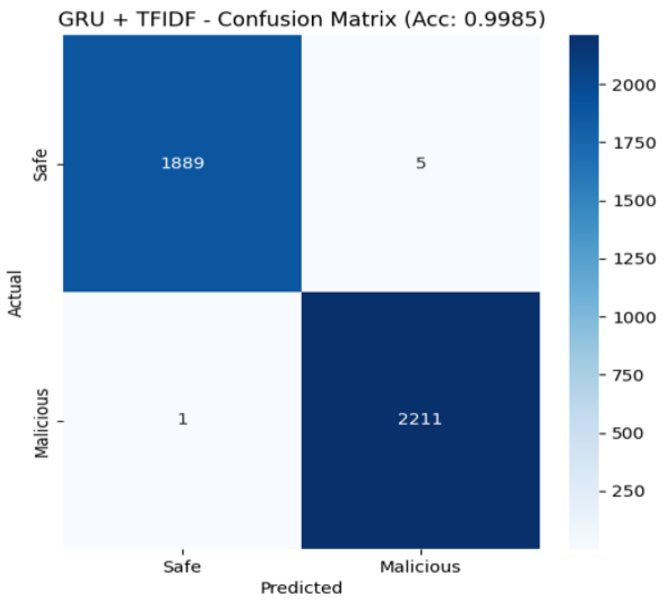

Figure 11. GRU+TFIDF confusion matrix of Dataset_1

The stability and consistency of the four analyzed models were investigated during several training runs to support the validity of the presented results. All models maintained a similar performance ranking in each trial, with GRU + TF-IDF consistently obtaining the highest results, despite slight variation. This consistency suggests that the given measures are not the product of random fluctuations, especially when compared with the nearly identical ROC curves produced throughout repeated runs.

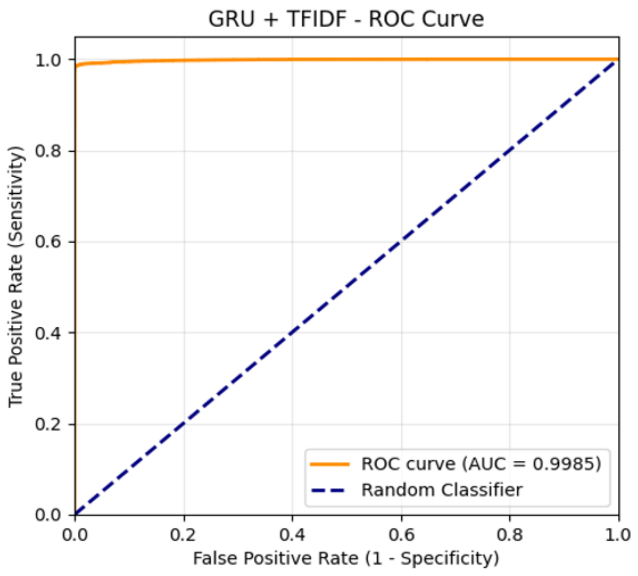

Figure 12. GRU+TFIDF ROC-AUC of Dataset_2

The fixed (train-test) evaluation protocol of the benchmark datasets does not include the use of confidence intervals and formal significance tests in this study; the convergence of accuracy, recall, and F1-score across several independent runs offers a useful indication that the observed performance differences are stable rather than incidental.

Ambiguous payloads in general, or those consisting of a mixture of benign and malicious scripts, may cause with many of false classifications. To handle ambiguous or confusing inputs, for this reason will need to employ more adaptive strategies; these critical cases illustrate the remaining scenarios in which the model may struggle.

The model was affected by PCA technology, which increased efficiency by lowering computational complexity and data noise, preserving most of the crucial data needed for classification. Because of this, the model is more suited for use in real-world settings where high efficiency and minimal resource consumption are required.

4.1 The results achieved by Dataset_1

Dataset_1 presented high performance for all proposed models. The accuracy ranged between 99.76% and 99.85%, while the AUC values (0.9999-1.0000) present a good ability to distinguish between benign and malicious samples. In addition, the Precision and Recall measures for the maliciously labeled samples exceeded 99.7% for all models, showing the effectiveness of the proposed approach in detecting attacks with low error rates. The GRU + TF-IDF model indicates better performance than all four models, with an accuracy of 99.85% and a recall of 99.95%, supported by an optimal AUC value of 1.0000, indicating that it is a highly promising option for real-time applications in crucial security settings. Table 3, and Figures 10 and 11 present these results by the ROC-AUC and confusion matrix for the GRU + TF-IDF model of Dataset_1.

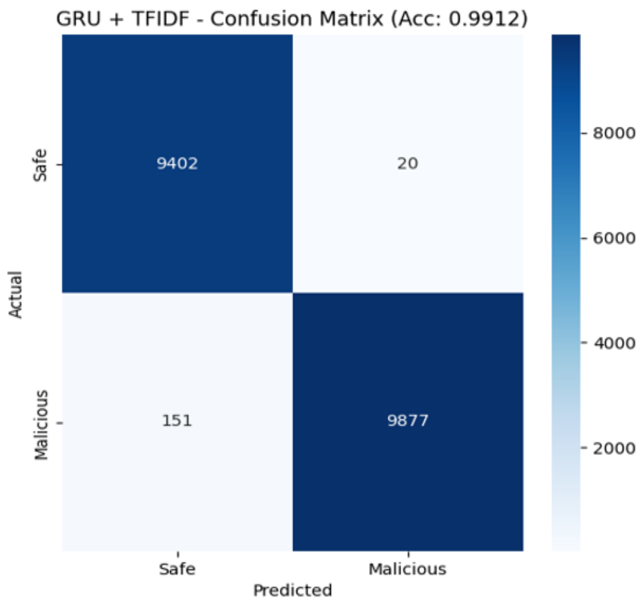

Figure 13. GRU+TFIDF confusion matrix of Dataset_2

4.2 The results achieved by Dataset_2

Dataset_2 results showed similar performance among the four proposed models, with accuracy ranging between 99.08% and 99.12%, reflecting a high ability to correctly classify samples. AUC values (0.9914-0.9985) also demonstrated an ability to distinguish between benign and malicious samples, with the GRU + TF-IDF model achieving the highest accuracy (99.12%) and the best AUC (0.9985). Precision metrics exceeded 99.8% for all models, demonstrating high reliability in correctly identifying malicious samples. Regarding Recall, values ranged between 98.34% and 98.50%, reflecting a strong ability to detect most attacks with a very low loss rate. Overall, these results demonstrate the efficiency of all models in dealing with this type of attack, with a clear advantage for the GRU + TF-IDF model, as shown in Table 4 and Figures 12 and 13, which represent the ROC-AUC and confusion matrix of the GRU + TF-IDF for the Dataset_2.

Table 3. Performance results of our proposed models (Dataset_1)

Model	Train Accuracy (%)	Test Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	MCC (%)	FNR	TNR (%)	ROC-AUC
LSTM + TFIDF	99.97	99.76	99.82	99.73	99.77	99.51	0.00271	99.79	0.99997
GRU + TFIDF	**99.96**	**99.85**	**99.77**	**99.96**	**99.86**	**99.71**	**0.00045**	**99.74**	**0.99998**
PCA-LSTM + TFIDF	99.98	99.76	99.82	99.73	99.77	99.51	0.00271	99.79	0.99990
PCA-GRU + TFIDF	99.94	99.81	99.86	99.77	99.82	99.61	0.00226	99.87	0.99996

Table 4. Performance results of our proposed models (Dataset_2)

Model	Train Accuracy (%)	Test Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	MCC (%)	FNR	TNR (%)	ROC-AUC
LSTM + TFIDF	99.41	99.10	99.84	98.40	99.12	98.2	0.01596	99.83	0.99832
GRU + TFIDF	**99.45**	**99.12**	**99.80**	**98.49**	**99.14**	**98.25**	**0.01506**	**99.79**	**0.99849**
PCA-LSTM + TFIDF	99.39	99.09	99.89	98.40	99.11	98.18	0.01596	99.81	0.99843
PCA-GRU + TFIDF	99.35	99.11	99.94	98.34	99.13	98.23	0.01665	99.94	0.99848

5. DISCUSSION

Our proposed system's findings demonstrated that using an early feature extraction fusion strategy within TF-IDF-extracted statistical representations and sequential contextual representations generated by LSTM or GRU models allowed deep learning techniques to understand the syntactic and semantic patterns of XSS attacks in a unified, integrated phase. This strategy created a learning environment that allowed the model to detect specific correlations between features from the beginning, as opposed to late fusion, which frequently examines features individually and then combines their results later, leading in the loss of key interaction information. Applying PCA to the TF-IDF outcomes before fusion also helped to reduce dimensionality and remove redundancy, which improved the models' capability to generalize and accelerate the training process while maintaining accuracy. This was reflected in the convergent performance by the four proposed models' high and convergent performance, with the GRU + TFIDF model surpassing the others on the majority of criteria. Figure 14 shows that the proposed model (GRU + TFIDF) achieved better results in terms of detection capabilities compared with the previous researches (mentioned in the related works) evaluated with the same dataset, demonstrating the efficiency of the methods used.

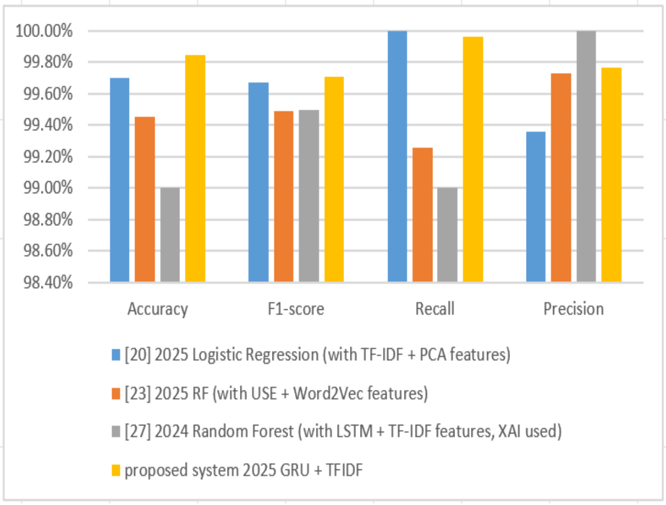

Figure 14. The related work based on Dataset_1

While the results are promising, claims of robustness and scalability remain limited to the two datasets evaluated and do not yet account for hostile or realistic deployment scenarios. Additional difficulties are introduced by practical integration into current Web Application Firewalls (WAFs), such as inference-time delay, resource constraints, and the requirement for regular model upgrades to support novel obfuscation techniques. While heavily obfuscated or borderline benign samples continue to be more challenging, the observed misclassification patterns indicate that early fusion offers special benefits for payloads combining statistical and contextual factors, such as partially obfuscated JavaScript fragments or encoded injections. All these elements highlight the current framework's practical limits and point to crucial areas for future research.

6. CONCLUSIONS

The work introduced the effectiveness of an Early Feature Fusion method that fuses statistical representations derived from TF-IDF with sequential contextual features derived from learning based on LSTM and GRU models and further enhanced by PCA to reduce dimension and computation complexity. Experimental findings on two benchmark datasets were high in performance with precision, recall, and F1-score rates all higher than 98%, and the most reliable results were obtained by the GRU + TF-IDF model. However, the reported results remain limited to controlled experimental settings and the two evaluated datasets, and do not yet reflect adversarial manipulation or real-world deployment scenarios. The findings reflect the meaningful performance of early fusion over late fusion and traditional methods. However, the work is limited to experimental settings and payload datasets; future work will be focused on extending the framework to real-time web applications and adversarial attack scenarios, which will be of more general applicability in modern intrusion detection systems. Future studies may explore additional architecture or evaluation settings, including transformer-based models or tests involving more diverse or adversarial payload variations.

REFERENCES

[1] Hannousse, A., Yahiouche, S., Nait-Hamoud, M.C. (2024). Twenty-two years since revealing cross-site scripting attacks: A systematic mapping and a comprehensive survey. Computer Science Review, 52: 100634. https://doi.org/10.1016/j.cosrev.2024.100634

[2] Maurel, H., Vidal, S., Rezk, T. (2022). Statically identifying XSS using deep learning. Science of Computer Programming, 219: 102810. https://doi.org/10.1016/j.scico.2022.102810

[3] Lee, S., Roh, D., Yu, J., Moon, D., Lee, J., Bae, J.H. (2025). Deep feature fusion via transfer learning for multi-class network intrusion detection. Applied Sciences, 15(9): 4851. https://doi.org/10.3390/app15094851

[4] Sharif, M.H.U. (2022). Web attacks analysis and mitigation techniques. International Journal of Engineering Research & Technology (IJERT), University of The Cumberlands, pp. 10-12.

[5] Hamzah, K.H., Osman, M.Z., Anthony, T., Ismail, M.A., Abdullah, Z., Alanda, A. (2024). Comparative analysis of machine learning algorithms for cross-site scripting (XSS) attack detection. JOIV: International Journal on Informatics Visualization, 8(3-2): 1678-1685. https://doi.org/10.62527/joiv.8.3-2.3451

[6] Taylor, O.E., Ezekiel, P.S. (2022). A robust system for detecting and preventing payloads attacks on web-applications using Recurrent Neural Network (RNN). European Journal of Computer Science and Information Technology, 10(4): 1-13. https://doi.org/10.37745/ejcsit.2013/vol10n4113

[7] Kolaib, R.J., Waleed, J. (2024). Crime activity detection in surveillance videos based on developed deep learning approach. Diyala Journal of Engineering Sciences, 17(3): 98-114. https://doi.org/10.24237/djes.2024.17307

[8] Li, Y., Hua, J., Wang, H., Chen, C., Liu, Y. (2021). Deeppayload: Black-box backdoor attack on deep learning models through neural payload injection. In 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), Madrid, ES, pp. 263-274. https://doi.org/10.1109/ICSE43902.2021.00035

[9] Abu, T.N.A., Doh, K.G. (2022). An analysis of machine-learning feature-extraction techniques using syntactic tagging for cross-site scripting detection. Journal of Software Assessment and Valuation, Journal of the Korean Society for Software Quality Assessment, 18(1): 107-118. https://doi.org/10.29056/jsav.2022.06.13

[10] Tadhani, J.R., Vekariya, V., Sorathiya, V., Alshathri, S., El-Shafai, W. (2024). Securing web applications against XSS and SQLi attacks using a novel deep learning approach. Scientific Reports, 14(1): 1803. https://doi.org/10.1038/s41598-023-48845-4

[11] Chen, X., Li, M., Jiang, Y., Sun, Y. (2019). A comparison of machine learning algorithms for detecting XSS attacks. In International Conference on Artificial Intelligence and Security, New York, NY, USA, pp. 214-224. https://doi.org/10.1007/978-3-030-24268-8_20

[12] Melicher, W., Fung, C., Bauer, L., Jia, L. (2021). Towards a lightweight, hybrid approach for detecting DOM XSS vulnerabilities with machine learning. In Proceedings of the Web Conference 2021, Ljubljana Slovenia, pp. 2684-2695. https://doi.org/10.1145/3442381.3450062

[13] Kumar, S., Pathak, S., Singh, J. (2022). An enhanced digital forensic investigation framework for XSS attack. Journal of Discrete Mathematical Sciences and Cryptography, 25(4): 1009-1018. https://doi.org/10.1080/09720529.2022.2072424

[14] Jain, D., Choudhary, D., Anand, A., Trivedi, N.K., Gautam, V., Mohapatra, S.K. (2022). Cybersecurity solutions using AI techniques. In 2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, pp. 1-8. https://doi.org/10.1109/ICRITO56286.2022.9965045

[15] Et-Tolba, M., Hanin, C., Belmekki, A. (2023). Intelligent systems for XSS attack detection: A brief survey. In 2023 International Wireless Communications and Mobile Computing (IWCMC), Marrakesh, Morocco, pp. 910-916. https://doi.org/10.1109/IWCMC58020.2023.10182407

[16] Gupta, C., Singh, R.K., Mohapatra, A.K. (2022). GeneMiner: A classification approach for detection of XSS attacks on web services. Computational Intelligence and Neuroscience, 2022(1): 3675821. https://doi.org/10.1155/2022/3675821

[17] Stiawan, D., Bardadi, A., Afifah, N., Melinda, L., et al. (2023). An improved LSTM-PCA ensemble classifier for SQL injection and XSS attack detection. Computer Systems Science & Engineering, 46(2): 1759. https://doi.org/10.32604/csse.2023.034047

[18] Oshoiribhor, E.O., John-Otumu, A.M. (2025). XSS-Net: An intelligent machine learning model for detecting cross-site scripting (XSS) attack in web application. Machine Learning Research, 10(1): 14-24. https://doi.org/10.11648/j.mlr.20251001.12

[19] Tan, X., Xu, Y., Wu, T., Li, B. (2023). Detection of reflected XSS vulnerabilities based on paths-attention method. Applied Sciences, 13(13): 7895. https://doi.org/10.3390/app13137895

[20] Li, X., Wang, T., Zhang, W., Niu, X., et al. (2023). An LSTM based cross-site scripting attack detection scheme for Cloud Computing environments. Journal of Cloud Computing, 12(1): 118. https://doi.org/10.1186/s13677-023-00483-x

[21] Bakır, R., Bakır, H. (2025). Swift detection of XSS attacks: Enhancing XSS attack detection by leveraging hybrid semantic embeddings and ai techniques. Arabian Journal for Science and Engineering, 50(2): 1191-1207. https://doi.org/10.1007/s13369-024-09140-0

[22] Luu, G.H., Duong, M.K., Pham-Ngo, T.P., Ngo, T.S., Nguyen, D.T., Nguyen, X.H., Le, K.H. (2024). XSShield: A novel dataset and lightweight hybrid deep learning model for XSS attack detection. Results in Engineering, 24: 103363. https://doi.org/10.1016/j.rineng.2024.103363

[23] Hu, Z., Zhang, J., Yang, H. (2025). XSS attack detection based on multisource semantic feature fusion. Electronics, 14(6): 1174. https://doi.org/10.3390/electronics14061174

[24] Alhamyani, R., Alshammari, M. (2024). Machine learning-driven detection of cross-site scripting attacks. Information, 15(7): 420. https://doi.org/10.3390/info15070420

[25] Younas, F., Raza, A., Thalji, N., Abualigah, L., Zitar, R.A., Jia, H. (2024). An efficient artificial intelligence approach for early detection of cross-site scripting attacks. Decision Analytics Journal, 11: 100466. https://doi.org/10.1016/j.dajour.2024.100466

[26] Farea, A.A., Amran, G.A., Farea, E., Alabrah, A., Abdulraheem, A.A., Mursil, M., Al-qaness, M.A. (2023). Injections attacks efficient and secure techniques based on bidirectional long short time memory model. Computers, Materials & Continua, 76(3): 3605-3622. https://doi.org/10.32604/cmc.2023.040121

[27] Shah, S.S.H. (2025). Dataset. https://www.kaggle.com/code/syedsaqlainhussain/cross-site-scripting-attack-detection-using-cnn/input.

[28] Fang, Y., Li, Y., Liu, L., Huang, C. (2018). DeepXSS: Cross site scripting detection based on deep learning. In Proceedings of the 2018 International Conference on Computing and Artificial Intelligence, Chengdu, China, pp. 47-51. https://doi.org/10.1145/3194452.3194469