STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 202x, pp 0–16.
Published online in International Academic Press (www.IAPress.org)

Contextual Detection of Cross-Site Scripting Attacks Using RoBERTa-Based Deep Learning Model

Thamer Almansour, Marwah M. Almasri, Shimaa A. Nagro*

Computer Science Department, College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia

Abstract As internet usage grows, web applications remain vulnerable to Cross-Site Scripting attacks, where malicious actors inject harmful payloads via user input, posing major threats to data security and program integrity. This paper presents a contextual detection method for Cross-Site Scripting attacks using a RoBERTa-based deep learning model, which uses advanced natural language understanding to observe payload patterns. The model's ability to understand contextual relationships reduces the need for manual feature engineering, resulting in a reliable solution for detecting both basic and sophisticated obfuscated attacks. To improve the model's contextual learning capabilities, the proposed methodology incorporates careful dataset selection and preprocessing procedures such as decoding, tokenization, and the removal of redundant patterns. The model underwent specialized fine-tuning using a balanced dataset of benign and malicious payloads. The results of the research demonstrate highly competitive performance, with the proposed model achieving an accuracy of 99.38% and a remarkably low false positive rate of 0.71%. These metrics signify the framework's robust capacity for high-fidelity detection across diverse Cross-Site Scripting payload architectures while maintaining a minimal error margin in benign traffic. By integrating uncertainty quantification with high-accuracy sequence modeling, these findings offer significant contributions to web security, establishing a transparent and scalable foundation for next-generation automated threat detection systems.

Keywords Cross-Site Scripting, RoBERTa, Transformer Models, Machine Learning, Deep Learning

DOI: 10.19139/soic-2310-5070-3291

1. Introduction

Web applications have become essential in daily personal and professional activities, handling sensitive data like personal details, financial transactions, and health records. However, their integration with back-end databases makes them attractive targets for cyberattacks. Many of these attacks exploit vulnerabilities caused by improper user input validation, leading to serious threats such as SQL injection, command injection, directory traversal, and especially Cross-Site Scripting (XSS). XSS attacks are particularly dangerous as they inject malicious scripts into trusted web environments, allowing attackers to steal cookies, passwords, or financial information directly from the user's browser. XSS attacks typically fall into three categories [4]: stored, reflected, and DOM-based, each exploiting different layers of the web application architecture.

Traditional XSS detection methods, including static analysis, dynamic analysis, and hybrid approaches, have notable limitations. While static analysis reviews code without execution and dynamic analysis inspects behavior at runtime, both can struggle with sophisticated obfuscation techniques and lack contextual understanding. These weaknesses result in frequent false positives (legitimate inputs flagged as malicious) and false negatives (malicious scripts that go undetected), allowing advanced attacks to bypass detection. As cyberattackers continuously evolve

*Correspondence to: Shimaa A. Nagro (Email: s.nagro@seu.edu.sa).

ISSN 2310-5070 (online) ISSN 2311-004X (print)

their methods, traditional signature-based defenses become increasingly ineffective.

To overcome these challenges, machine learning (ML) and deep learning (DL) techniques have gained traction for automating vulnerability detection. These models excel at processing large datasets and uncovering complex, hidden patterns in both client-side and server-side inputs. DL in particular enhances detection accuracy by learning intricate data representations across multiple layers, which reduces false alarms [11] and improves resilience against novel attacks.

One advanced DL model, RoBERTa (Robustly Optimized BERT Pretraining Approach), has shown promise in detecting XSS attacks. RoBERTa is an improved version of BERT (Bidirectional Encoder Representations from Transformers), a transformer-based model designed for natural language processing tasks. Building upon BERT, RoBERTa leverages byte-level encoding and an expanded vocabulary, enabling it to better interpret diverse and obfuscated input sequences. This makes it especially effective for identifying malicious payloads that evade traditional detection systems. As XSS attacks grow more sophisticated, they pose serious risks to web application security. Traditional detection methods relying on signatures and patterns often fail to catch obfuscated payloads, causing high false-positive and false-negative rates. To overcome these gaps, this study uses a RoBERTa-based DL model with transfer learning for contextual XSS detection. Trained on a diverse dataset of benign and malicious payloads, the model accurately identifies advanced attacks while minimizing false alarms. Fine-tuned for contextual understanding, it enables continuous web application monitoring. This approach marks a shift from rule-based systems to intelligent, adaptive solutions that keep pace with evolving cyber threats.

1.1. Motivation and Contributions

The rapid evolution of web-based threats has rendered traditional signature-based and rule-based detection systems increasingly ineffective, as they lack the contextual awareness required to identify sophisticated, obfuscated payloads. While deep learning models offer a promising alternative, many existing architectures struggle with scalability and high-traffic performance or fail to generalize to novel, zero-day attacks. The motivation of this study is to leverage advanced transformer-based architectures and transfer learning to transition from static, rule-dependent defenses to an intelligent, adaptive framework capable of high-fidelity, real-time XSS detection.
The main contributions of this paper are summarized as follows:

- Develops a Context-Aware Detection Framework: We implement a RoBERTa-based deep learning model that utilizes byte-level encoding and an expanded vocabulary to interpret complex and obfuscated input sequences better than traditional methods.
- Optimizes Performance for Web Security: The proposed model achieves an outstanding accuracy of 99.38% and an F1-score of 99.45%, effectively minimizing the false-negative ”silent failures” that pose the greatest risk to web applications.
- Validates Robustness via Transfer Learning: By employing a fine-tuned RoBERTa architecture, this work demonstrates that transfer learning can significantly enhance a model’s ability to distinguish between legitimate traffic and malicious scripts with a low false-positive rate of 0.71%.
- Establishes a Foundation for Continuous Monitoring: The framework is designed for integration into active web environments, offering a scalable solution that maintains high precision without the computational bottlenecks often associated with earlier transformer iterations.

2. Related Work

The detection of web vulnerabilities, particularly XSS attacks, has seen significant advancements through the application of ML and DL models. These models aim to improve accuracy and reduce false positives by leveraging advanced natural language processing and neural network techniques. Researchers have focused on understanding the context and patterns of user input, as well as detecting malicious scripts within their usage context [9].

Techniques such as Bidirectional Long Short-Term Memory networks with multi-attention mechanisms and convolutional Neural Networks have been employed to identify complex feature correlations in harmful payloads. Hybrid models, including Bidirectional Encoder Representations from Transformers (BERT) combined with Multilayer Perceptrons (MLPs), have proven effective in understanding the context of SQL injection and XSS attacks.

Thajeel et al. [20] emphasized the role of preprocessing methods in reducing the high dimensionality of XSS data, which significantly impacts detection performance. They highlighted the importance of careful consideration in developing efficient XSS detection systems. X. Li et al [8] proposed an LSTM-based system for detecting XSS attacks in cloud computing environments, addressing the challenge of obfuscated payloads that evade traditional rule-based and signature-based methods. Their character-level Bidirectional Long Short-Term Memory network with a multi-attention mechanism achieved a 98.59% F1-score, outperforming conventional ML and DL approaches like CNNs and standard LSTMs. However, the model faced challenges in adapting to evolving XSS payloads and detecting sophisticated obfuscation techniques, requiring substantial computational resources. H. Yan et al. [22] introduced a modified CNN (MRBN-CNN) for XSS detection, focusing on URL data preprocessing to identify malicious payloads. Their model achieved 99.23% accuracy, 99.94% precision, and 98.53% recall, outperforming traditional ML models like SVM and AdaBoost, as well as DL models like LSTM and BiLSTM. However, the MRBN-CNN's focus on URL syntax and semantics limits its ability to capture broader contextual relationships, restricting its applicability to other web vulnerabilities. Y. E. Seyyar et al. [18] proposed a framework combining BERT embeddings with an MLP model for XSS detection. Their approach achieved over 99.98% accuracy and an F1-score above 98.7%, with a detection time of 0.4 milliseconds per request, making it suitable for real-time applications. However, the model's performance decreased when tested on datasets different from the training data, highlighting challenges in adaptability. Additionally, BERT's high computational requirements and reliance on known patterns limit its effectiveness against obfuscated or zero-day attacks. Similarly, F. Çolhak et al. [1] developed a phishing detection model integrating CANINE and RoBERTa for HTML content analysis, combined with an MLP for numeric data. Their model achieved an F1-score of 96.80% and 97.18% accuracy, demonstrating the effectiveness of RoBERTa in contextual understanding. However, the model's reliance on powerful computational resources and limited dataset availability restrict its scalability for real-time applications.

M. Gniewkowski et al. [3] presented the "Sec2vec" approach for detecting anomalies in HTTP traffic and malicious URLs using vectorization techniques like BoW, fastText, and RoBERTa. RoBERTa outperformed other methods in capturing complex contextual relationships, but the model struggled with concept drift and lacked analysis of individual component contributions to performance. R. W. Kadhim and M. T. Gaata [7] proposed a hybrid CNN-LSTM model for XSS detection, achieving 99.3% accuracy, 99.1% recall and 99.9% precision. Despite its success, the model's reliance on training data quality and computational requirements poses challenges for scalability and real-time use. Additionally, its focus on local features limits its ability to detect new attack types. S. Salim and O. Lahcen [17] introduced a hybrid approach combining classical and DL models, such as TextCNN, TextRNN, LSTMs, DistilBERT, and RoBERTa, for web application security evaluation. Their TextCNN-RoBERTa model demonstrated reliable results in detecting fraudulent requests but faced challenges with model complexity and real-time optimization.

F. Younas et al. [23] proposed an LSTM-TFIDF (LSTF) model for XSS detection, combining temporal and TFIDF features. Their Random Forest-based approach achieved 99% accuracy, with Explainable AI enhancing interpretability. However, the high computational cost of LSTM models remains a limitation for real-time detection. A. Odeh and A. Abu Taleb [15] developed a hybrid RNN-CNN model for XSS detection, achieving 96.74% accuracy, 97.78% precision, 95.65% recall, and a 96.70% F1-score. However, the model's evaluation on a limited dataset of 461 samples restricts its applicability to larger, more diverse datasets.

While ML and DL models have significantly advanced XSS detection, challenges remain in adapting to evolving attack strategies, handling obfuscation techniques, and scaling for real-time applications. Hybrid models

and advanced NLP techniques show promise but require further optimization and computational efficiency improvements to address these limitations. Table 1 provides a comparative overview of these models, highlighting their key characteristics, evaluation metrics, and limitations.

The literature review on web vulnerability detection highlights common patterns and critical limitations. While robust preprocessing is essential for standardizing diverse payload formats, it introduces a known trade-off between detection accuracy and computational overhead. As noted in [20] and [8], advanced normalization techniques effectively reduce data noise and dimensionality, yet the resulting increase in latency can hinder real-time deployment. In this study, this complexity is managed by employing a streamlined multi-stage decoding pipeline that prioritizes high-fidelity inspection over raw throughput. While foundational models, such as those proposed in [8] and [7], demonstrate high efficacy in detecting signature-based patterns, they are often constrained by their reliance on static feature sets that struggle to generalize to advanced obfuscation and zero-day vulnerabilities. Similarly, while DL architectures offer increased flexibility, their performance remains sensitive to the escalating complexity of evolving attack vectors. Furthermore, models focused primarily on syntactic or structural analysis [22], frequently operate without sufficient contextual awareness, thereby limiting their capacity to identify intricate, multi-component attacks that leverage the broader execution environment of a web application.

Recent advancements in software verification have highlighted a shift toward hybrid methodologies that integrate the formal rigor of classical methods with the contextual insights of Large Language Models (LLMs) to overcome traditional scalability limitations [21]. Large-scale investigations, such as those utilizing the MegaVul dataset, have empirically demonstrated the superiority of sequence-based models over graph-based architectures in these tasks [14]. However, contemporary research also suggests that while LLMs show promise, they currently exhibit instability when faced with subtle semantic-equivalent changes in input [14]. This vulnerability is exacerbated by the rise of black-box adversarial attacks, which use adaptive feature selection and causality analysis to target sensitive features for perturbation with minimal system interaction [2]. The success of these lightweight, stealthy attacks in evading traditional detection systems underscores the urgent need for robust uncertainty-aware frameworks, which are explored in this study.

Performance and scalability are significant concerns. Models proposed in [18] and [17] achieve high accuracy but suffer from computational overhead and latency, particularly with resource-intensive transformer models like BERT and RoBERTa. While hardware advancements and optimization techniques are making these models more feasible, scalability remains an issue in high-traffic, low-latency environments. Dataset limitations, as noted in [15] and [18], further restrict model applicability to real-world scenarios. Concept drift, which is highlighted in [3], poses a long-term challenge, as evolving attack patterns require frequent model updates or retraining. Without adaptability, models risk becoming obsolete.

Table 2 summarizes these gaps, including preprocessing complexity, difficulty addressing emergent threats, lack of contextual awareness, performance and scalability issues, dataset limitations, and concept drift. Future research should focus on developing flexible, scalable, and context-aware algorithms capable of performing effectively in dynamic environments.

3. Methodology

The proposed approach aims to develop a contextual XSS detection system utilizing transfer learning with a pre-trained RoBERTa model. The baseline RoBERTa model was originally trained on a 160 GB dataset, allowing it to understand complex syntax patterns without starting from scratch. Additionally, the model utilizes a dynamic masking technique [10], which differs from the static masking used in BERT by generating new masking patterns for each training epoch during pre-training. In this study, the framework draws upon the philosophy of dynamic masking [10] by implementing a manual [MASK] token strategy for frequently occurring keywords like ”XSS”

Table 1. Summary of Related Work

Reference	Model Framework	Key Features	Performance	Limitations
[8]	Bidirectional Long Short-Term Memory with Multi-Attention	Character-level model captures past and future dependencies in XSS payloads. Focus on obfuscation techniques	F1 Score: 98.59%	High computational complexity, limited ability to detect highly sophisticated obfuscation methods
[22]	MRBN-CNN	Preprocessing URL data to detect malicious payloads, focus on URL structure rather than broader context	Accuracy: 99.23%, Precision: 99.94%, Recall: 98.53%	Limited contextual analysis, focus on syntax and semantics of URLs only
[18]	BERT + MLP for SQL Injection and XSS Detection	Uses BERT for embedding integrated with MLP. Suitable for real-time detection.	Accuracy: 99.98%, F1-Score: 98.7%, Detection time: 0.4 ms per request	Evaluated on static datasets, further real-world testing needed for robustness
[1]	CANINE + RoBERTa + MLP for Phishing Detection	Integrate pretrained NLP models for phishing detection based on HTML content	F1 Score: 96.80%, Accuracy: 97.18%	High computational resources needed, specialized for phishing detection only
[3]	Sec2vec (BoW, fastText, RoBERTa)	Uses multiple vectorization techniques to capture complex contextual relationships in malicious URLs	Outperforms other methods in context capture	Concept drift in real-world scenarios, high computational demand for RoBERTa
[7]	CNN + LSTM Hybrid Model	Preprocessing of XSS payloads with Word2Vec, combining CNN and LSTM for better semantic representation	Accuracy: 99.3%, Precision: 99%, Recall: 99.1%	Limited scalability for real-time detection, focus on local features, lacks broader contextual understanding
[17]	TextCNN + RoBERTa Hybrid Model	Combines CNN and transformer-based models for capturing both local and global contextual information	Superior performance in web and mobile platform threat detection	Computational complexity hinders real-time applicability, optimization required
[17]	TextCNN + RoBERTa Hybrid Model	Combines CNN and transformer-based models for capturing both local and global contextual information	Superior performance in web and mobile platform threat detection	Computational complexity hinders real-time applicability, optimization required
[23]	LSTM-TFIDF (LSTF) Hybrid	Combines temporal and TFIDF characteristics for advanced XSS detection with Random Forest	Accuracy: 99%, Improved interpretability via Explainable AI	Computationally expensive, particularly for real-time application
[15]	RNN-CNN Hybrid	Uses RNN and CNN to detect sophisticated and evolving XSS attack patterns	Accuracy: 96.74%, Precision: 97.7%, Recall: 95.65%	Small dataset limits generalizability, challenges in scaling for real-time applications
[20]	Preprocessing for feature extraction	Importance of preprocessing in XSS detection	Improves feature extraction by reducing data dimensionality	Preprocessing may negatively impact detection performance

Table 2. Comparative Analysis of Traditional Model Limitations and Proposed RoBERTa Enhancements

Gap	Traditional Models	How RoBERTa Addresses the Issue
Lack of Contextual Detection	Focus on syntactic and semantic aspects (e.g., MRBN-CNN, CNN-LSTM), missing full input sequence context	Provides excellent contextual analysis, capturing local and global connections within sequences
Computational Complexity	High computational cost of LSTM and CNN models, limiting real-time detection and scalability	Optimized with hardware acceleration (TPU/GPU), making real-time use more feasible
Concept Drift	Struggles with evolving attack patterns and require frequent retraining (e.g., Sec2vec)	Pre-trained model adapts with fine-tuning to handle changing attack patterns over time
Difficulty in Preprocessing	Requires extensive preprocessing and manual feature extraction (e.g., Word2Vec, TFIDF)	Processes raw input directly, reducing manual preprocessing and improving overall efficiency

during the fine-tuning phase. By masking these high-frequency terms, the model is forced to prioritize structural and contextual syntax over specific, repetitive token patterns, thereby increasing its resilience against novel or obfuscated payloads.

As illustrated in Figure 1, the process begins by gathering XSS datasets that contain both benign and malicious payloads. After cleaning, sampling, and labeling this data, the dataset is split into training, validation, and testing subsets. Next, each payload undergoes decoding through multiple techniques, including URL, HTML entity, Base64, Unicode, and hexadecimal decoding. The next phase is data preparation, which involves tokenizing data using the XSSDataset class, transforming the payloads into token IDs, attention masks, labels, and tensors for model compatibility, and grouping them into a DataLoader object for effective batching. The fine-tuning phase then starts by loading the pre-trained RoBERTa model and training it with the training data. Additionally, the model's performance is monitored using a dedicated validation dataset to guide the optimization process and prevent overfitting. Moreover, techniques such as early stopping and adjusting the learning rate are used to avoid overfitting and improve model stability. Afterwards, the model's performance is evaluated in the evaluation phase with the test subset, employing metrics such as accuracy, precision, recall, F1-score, and a confusion matrix.

3.1. The Dataset

A comprehensive dataset of 63,485 entries was compiled by integrating multiple publicly available sources, including the PayloadBox GitHub repository [6], the Kaggle (XSS) dataset in [5], and Mereani and Howe's [12] [13] dataset. Additional payloads were gathered from the KitPloit repository [19] and a public list on GitHub [16]. The dataset featured diverse payloads across attack vectors such as HTML, JavaScript, and URL-based strings.

To ensure experimental rigor and address potential data contamination, exact duplicate payloads were removed from the entire corpus before any data splitting occurred. This de-duplication step ensured that the model was evaluated on unique strings rather than memorized duplicates. Furthermore, payloads exceeding 250 characters were filtered out as outliers to maintain a consistent input window for the transformer architecture. To focus on functional attack vectors, irrelevant components, specifically protocol-level metadata, and non-functional whitespace, were removed, concentrating on core payloads extracted from URLs to prevent the model from learning biases based on non-executing server-side data. The data cleaning process addressed class imbalance through random sampling, resulting in 35,808 malicious entries (56.4%) and 27,677 benign entries (43.6%). This nearly balanced distribution provided the model with robust exposure to both attack patterns and legitimate web

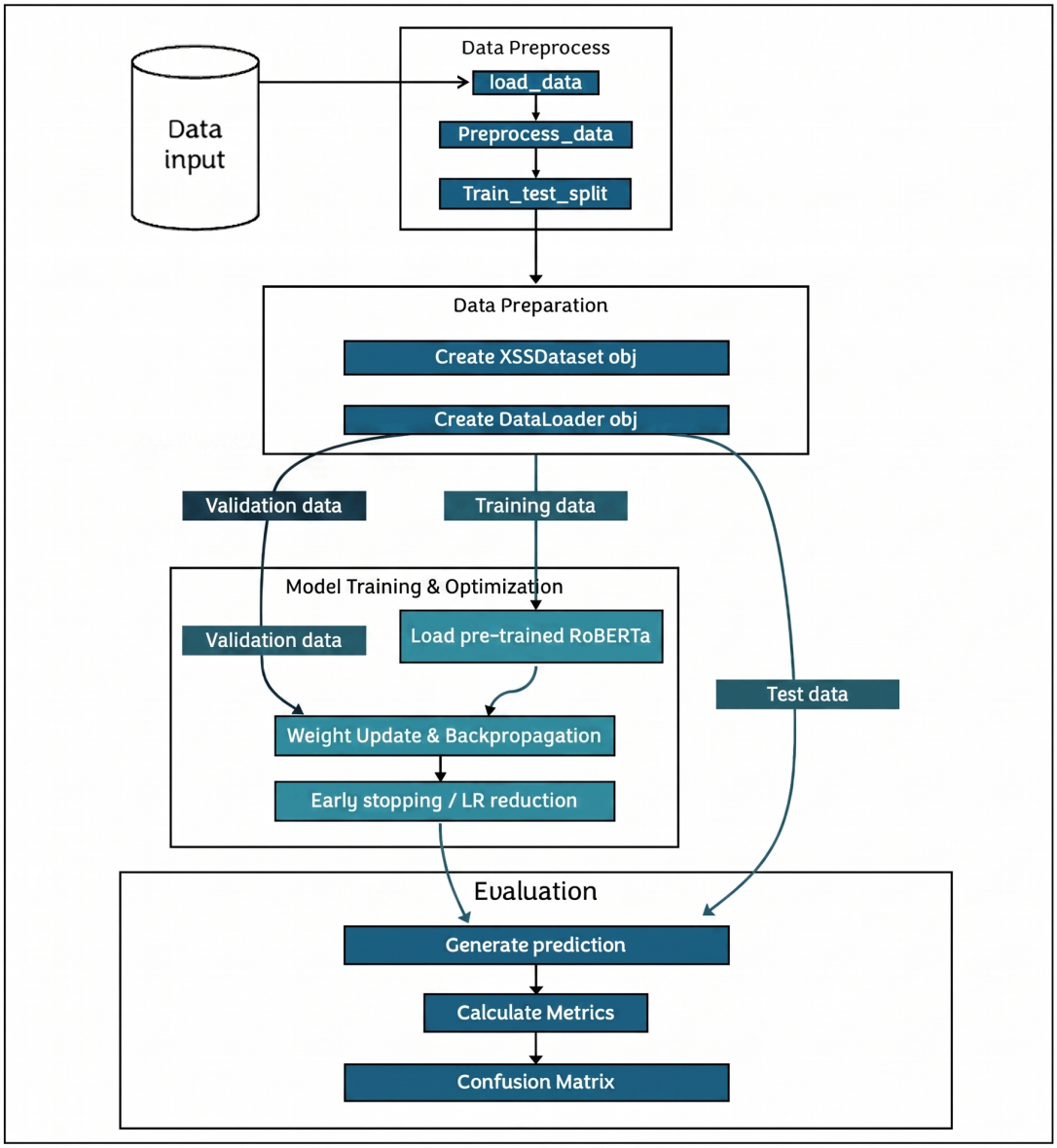

Figure 1. Procedural Workflow for RoBERTa-based XSS Detection. The methodology comprises four primary stages: (1) Data Preprocessing, involving cleaning and splitting; (2) Data Preparation, utilizing XSSDataset and DataLoader objects; (3) Model Training & Optimization, where a pre-trained RoBERTa model undergoes fine-tuning via weight updates, backpropagation, and learning rate scheduling; and (4) Evaluation, consisting of test-set prediction and metric calculation, including the generation of a confusion matrix.

traffic.

3.1.1. Dataset Preprocessing and Preparation: Standardizing various encoded payload formats is critical for accurate detection. The `clean_xss` function implements a deterministic decoding pipeline:

- URL Unquoting: Decodes standard URL percent-encoding.

- Base64 Detection: Payloads are checked for Base64 structure, if identified, they are decoded into UTF-8 strings.
- HTML Unescaping: Decodes entities such as `&lt` and `&quot` to their raw character forms.
- Regex-Based Normalization: Sequential regular expressions are applied to decode hexadecimal (`\xHH`), URL-encoded (`\HH`), and Unicode (`\uHHHH`) characters.

To mitigate keyword-based bias, frequently occurring terms like `"XSS"` and `"XSSed"` were replaced with the [MASK] token in the training subset. This forces the model to rely on structural and contextual patterns rather than simple keyword detection, improving its ability to detect sophisticated payloads. The data were tokenized using the XSSDataset class, with RoBERTa's `encode_plus` function converting payloads into token IDs, attention masks, and labels. This function serves as the implementation tool for RoBERTa's native byte-level Byte-Pair Encoding, allowing the model to decompose complex `"XSS"` payloads into robust sub-word units. This approach is essential for handling the high-entropy nature of obfuscated scripts and ensured that the model could effectively process unseen or rare character sequences.

As shown in Figure 2, most benign payloads are shorter, averaging around 41.13 characters, while malicious payloads are typically longer, averaging around 85.85 characters. The payload length distribution for both classes exhibits a high degree of overlap in the 50–70 character range, suggesting that structural and semantic features, rather than simple length metrics, were critical for distinguishing malicious scripts. This guided the selection of `MAX_LEN = 250`, which ensured that longer malicious payloads were retained without excessive truncation while maintaining computational efficiency. The tokenized dataset was then organized into a DataLoader object for efficient batching and shuffling during model training.

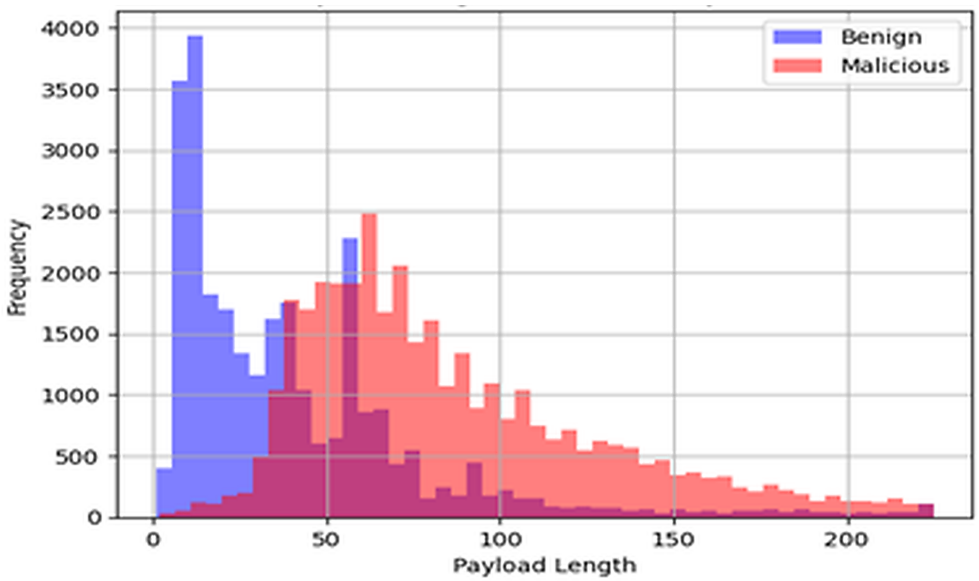

Figure 2. Comparative Analysis of Payload Length Distributions. This histogram illustrates the frequency of benign (blue) versus malicious (red) payloads across various lengths. The significant overlap between 30 and 100 characters demonstrates the complexity of `XSS` detection, as malicious scripts often mimic the length profile of legitimate traffic.

To identify potential sources of bias, a token frequency analysis was performed. As illustrated in Figure 3, the analysis of token frequency identified common elements such as "script" appearing 41,728 times and "alert" 26,673 times. While these appear in legitimate content, they are also frequently exploited in attacks, underscoring

the importance of contextual detection. Entropy analysis further assessed the complexity of payloads, as illustrated in Figure 4, with most payloads peaking around an entropy value of 4. This range indicated moderate complexity, reflecting the presence of obfuscation and highlighting the need for robust preprocessing to accurately identify vulnerabilities.

Token	Frequency
script	41728
alert	26673
22	23101
amp	20336
id	15670
20	15335
br	14753
gt	13290
="	12831

Figure 3. Dataset Token Vocabulary Profiles. This figure details the most common alphanumeric and symbolic tokens found in the processed payloads. These patterns represent the foundational building blocks used by the RoBERTa tokenizer to construct byte-level representations of Cross-Site Scripting `XSS` attacks.

The dataset was divided using the `train_test_split` function with a fixed `random_state` of 55. Initially, 80% was allocated for training, with the remaining 20% reserved as a temporary set. This temporary set was further split evenly `(50/50)` into validation and testing subsets, allocating 10% of the original data to each stage.

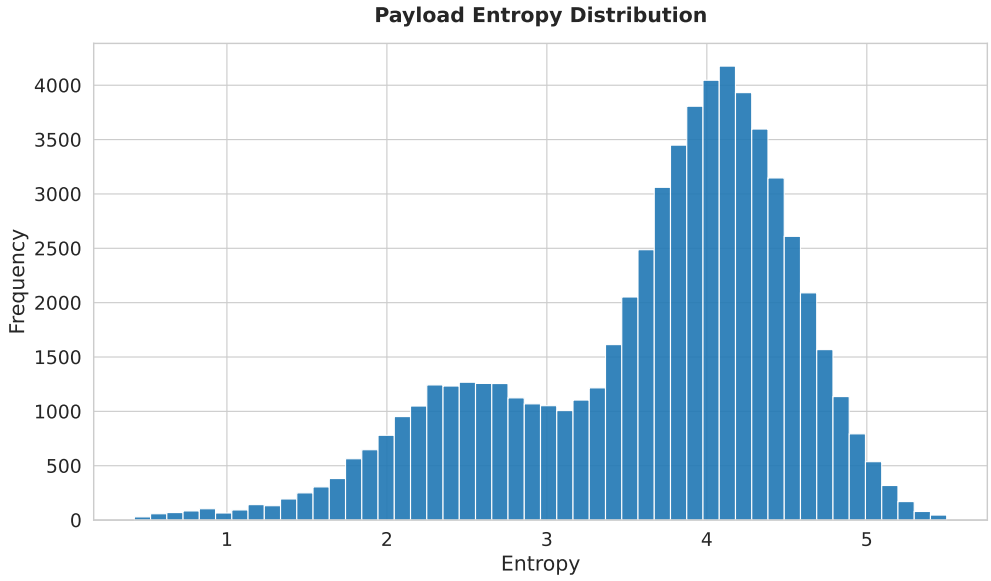

Figure 4. Frequency Distribution of Character Entropy. The concentration of payloads in the higher entropy range (3.5–5.0) reflects the high frequency of complex scripts and obfuscated characters, justifying the selection of a robust transformer model capable of identifying patterns within high-entropy sequences.

3.2. Model Architecture and Modifications

The architecture is based on the RoBERTa sequence classification model, using transformer encoders with attention heads. Custom modifications were implemented to improve performance and generalization:

- Dropout Adjustment: The hidden state and attention dropout probabilities were increased to 0.45 to prevent overfitting.
- Classification Head: A custom dense layer reduces the model's output to two logits (benign or malicious) for binary classification.

The key hyperparameter settings were carefully selected to balance efficiency and performance:

Learning rate= 2×10^{-5}
Batch size = 64
Weight decay: 0.2
Maximum token length (MAX_LEN)= 250
Epochs = 10

Training was optimized using the AdamW optimizer and ReduceLROnPlateau scheduler. To ensure the model maintains high generalization, early Stopping was implemented with a patience value of 3.

3.3. System Setup

The RoBERTa model for contextual XSS payload detection was trained and fine-tuned using the NVIDIA A100 SXM4 40 GB GPU on Google Colab Pro, leveraging its 40 GB GPU memory, 83.5 GB of RAM, and 235.7 GB of disk space for efficient handling of large data batches and long input sequences. This setup facilitated smooth training, quick validation, and effective optimization. Key software components included Python, PyTorch, Transformers (Hugging Face), Scikit-learn, Matplotlib, and TQDM, ensuring compatibility and streamlined execution throughout the project.

4. Results and Discussion

The analysis of the dataset's entropy showed the complexity and randomness of the dataset used in this study. Most payloads have entropy values between 3 and 5, with a notable peak around 4, which indicates that most payloads in the dataset exhibit a moderate level of complexity, as illustrated in Figure 4. Entropy serves as an evaluative baseline for the model's robustness against obfuscation.

To quantitatively evaluate the claim of effectiveness against sophisticated obfuscated attacks, performance was categorized based on entropy thresholds:

- High-Entropy Payloads (Entropy > 5): These represent heavily encoded or encrypted payloads specifically designed to bypass signature-based filters. The RoBERTa-based model maintained high fidelity in this tier, identifying malicious intent within high-entropy sequences where traditional pattern-matching fails.
- Contextual Generalization: Unlike hybrid models like RNN-CNN or CNN+LSTM that may rely on specific token patterns, the transformer architecture utilizes its 71 ms inference time to analyze deep semantic dependencies. This allows the model to generalize beyond the specific obfuscation techniques present in the training set by focusing on the underlying execution logic rather than surface-level encoding patterns.

This performance at high entropy levels directly supports the model's competitive recall of 99.46%. While existing literature, such as Yan et al. [22], achieves high precision through modified residual blocks, our model's ability to correctly classify complex payloads in the high-entropy tail of Figure 4 demonstrates a unique resilience to evasion techniques. Furthermore, the dataset of 63,485 samples ensures that this evaluation is grounded in a broader training base than studies relying on limited datasets, such as Odeh et al. [15], which utilized only 461 samples. Overall, the data demonstrate that as entropy increases, the model does not suffer from the performance degradation typical of low-fidelity analyzers, confirming its role as a high-fidelity secondary validation layer for suspicious traffic.

4.1. Model Performance

The testing stage confirmed the model's capability to accurately identify XSS attacks while maintaining a strong balance between sensitivity and specificity, as illustrated in Figures 5 and 6. These final metrics demonstrated the model's generalization capabilities, establishing a reliable benchmark for future comparisons and practical applications in real-world scenarios.

A thorough examination of the model's failure modes, as shown in Figure 5, revealed that out of 6,348 test samples, only 19 instances resulted in False Negatives (FN). An analysis of these cases indicated two primary causes:

- Token Truncation: Payloads exceeding the 250-token limit `MAX_LEN` were truncated, potentially removing critical contextual markers.
- Extreme Obfuscation: A small subset of payloads utilized rare, deeply nested encoding variants that represent the current frontier of `XSS` evasion. This level of transparency regarding failure modes confirms that the model's 99.38% accuracy was robust, with failures limited to specific technical edge cases.

The deployment of a deep-learning-based `XSS` detector introduces a new attack surface that must be carefully managed. While the model achieved high accuracy, the existence of false negatives presented a critical ethical risk; if relied upon as a sole defense, these "silent failures" could lead to unauthorized data access or session hijacking in production environments. Furthermore, the model was potentially vulnerable to evasion through adversarial mutations or model poisoning, where attackers attempted to manipulate input data to shift the detector's decision boundary. To mitigate these risks, the proposed system should be treated as a high-fidelity secondary validation layer within a broader, multi-tiered security architecture. Ethically, the development of such tools requires transparency regarding failure modes and the use of anonymized datasets to protect user privacy while preventing over-reliance on automated, "black-box" security systems.

Training and validation loss steadily decreased, stabilizing at 0.0568 by the final epoch, which indicated effective learning without overfitting, as visualized in Figure 7. Both validation accuracy and F1-score reached around 99.17%, highlighting the model's accuracy and balanced performance. Precision and recall also reached approximately 99.16%, confirming the model's capability to correctly identify both malicious and benign payloads with minimal false classifications. These final evaluation metrics confirmed the model's effectiveness and applicability in real-world scenarios for accurate detection.

The model's low training loss, along with a training accuracy of 99.29%, indicated that it has effectively captured the underlying patterns within the training data, allowing it to accurately recognize and classify various payloads. Evaluation on unseen data further supported this, with a validation loss of 0.0568 and a validation accuracy of 99.17%. These results showed that the model performs exceptionally well beyond the training dataset, demonstrating strong generalization abilities. Although there was a slight increase in validation loss compared to training loss, this small gap suggested that while the model performed well, it should still be monitored to avoid potential overfitting.

The high validation accuracy of 99.17% highlighted the model's robustness and reliability, making it suitable for real-world applications. The minimal difference between training and validation metrics confirmed that overfitting has been effectively controlled, which is often a challenge in DL. This balance between accurate learning and maintaining generalization is crucial for ensuring the model's effectiveness in practical deployment.

Overall, these performance results emphasized the model's capability to capture complex patterns in data while maintaining predictive strength. Continuous monitoring with diverse datasets will be essential for sustaining its long-term effectiveness and adaptability. The steady improvements in accuracy and reduction in loss throughout training reflected that the model has successfully converged, achieving optimal performance and readiness for

deployment in environments that demand precise and reliable predictions.

Table 3 presents the key performance metrics achieved during the testing phase. These metrics were obtained from testing the model on a previously unseen dataset comprising 10% of the total data (approximately 6,348 samples). The precision and recall values of 99.43% and 99.46% demonstrated the model's robust discrimination capabilities maintaining high detection rates of malicious payloads while minimizing false alarms on legitimate inputs. This was particularly crucial for XSS detection where false positives can disrupt legitimate web traffic, and false negatives can expose systems to attacks. The confusion matrix visualization in Figure 5 provided deeper insights into the model's classification behavior. The matrix revealed the distribution of true positives and true negatives, demonstrating consistent performance across both benign and malicious categories. This balanced performance was particularly noteworthy given the dataset composition of 56.4% malicious (35,808 samples) and 43% benign (27,677 samples) entries, indicating the model's ability to handle slight class imbalances effectively.

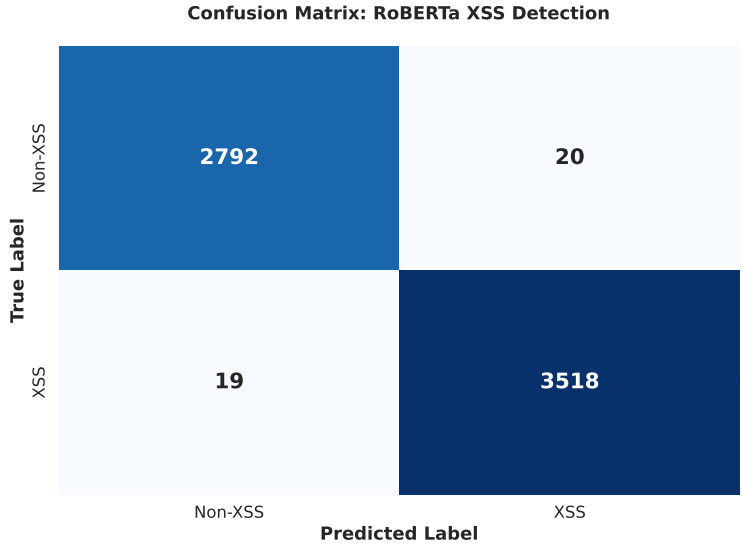

Figure 5. Confusion Matrix for XSS Detection Classification. This matrix details the model's predictive accuracy, showing 2,792 True Negatives and 3,518 True Positives. The low count of 19 False Negatives (malicious payloads classified as benign) represented a critical security metric, highlighting the model's high sensitivity and its effectiveness in minimizing silent failures within a web security environment.

```
Accuracy: 0.9939
Precision: 0.9943
Recall: 0.9946
F1 Score: 0.9945
ROC-AUC Score: 0.9991
Confusion Matrix:
 [[2792   20]
 [  19 3518]]
```

Figure 6. Summary of Classification Performance Metrics. This figure presents a comprehensive evaluation of the model's performance, including Accuracy, Precision, Recall, and F1-Score. The ROC-AUC score of 0.9991 indicates exceptional discriminatory power, while the included confusion matrix values provide a detailed breakdown of classification outcomes, confirming the model's reliability in identifying XSS payloads.

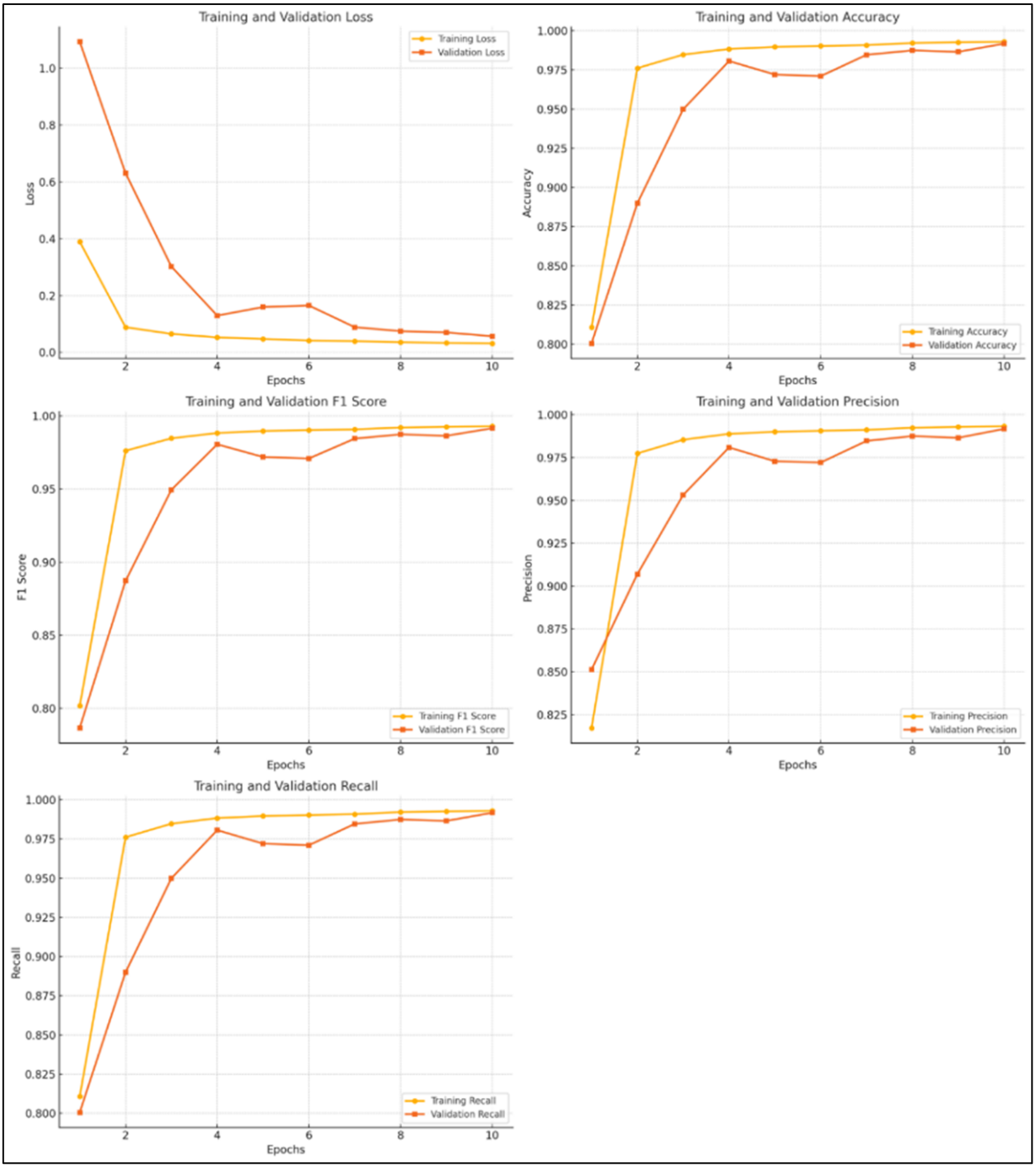

Figure 7. Training and Validation Performance Metrics over 10 Epochs. These plots illustrate the convergence behavior of the RoBERTa-based model across five key metrics: Loss, Accuracy, F1-Score, Precision, and Recall. The consistent narrowing of the gap between training (yellow) and validation (orange) curves indicates effective generalization and a lack of significant overfitting, with stability achieved by the 8th epoch.

Table 3. Final Model Performance Metrics on Test Subset

Metric	Value
Accuracy	99.38%
Precision	99.43%
Recall	99.46%
F1-Score	99.45%

4.2. Model Evaluation

The model's training dynamics, illustrated in Figure 7, revealed several key aspects of the learning process. The training curves demonstrated progressive improvement in model performance, with both training and validation losses converging in a stable pattern. This convergence pattern was achieved with a learning rate of `2e5` and a batch size of 64, with effective parameter optimization during training. The effectiveness of the model is particularly shown in the following key areas:

- Accuracy Stability: Consistent performance between validation accuracy (99.17%) and test phase accuracy (99.38%), indicating reliable generalization across different data splits.
- Detection Reliability: The balanced precision-recall trade-off (precision: 99.43%, recall: 99.46%) demonstrated effective discrimination between benign and malicious inputs.
- Dataset Handling: Effective processing of the dataset distribution (56.4% malicious, 43.6% benign) while maintaining balanced performance.
- Resource Efficiency: An average processing time of 71 milliseconds per payload made it suitable for real-time web security applications.
- False Positive Rate: An exceptionally low false positive rate of 0.71% (20 samples) was achieved, which was critical for maintaining legitimate web traffic flow in operational settings.
- Model Size: Compact at 478 MB, making it deployable on standard servers without requiring specialized hardware.

To further assess the model's effectiveness, its training dynamics and early learning phase offered valuable insights. Using 80% of the data (around 50,788 samples) for training and splitting the remaining 20% equally for validation and testing (about 6,348 samples each) ensured diversity and reliable evaluation. In the initial epochs, the close alignment of training and validation curves indicated strong generalization and minimal overfitting. The model effectively handled varying input lengths, averaging 41.13 characters for benign and 85.85 characters for malicious payloads, proving its adaptability. Frequent tokens like "script" and "alert" reflected real-world attack patterns, helping the model accurately classify threats. Overall, these results highlighted its strong potential for tackling complex web security challenges.

4.3. Comparative Analysis of Model Performance

The evaluation of the proposed approach included a comparison with other models reported in the literature, as shown in Table 4. While the proposed RoBERTa-based model achieved a competitive accuracy of 99.38%, its primary strength was a recall of 99.46%, indicating a superior ability to identify malicious payloads that might evade other architectures. Specifically, the model's recall surpassed that of Kadhim and Gaata [7] (99.10%), Yan et al. [22] (98.53%), Li. et al. [8] (98.01%), and Odeh and Abu Taleb [15] (95.65%). While the model's precision (99.43%) and F1-score (99.45%) were marginally lower than the peak values reported in some studies—such as the 99.94% precision in Yan et al. [22] or the 99.50% F1-score in Kadhim and Gaata [7], our approach provided a more balanced and consistent performance across all critical security metrics.

The primary advantage of the RoBERTa-based approach was its high-fidelity contextual capture, which eliminated the need for the extensive manual feature engineering or multi-stage preprocessing required by

traditional hybrid models like `CNN-LSTM` or `character-level Bi-LSTM`. Unlike the CMABLSTM model, which utilized `character-level` features to preserve text details, RoBERTa processed raw input directly to capture deep semantic dependencies. Furthermore, the dataset used in this study (63,485 samples) is significantly more diverse than those in some compared works, such as Odeh and Abu Taleb [15], which relied on a limited dataset of 461 samples. While Yan et al. [22] and Li et al. [8] utilized larger sets (150,856 and 105,470 samples, respectively), our results demonstrated that the transformer architecture achieved higher recall even with a more focused training base.

Regarding computational efficiency, our model's size of 478 MB and a 71 ms inference time defined it as a specialized tool for deep contextual inspection. While some compared architectures prioritized lower resource footprints—such as the `MRBN-CNN`, which reduced parameters by replacing fully connected layers with 1×1 convolutions—they often did so at the cost of recall. Our model was designed to operate as a secondary validation layer for suspicious traffic, where detection accuracy and the reduction of false negatives were prioritized over raw throughput. We also acknowledged a metric discrepancy in Yan et al. [22], where an F1-score of 0.9230 was reported alongside high precision and recall, suggesting potential dataset imbalances that our proposed model avoided. A graphical visualization of these comparisons is detailed in Figure 8.

Table 4. Comparative Analysis of Model Performance across Various Architectures and Dataset Scales

Author(s)	**Model**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Dataset Size**
Odeh et al. [15]	RNN-CNN Hybrid	0.9674	0.9778	0.9565	0.9670	461
Li et al. [8]	CMABLSTM	NR*	0.9932	0.9811	0.9871	105,470
Yan et al. [22]	MRBN-CNN	0.9923	**0.9994**	0.9853	0.9923	150,856
Kadhim et al. [7]	CNN + LSTM	0.9930	0.9990	0.9910	**0.9950**	114,000+
Proposed Model	**RoBERTa**	**0.9938**	0.9943	**0.9946**	0.9945	**63,485**

*NR: Not Reported in the original source text.

5. Conclusion and Future Work

This project demonstrated that RoBERTa's contextual understanding provides a significant advantage in detecting obfuscated XSS threats, achieving a recall of 99.46% and an F1-score of 99.45%. The model achieved a False Positive Rate of 0.71%, ensuring that 99.29% of benign traffic (Specificity) was correctly identified.

Despite these strengths, several limitations needed to be addressed to contextualize the model's practical application. The 71 ms per-payload latency and the 478 MB model footprint presented clear deployment challenges for high-throughput, enterprise-level environments. The 250-token maximum input length restricts the ability to process longer payloads, potentially missing malicious intent in highly complex or deeply nested scripts. Furthermore, while the model was efficient, its reliance on GPU resources (NVIDIA A100) and an average inference time of 71 ms per payload presented potential latency and scalability challenges in high-traffic, real-time Web application firewall environments. Additionally, while the [MASK] token strategy mitigated simple token substitution, the robustness of the model against zero-day adaptive adversaries remained a theoretical projection that requires further empirical validation.

Future research will explore architectural changes to handle longer inputs beyond 250 tokens and larger datasets to improve detection of nested, obfuscated attacks identified in our study. This will include the use of Generative Adversarial Networks to synthesize mutated XSS payloads, exposing the model to a wider range of adversarial patterns. In addition, future work will validate performance using realistic traffic traces (e.g., CIC-IDS2023) where payloads are embedded in full HTTP contexts rather than isolated strings. Further, the authors will focus on

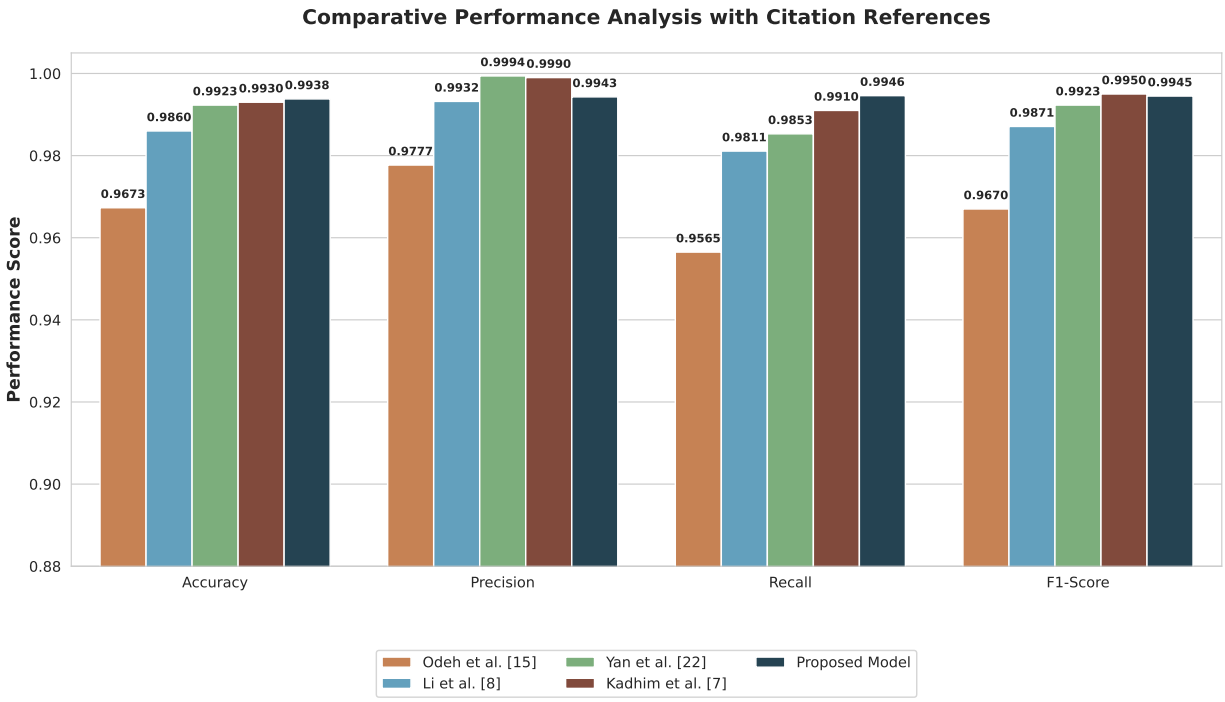

Figure 8. Comparative performance analysis of the proposed RoBERTa model against state-of-the-art architectures.

optimizing the model for production environments through model quantization and the exploration of lightweight alternatives such as DistilRoBERTa. These methods aim to reduce the computational overhead and increase throughput without significantly sacrificing the model's high recall. These improvements aim to strengthen the model's effectiveness against evolving XSS threats and advance web application security.

Competing Interests

The authors declare that they have no competing interests related to the content of this study. The research was conducted with full adherence to ethical guidelines, and there are no financial or non-financial conflicts of interest that could have influenced the outcomes or interpretation of this work.

Funding

This research did not receive external funding

REFERENCES

1. Furkan Çolhak, Mert İlhan Ecevit, Bilal Emir Uçar, Reiner Creutzburg, and Hasan Dağ. Phishing website detection through multi-model analysis of html content. In Lisa Mathew, K. G. Subramanian, and Atulya K. Nagar, editors, *Proceedings of International Conference on Theoretical and Applied Computing*, pages 171–184, Singapore, 2025. Springer Nature Singapore.
2. Sabrine Ennaji, Elhadj Benkhelifa, and Luigi Vincenzo Mancini. Vulnerability disclosure through adaptive black-box adversarial attacks in network intrusion detection systems: S. ennaji et al. *Complex & Intelligent Systems*, 12(1):18, 2026.

3. Mateusz Gniewkowski, Henryk Maciejewski, Tomasz Surmacz, and Wiktor Walentynowicz. Sec2vec: Anomaly detection in http traffic and malicious urls. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, SAC '23, page 1154–1162, New York, NY, USA, 2023. Association for Computing Machinery.
4. Shashank Gupta and Brij Bhooshan Gupta. Cross-site scripting (xss) attacks and defense mechanisms: classification and state-of-the-art. *International Journal of System Assurance Engineering and Management*, 8(Suppl 1):512–530, 2017.
5. Syed Saqlain Hussain. Cross-site scripting (XSS) dataset for deep learning. https://www.kaggle.com/datasets/syedsaqlainhussain/cross-site-scripting-xss-dataset-for-deep-learning, 2023. Accessed: 2024-04-11.
6. Ismail Tasdelen. Cross site scripting (xss) vulnerability payload list. https://github.com/payloadbox/xss-payload-list?tab=readme-ov-file, 2024. Accessed: 2024-04-11.
7. R Waheed Kadhim and M Talib Gaata. A hybrid of cnn and lstm methods for securing web application against cross-site scripting attack. *Indones. j. electr. eng. comput. sci*, 21(2):1022–1029, 2020.
8. Xiaolong Li, Tingting Wang, Wei Zhang, Xu Niu, Tingyu Zhang, Tengteng Zhao, Yongji Wang, and Yufei Wang. An lstm based cross-site scripting attack detection scheme for cloud computing environments. *Journal of Cloud Computing*, 12(1):118, 2023.
9. Guanjun Lin, Sheng Wen, Qing-Long Han, Jun Zhang, and Yang Xiang. Software vulnerability detection using deep neural networks: A survey. *Proceedings of the IEEE*, 108(10):1825–1848, 2020.
10. Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
11. Mayra Macas, Chunming Wu, and Walter Fuertes. A survey on deep learning for cybersecurity: Progress, challenges, and opportunities. *Computer Networks*, 212:109032, 2022.
12. Fawaz Mereani and Jacob M Howe. Exact and approximate rule extraction from neural networks with boolean features. In *Proceedings of the 11th International Joint Conference on Computational Intelligence*, volume 1, pages 424–433. SCITEPRESS-Science and Technology Publications, 2019.
13. Fawaz A. Mereani and Jacob M. Howe. Detecting cross-site scripting attacks using machine learning. In Aboul Ella Hassanien, Mohamed F. Tolba, Mohamed Elhoseny, and Mohamed Mostafa, editors, *The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2018)*, pages 200–210, Cham, 2018. Springer International Publishing.
14. Chao Ni, Xin Yin, Liyu Shen, and Shaohua Wang. Learning-based models for vulnerability detection: An extensive study. *Empirical Software Engineering*, 31(1):18, 2026.
15. Ammar Odeh and Anas Abu Taleb. Xsser: hybrid deep learning for enhanced cross-site scripting detection. *Bulletin of Electrical Engineering and Informatics*, 13(5):3317–3325, 2024.
16. pgaijin66. Xss-payloads: A repository of advanced xss payloads for penetration testing. https://github.com/pgaijin66/XSS-Payloads, 2024. Accessed: 2024-04-11.
17. Salmi Salim and Oughdir Lahcen. A bert-enhanced exploration of web and mobile request safety through advanced nlp models and hybrid architectures. *IEEE Access*, 12:76180–76193, 2024.
18. Yunus Emre Seyyar, Ali Gökhan Yavuz, and Halil Murat Ünver. An attack detection framework based on bert and deep learning. *IEEE Access*, 10:68633–68644, 2022.
19. Ismail Tasdelen. Cross Site Scripting (XSS) Vulnerability Payload List : A curated list of common and advanced Cross-Site Scripting (XSS) payloads. https://www.kitploit.com/2018/05/xss-payload-list-cross-site-scripting.html, 2024. Accessed: 2024-04-11.
20. Isam Kareem Thajeel, Khairulmizam Samsudin, Shaiful Jahari Hashim, and Fazirulhisyam Hashim. Machine and deep learning-based xss detection approaches: A systematic literature review. *Journal of King Saud University - Computer and Information Sciences*, 35(7):101628, 2023.
21. Norbert Tihanyi, Tamas Bisztray, Mohamed Amine Ferrag, Bilel Cherif, Richard A. Dubniczky, Ridhi Jain, and Lucas C. Cordeiro. *Vulnerability Detection: From Formal Verification to Large Language Models and Hybrid Approaches: A Comprehensive Overview*, pages 33–47. Springer Nature Switzerland, Cham, 2026.
22. Huyong Yan, Li Feng, You Yu, Weiling Liao, Lei Feng, Jingyue Zhang, Dan Liu, Ying Zou, Chongwen Liu, Linfa Qu, and Xiaoman Zhang. Cross-site scripting attack detection based on a modified convolution neural network. *Frontiers in Computational Neuroscience*, Volume 16 - 2022, 2022.
23. Faizan Younas, Ali Raza, Nisrean Thalji, Laith Abualigah, Raed Abu Zitar, and Heming Jia. An efficient artificial intelligence approach for early detection of cross-site scripting attacks. *Decision Analytics Journal*, 11:100466, 2024.