Impact of XSS Attacks on Cybersecurity and Detection Approaches Using Machine Learning Techniques: A Survey

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

[1]Department of Computer Science, College of Science, University of Diyala, Baqubah, Iraq

[2]University of Information Technology and Communication, Baghdad, Iraq

Emails: scicomphd232404@uodiyala.edu.iq; ziyad1964tariq@uodiyala.edu.iq

Abstract

The dramatically increasing use of web applications and the rapid development of cloud services and interactive websites that provide integrated online services, relying on user data entry and server response, have been the primary drivers of the increase in cyber-attacks and threats, most notably cross-site scripting (XSS). Cross-site scripting attacks exploit available security vulnerabilities to inject malicious code, leading to numerous risks such as malware distribution, session hijacking, and data theft. Most traditional defense methods, such as input validation and output encoding, are reasonably ineffective against advanced threats. The advances in machine learning and artificial intelligence models have provided more accurate detection and prevention capabilities for these threats with significant accuracy. This study reviews the types and mechanisms of XSS attacks, existing mitigation techniques, and detection methods based on machine and deep learning. It also highlights several previous studies and related work on detecting and preventing these attacks, compares these works' performance using evaluation metrics and several aspects, identifies research gaps, and outlines future directions for improving XSS detection methods.

Keywords: Cross-Site Scripting (XSS); Cybersecurity; Machine learning; Web application security; AI-based threat detection

1. Introduction

The use of the internet in most areas of modern technology and the increasing reliance on web applications in various aspects of life have led to some of the most important global priorities, such as protecting data and the privacy of users' information from cyber threats. In a related context, cyberspace is considered a fertile environment for most cybercrimes, as hackers can access and exploit security vulnerabilities to breach systems and thus steal sensitive data, such as users' personal and financial information and others, without the need for direct interaction with the victims [1]. Web applications are the most widespread means in cyberspace; due to their ease of use and availability, these applications often contain security vulnerabilities that make them vulnerable to cyber-attacks. Among these vulnerabilities, cross-site scripting (XSS) attacks stand out as one of the top 25 most dangerous programming errors, ranking second in the Common Weakness Enumeration (CWE) according to MIT Research and Engineering (MITRE) Corporation and SysAdmin, Audit, Network, and Security (SANS) Institute [2]. XSS attacks have been ranked many times among the top 10 threats and security risks according to the Open Web Application Security Project (OWASP) [3][4], as shown in Figure (1). These attacks allow attackers to inject malicious code into trusted web pages, exposing sensitive user data, such as cookies and user sessions, to theft or manipulation. XSS attacks are classified into three main types: reflected XSS, stored XSS, and Document Object Model (DOM)-based XSS. Reflected attacks occur when malicious code is injected through links that are delivered to victims, whereas stored attacks are found in the storage of malicious code in infected web page databases and, as such, expose all users of the infected page to risk. The DOM-based XSS attacks, however, leverage the web browser's environment to carry out malicious code, with the latter not needing to directly communicate with the server.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

Despite significant advancements in protection techniques, such as encryption and attack detection mechanisms, cross-site scripting attacks remain a serious threat to information security. Therefore, cybersecurity researchers continue to develop new techniques for identifying and avoiding attacks based on targeting the improvement of artificial intelligence and machine learning algorithms, like deep neural networks and decision trees, to make detection systems more effective.

This research is distinguished by its reliance on the latest studies published between 2021 and 2025, which examined modern techniques in machine learning and deep learning, including hybrid approaches that integrate multiple models to enhance detection performance. In this review, the results are extracted using specific tables according to standardized evaluation criteria (Accuracy, Precision, Recall, and F1-score), with a critical analysis of the limitations of these models. Furthermore, the research highlights new research gaps, such as the need to integrate explainable AI techniques, achieve real-time detection, and expand to cloud computing and IoT environments.

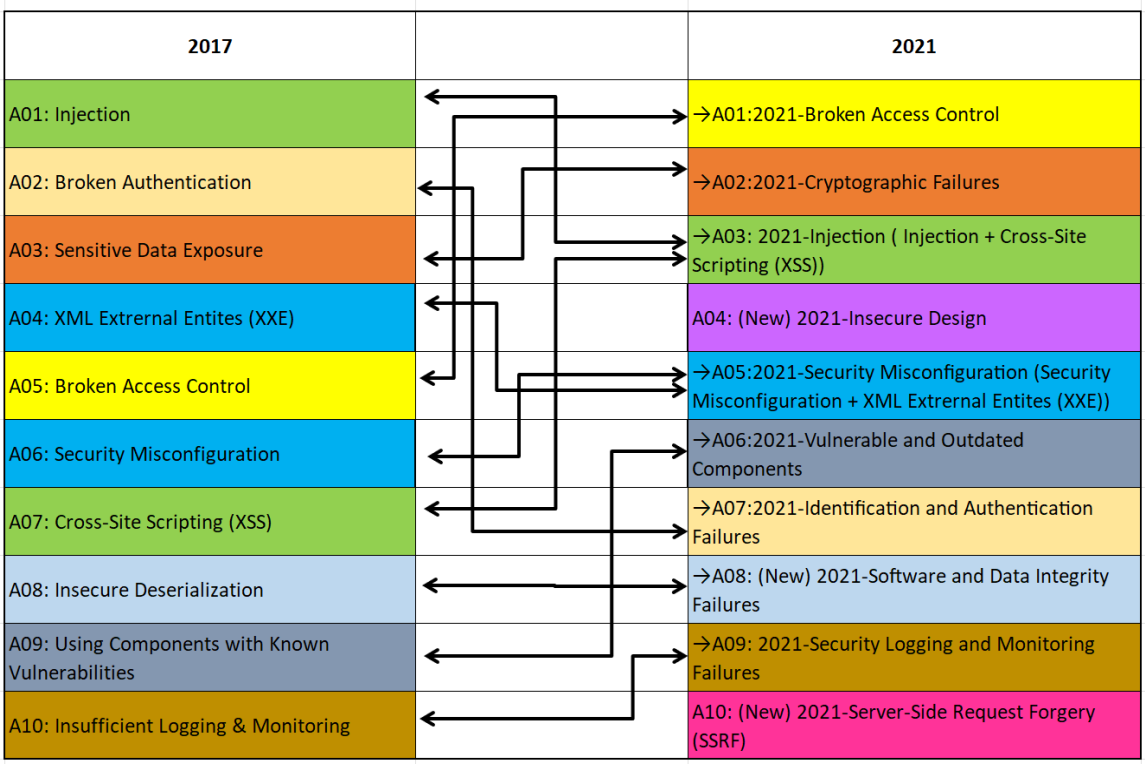

Figure 1. The top 10 security risks (2017, 2021) according to the Open Web Application Security Project (OWASP). [3] https://owasp.org/www-project-top-ten/

2. Fundamental Concepts of Cybersecurity

One of the fundamental pillars of the modern technological world is the term cybersecurity, which has become increasingly important in the fields of security and protection of all digital aspects, including computers, servers, mobile devices, networks, databases, and applications. The primary goal in this field is to focus on protecting digital systems and data from cyber threats and attacks, which have begun to increase in various forms, such as threats to achieve unauthorized access, modification, or destruction. In a more general sense, cybersecurity goes beyond data security, information security, and network security, encompassing the flow of digital information from end to end [5]. Cybersecurity is more comprehensive, adopting a wide range of concepts, technologies, strategies, and frameworks designed to detect and defend against malicious attacks. These threats and attacks can be considered and classified into several types, the most important of which are unauthorized access, tampering with or modifying data, which pose risks to individuals and organizations.

2.1 Data Security

Data security focuses on safeguarding digital data throughout its lifecycle, including creation, storage, transfer, processing, and deletion. Essentially, data security seeks to secure data from tampering or fake modification and prevent unauthorized access. Data security is one of the vital parts of cybersecurity, as data is the primary target and goal of cyber-criminals.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

2.2 Information Security

Information security is a broader concept that includes the protection of information, whether digital or physical [6]. Its main goal is to ensure confidentiality, integrity, and availability of information. Information security involves preventing unauthorized access, use, modification, or unlawful destruction of information.

2.3 Network Security

Network security aims to protect the security of computer networks and data delivered over communication media; this involves protecting network infrastructure from threats that may disrupt services or compromise data [7]. Network security is an important aspect of cybersecurity since networks are the primary means of data transmission and contribute to the dependability of today's digital communication systems.

2.4 The Importance of Cybersecurity

In an era of greater reliance on technology, cybersecurity has become vital to protect individuals, businesses, and governments from cyber-attacks. Cybersecurity is a combination of technical, procedural, and administrative methods for protecting digital information, data, and networks from internal attacks and external threats, including cyber-attacks and data breaches [8]. Cybersecurity also contributes to the development of information technology and Internet services by facilitating secure delivery of digital services.

2.5 Web Applications and Cybersecurity

Web applications are one of the most common platforms for providing information and services through the Internet. They are, though, an open target for cyber-attacks since they might contain security vulnerabilities. Web application security is, therefore, a key part of cybersecurity initiatives since advanced techniques and strategies are used to prevent hacking or unauthorized access to data [9].

3. Types of Cross-Site Scripting (XSS) Attacks

Cross-Site Scripting (XSS) attacks are one of the most perilous security threats to modern web applications. Statistics reveal that a vast majority of web applications worldwide are vulnerable to this type of attack. XSS attacks rely on the exploitation of flaws in web-based applications to place malicious code (e.g., JavaScript) inside web pages to have this code executed in targets' browsers unconsciously. XSS attacks lead to the theft of sensitive data such as cookies, sessions of users, and even session hijacking of accounts. Table 1 explains the mechanism of these types briefly.

3.1 Reflected XSS or Non-Persistent XSS Attacks

Also known as a type-1 client-based attack, this is the most common type of XSS attack and is characterized by the ease of detecting the vulnerabilities it exploits. Here, malicious code is injected into a URL, and this link is then sent to the victim. When the victim clicks on the link, the request is sent to the server, which returns the malicious code to the browser for execution. The success of this attack depends on tricking users into clicking on malicious links, making it closely related to social engineering techniques such as Clickjacking [10]. Clickjacking is a technique used to trick users into clicking on malicious links or buttons that appear to be normal user interface elements. Clickjacking relies heavily on dynamic HTML.

3.2 Stored XSS or Persistent XSS Attacks [11]

Also known as type-2 (server/client)-based attacks, this type is more dangerous than reflected attacks; the malicious code is stored in the databases of the targeted site, such as comments or forums. For the users that visit the page that contains the stored code, the malicious code is automatically executed in their browsers. So, these attacks can steal cookies, which contain user data and session information, allowing attackers to impersonate victims or access their accounts.

3.3 DOM based XSS Attack [12]

Also known as a type-0 client-based attack, it relies on exploiting vulnerabilities in the Document Object Model (DOM), a programming interface that allows applications and scripts to access and modify the contents of HTML and XML documents. In DOM-based XSS attacks, the malicious code is executed directly in the victim's browser without requiring interaction with the server, particularly when web applications manipulate the Document Object Model (DOM) using unsanitized user inputs. One of the main vulnerabilities causing this attack is input validation and output encoding, along with the use of insecure JavaScript methods (such as innerHTML and document.write). The DOM of a web page is modified by injecting malicious JavaScript code, which causes it to be executed in the victim's browser. Its challenges are summarized in the difficulty to detect it because it does not require interaction with the server, making it more complex than reflected or stored attacks.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

Table 1: The Operation of XSS Attacks

Attack Type	Execution Location	Interaction with Server	Execution Method	Classification
Reflected XSS (Type-1)	Browser (Client-Side)	The payload is passed through the server but not stored	Payload is injected into a request and reflected in the response	Client-Side Attack relying on server response
Stored XSS (Type-2)	Browser (Client-Side)	The payload is stored on the server but executed in the browser	The payload is stored in the server (e.g., database) and later executed in the browser	Hybrid Attack (Server-Side for storage, Client-Side for execution)
DOM-Based XSS (Type-0)	Browser (Client-Side)	No interaction with the server during execution	JavaScript modifies the DOM directly in the browser	Fully Client-Side Attack

4. Effects of XSS Attacks

XSS attacks can lead to several severe consequences, including (but not limited to):

- Stealing sensitive data, such as login credentials, banking details, and cookies.
- Session hijacking: allowing attackers to take control of victims' accounts.
- Installing malware, such as viruses and Trojans.
- The attacks by social engineering, such as phishing and defrauding users.
- Distributed denial of service (DDoS) attacks: used to flood servers with many fake requests by facilitating botnets for cyberattacks.

5. XSS Protection Mechanisms

Since XSS attacks can be used to install malware, hijack accounts, and steal data, they are a major Internet application security threat. By learning about the different kinds of these attacks and how they work, researchers and developers can come up with good mechanisms for detecting and preventing them. With dynamic threats to data and privacy online, innovation in cybersecurity remains crucial to dealing with such issues.

To date, several defense strategies have been released in order to ward off XSS attacks, including (but not limited to) the following:

5.1 Traditional Methods [13]

- **Input Validation:** Ensure that no malicious code is embedded in any user input.
- **Output Encoding:** Stop any malicious code from running, by making sure that all content in the browser is safe and free of executable code.
- **Implement Content Security Policies (CSP):** Prevent malicious scripts from being downloaded or executed, detect unauthorized code from executing, and identify trusted sources of code.
- **Continuously regular security updates and patches:** To address existing security flaws, address newly found flaws and stop their exploitation.

Filtering the input and encoding the output to stop their injection and execution, is the method of detecting the illegal script from the source code, since XSS attacks are carried out by inserting a malicious script into a vulnerable webpage [6]. The concepts of output encoding and input validation are illustrated in Table (2). Although whitelisting is typically more secure, input filtering (validation) is carried out largely using white list and blacklist values. While a white list greatly lowers the possibility of illegal access, it does not offer perfect defense. On the other hand, output encoding converts special characters to their safe form so stopping hostile code from running, such as # can be replaced with &#35 , < with < , > with > .

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

Table 2: The most common traditional methods

concept	Issue	Potential Consequences	Mitigation Techniques
Input Validation	Failure to properly validate or sanitize user input before processing	Allows injection attacks such as SQL Injection (SQLi) and Cross-Site Scripting (XSS) by accepting malicious payloads	- Implement allow list (whitelist) validation for expected input formats - Use strong data type enforcement (e.g., int for numeric fields) - Apply server-side validation alongside client-side checks
Output Encoding	Failure to properly encode data before rendering it in the browser	Leads to XSS vulnerabilities, where injected scripts execute within the victim's browser, compromising cookies, session tokens, and user credentials	- Use context-aware encoding functions (e.g., htmlspecialchars() in PHP, HTMLEncode() in .NET) - Replace special characters with HTML entity encoding (e.g., <script> → <script>) - Prefer .textContent or .createTextNode instead of .innerHTML in JavaScript

5.2 Source Code Analysis

Programmers utilize static, dynamic, or hybrid code analysis to test the web application as their protection measure.

• Static Code Analysis: does not need the program to be run and can be achieved by just reviewing each line or part of code for any potentially harmful or unwanted activity [14].

• Dynamic Code Analysis: performed by running the program depending on the true input values while the program is executing [14].

• Hybrid Code Analysis: This is where static and dynamic analysis methodologies are combined.

These testing methods, though tedious, are not necessarily successful in detecting new or previously unknown vulnerabilities in the site that can change the content of the webpage.

5.3 Machine Learning Defense Mechanisms

Over the past ten years, attackers have increased their activity due to the growth of the Internet and the ongoing rise in website usage. XSS attack techniques, transmitted as hidden values within URLs, pictures, and scripts, have expanded in variety. Conventional techniques for identifying these attacks struggle to keep up with the constantly changing threats [15]. Consequently, the best approach to achieving precise results that may be adjusted to contemporary attack techniques is to incorporate machine-learning algorithms into attack detection models [16]. Machine learning techniques outperform traditional methods because they are capable of learning from massive, constantly changing data and extracting hidden patterns that are difficult for manual methods to detect. These techniques also can adapt to new attacks by retraining updated data, increasing the efficiency and accuracy of detection systems compared to traditional, static methods that often fail against sophisticated XSS attacks.

- Supervised Learning: To help the model understand the relationship between inputs and outputs, rely on training data that includes the appropriate inputs and outcomes.

Examples: classifying email as "spam" or "not spam" image recognition.

- Unsupervised Learning: This method looks for patterns in the data without assistance rather than using training data with predetermined responses (labeled data).

Examples include identifying odd trends in data (like fraud detection) and segmenting customers in marketing.

- Semi-supervised Learning: A cross between supervised and unsupervised learning, it makes use of a lot of unlabeled data and a small amount of labeled data with known answers.

Examples include speech recognition and text analysis.

- Reinforcement Learning: This method uses a system of rewards and penalties to teach the model to behave almost optimally through trial and error.

Examples include self-driving systems and AI programs in video games.

- Deep Learning: To tackle complicated issues, it uses artificial neural networks.

Examples include machine translation, facial recognition, and medical image analysis.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

6. Related work

The use of machine and deep learning has been widely applied to enhance security mechanisms and mitigate vulnerabilities in various fields, including cybersecurity, anomaly detection, and web security. Many research studies have focused on the importance of applying these techniques to achieve almost optimal results. In this section, a summary of several recent research papers published in global repositories will be presented, which have leveraged machine and deep learning-based techniques to detect and prevent cyber threats (focusing on the context of XSS attacks). These studies (as shown in Table 3) have shown different results according to evaluation metrics such as accuracy, precision, recall, and F1-Score. Subsequently, an outline of the key methodologies, techniques, and results of each study.

6.1 Malviya, Vikas K. et al. (2021) [17]

The study addresses the development of a prototype web browser with an integrated automatic classification feature to detect XSS attacks during browsing. The researchers relied on machine learning algorithms such as Naïve Bayes, Decision Tree, and Random Forest to classify inputs in real time. The models were trained on a dataset containing samples of malicious and valid web requests. A classification module was embedded within the browser to filter requests before they were executed, providing immediate protection for the user. The results showed that the Random Forest model had an accuracy of approximately 97%, outperforming other models. The article contributes to offering an applied framework that combines application security and user experience through integrating a machine-learning model into a browser. Yet, there is a gap wherein the focus is upon traditional models and none of the more advanced techniques, such as neural networks or fusion methods, which would increase the ability to detect advanced attacks even more.

6.2 Odun-Ayo., et al. (2021) [18]

This study focused on developing a real-time XSS detection system in cloud computing environments using deep learning techniques. A deep neural network (DNN) was utilized for processing text features of requests so that responses to attacks could be made real-time as and when they occurred. The result was satisfactory, with 97.8% accuracy in distinguishing attacks from normal requests, indicating the model's effectiveness in practical applications. The research assists in providing a framework for applications that can be embedded in cloud web applications for enhanced real-time security. However, the paper focuses on one model only, and the lack of comparison against hybrid methods or feature fusion techniques has left a research gap regarding the potential for performance improvement through the integration of different data representations.

6.3 Yan, H. et al. (2022) [19]

In this study, a modified convolutional neural network (CNN) model was used in detecting XSS attacks based on improved extraction of textual features from web data. There was a special architectural structure that was adhered to in order to advance the model's understanding of complex contextual patterns linked to attacks, which worked to reduce errors and improve predictive capability. The model accuracy was 99.23%, which is its capacity to distinguish between malicious and benign inputs. The major contribution of the research is the improvement of the traditional CNN architecture by unaltered adaptations to process secure textual information. However, dependence on a single model remains a research gap because the use of feature extraction fusion or hybrid architecture has the potential to improve the potential for preventing sophisticated XSS attacks.

6.4 Gupta, C., Singh, R. K., & Mohapatra, A. K. (2022) [20]

The GeneMiner framework is utilized here, which is a combination of GeneMiner-E for extracting features and GeneMiner-C for classification, aided by an incremental genetic algorithm (IGA) to generate/update the features to learn changing patterns of attack payloads. GeneMiner achieved 98.50% accuracy and an F1 coefficient of 98.03, outperforming techniques such as GAN, RL, RF, and others. The drawback is that the evaluation is based on open data and payloads generated by an improved genetic algorithm (IGA) without field-testing on real traffic.

6.5 Li, X., Wang, T., et al. (2023) [21]

This study presented a BiLSTM-based model with a multi-head attention mechanism for XSS attack detection. BiLSTM supports bidirectional context capture, and the multi-head attention mechanism focuses on the most discriminative features in attack text. The authors employed a large dataset consisting of real XSS attack samples. The model could achieve an F1-score of 98.7%, outperforming traditional models in differentiating benign and malicious samples. There exists a research gap because the experiment merely utilized texts mainly and lacked other features or other levels of representation. This led to subsequent work to utilize feature fusion or hybrid architecture techniques to achieve enhanced performance and generalization.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

6.6 Odeh, A., & Taleb, A. A. (2024) [22]

This study proposes a hybrid XSS detection model that combines RNNs and CNNs to capture HTML/JavaScript sequential dependencies while extracting textual/contextual features, and then combines the outputs of the two networks to make decisions. The hybrid model achieved the best performance compared to CNNs and RNNs alone: 96.74% accuracy and F1 = 96.70. The main benefit is the integration of two deep learning architectures into a single system to improve the balance of precision and recall on web data. Key shortcomings include small data sizes and limited features (not including detailed textual request payloads), and the lack of real-world validation in production web environments or sensitivity/stability analysis of the methodology. This study can be used as a benchmark for the effectiveness of deep fusion, with larger datasets and evaluations needed to generalize the results.

6.7 Bakır, R., & Bakır, H. (2024) [23]

The Swift Detection of XSS Attacks research proposed a hybrid approach to text representation based on Universal Sentence Encoder (USE) and Word2Vec as a feature extraction mechanism, which is then combined and employed in machine learning algorithms (SVM, RF, LR, KNN, MLP) and deep learning models (CNN, LSTM, GRU, Vanilla NN). The results demonstrated the superiority of the hybrid model, achieving a top accuracy of 99.45% using Random Forest, with outstanding performance in F1 and ROC-AUC. The study contributes to demonstrating the power of combining semantic representations at the word and sentence levels to enhance detection accuracy and speed in real time. However, the gap remains in its reliance on limited data and the lack of testing in multiple scenarios or new attacks.

6.8 Luu, G. H., et al. (2024) [24]

In this research, two techniques (CNN and LSTM) were combined into a framework for XSS detection using a hybrid approach. The development of a new dataset and a highly efficient lightweight model capable of learning from complex patterns was the main contribution. The model achieved 99.27% accuracy, outperforming traditional models. However, the gap lies in its reliance on specific data without testing on real traffic or zero-day attacks, as well as its reliance on hand-engineered features that may limit its ability to adapt to changing attacks.

6.9 Okusi, O. (2024) [25]

This paper presents methods for identifying and blocking Cross-Site Scripting (XSS) attacks on web applications using artificial intelligence. Specifically, the study proposes the use of a Deep Forest (DF) model, among other techniques, to address the class imbalance problem, a common problem in XSS datasets. By incorporating rebalancing methods such as the synthetic minority oversampling technique (SMOTE), the study can efficiently enhance the model to effectively detect and prevent XSS attacks. The result confirms that the DF-based method achieves an accuracy of 99.89%. The paper acknowledges that there is a gap in the need for larger datasets and the use of more techniques to deal with imbalance and improve tuning (e.g., parameter, grid/random search) for the experiment, considering that oversampling may lead to overfitting.

6.10 Bacha, N. U., et al. (2024) [26]

This paper presents a combination of traditional machine learning techniques with ensemble methods, based on a hybrid methodology to improve the accuracy of prediction and attack detection. The methodology included the use of multiple models, such as logistic regression (LR), support vector machine (SVM), and random forests (RF), within an ensemble framework aimed at enhancing overall performance and reducing error rates. The study contributed to highlighting the power of hybrid ensemble models compared to individual methods, with results showing a high accuracy of approximately 98% when applied to a realistic security dataset, demonstrating the effectiveness of combining more than one algorithm in providing more detection that is reliable. However, this study focused on traditional ensemble fusion and did not sufficiently address leveraging contextual or deep features of malicious scripts, such as XSS or SQLi attacks. This leaves a research gap that could be filled by integrating deep learning techniques and early integration of textual and contextual features to increase the ability to detect advanced and complex attacks.

6.11 Hu, Z., Zhang, J., and Yang, H. (2025) [27]

The research presented a model based on the fusion of multi-source features using DSCNN to extract local features and Bi-LSTM to extract global features, with a multi-head attention mechanism to intelligently combine them. The study relied on custom segmentation rules for XSS codes and achieved an accuracy of 99.93% and a 99.92% F1-score, outperforming previous methods. The main gap is the limited validation of the model in diverse real-world environments, which may affect its ability to withstand disguised or novel XSS attacks.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

6.12 Oshoiribhor, E.a.J.-O., Adetokunbo (2025) [28]

The research presented a model based on the Logistic Regression algorithm, with feature extraction using TF-IDF and n-grams, and dimensionality reduction via PCA. It was trained on Kaggle data and achieved 99.70% accuracy with 100% recall and 99.36% precision, demonstrating its effective detection while reducing false alarms. However, its limitations lie in its limited use of a traditional model and data structure, which limits its generalization to more complex attacks or real-world environments.

Table 3: The key methodologies, techniques, and results of related works

Ref.	Paper topic	Techniques	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Contributions	Year
[17]	Web Browser Prototype with Embedded Classification	Naïve Bayes, KNN, SVM, Random Forest	99.83	99.82	99.74	99.80	Embedded Classification System in real-time	2021
[18]	Real-time XSS Detection on Cloud Apps	MLP, DBN, LSTM, Ensemble	99.47	99.00	99.00	99.00	Real-time cloud-based detection	2021
[19]	MRBN-CNN for XSS Attack Detection	Modified CNN, MRBN-CNN, ResNet	99.23	99.94	98.53	99.23	Modified CNN for URL-based detection	2022
[20]	GeneMiner Classification for XSS Detection	GeneMiner, Genetic Algorithm	98.50	97.99	96.25	98.03	Genetic algorithm for evolving XSS patterns	2022
[21]	LSTM-Based Detection Scheme for Cloud Computing	Bidirectional LSTM, Multi-head Attention	-	99.32	98.11	98.71	LSTM with attention in cloud environments	2023
[22]	XSSer Hybrid Deep Learning	CNN, RNN	96.74	97.78	95.65	96.70	CNN-RNN hybrid detection approach	2024
[23]	Swift Detection of XSS Attacks (USE, Word2Vec)	Universal Sentence Encoder, Word2Vec	99.45	99.73	99.26	99.49	Enhanced detection accuracy via semantic embeddings	2024
[24]	XSShield Lightweight Hybrid Deep Learning	CNN, LSTM	99.27	99.65	95.61	97.59	Lightweight hybrid model and comprehensive dataset	2024
[25]	Cyber Security Techniques (Deep Forest)	Deep Forest, SMOTE	99.89	99.78	99.9	99.87	Deep Forest addressing class imbalance	2024

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

[26]	Hybrid Ensemble ML for XSS Detection	Logistic Regression, SVM, XGBoost, CatBoost, DNN	99.87	99.8	99.7	99.87	Ensemble models for real-time detection	2024
[27]	Novel XSS Detection Approach Based on Multisource Semantic	DSCNN + BiLSTM + Multihead Attention	99.87	99.96	99.89	99.92	Multisource Semantic Feature Fusion	2025
[28]	XSS-Net: An Intelligent Machine Learning Model	Logistic Regression (with TF-IDF + PCA features)	99.70	100	99.36	99.67	efficient mechanism for extracting and processing features from web inputs	2025

7. Evaluation of Related Work Results, Discussion, Gaps, and Suggestions

After reviewing and examining previous studies (referred to in the previous studies section) related to detecting XSS attacks, it was noted that many methods and techniques used to detect and prevent XSS attacks have evolved over time. These methods range from deep neural networks and traditional machine learning tohybrid approaches that combine more than one model to improve performance.

Review some ideas and methodologies in order to discuss the strengths, weaknesses, and gaps that were identified in this research and to present a future proposal for developing more methodologies that are effective.

- The use of CNN and LSTM with Word2Vec feature extraction techniques. The suggested approach detects XSS attacks with an accuracy of up to 99.4%.
- Created a modified ResNet-CNN neural network that included sophisticated feature extraction techniques like the Network in Network (NiN) model, resulting in an accuracy gain of up to 99.23%.
- Described an attack classification approach that integrates AI approaches with advanced classification algorithms that utilize evolutionary genetics, to enhance the detection of complex attacks.
- Created an LSTM model with a multi-attention mechanism. The suggested approach improved knowledge of XSS detection, with an accuracy of 98.71%.
- Suggested a dynamic detection model with adaptive feature selection multi-agent deep Q-learning, enabling real-time detection of XSS attacks with an accuracy of 98.92% and reducing feature dimensionality for faster response.
- Developed a hybrid of CNN and LSTM concentrated on balancing accuracy and speed, therefore enabling real-time XSS detection at a processing speed above 1000 samples per second.
- Introduced a proposed layered architecture that combines static analysis, rule-based filtering, and browser-side script control to mitigate XSS attacks.

7.1 Discussion of Related Work

This section will conclude:

- Strengths

– Most studies employed hybrid models or deep neural networks, which produced accuracy levels over 98%.
– A limited number of studies used Universal Sentence Encoder and Word2Vec to enhance performance by means of feature extraction methods.
– By means of a model established in some trials, false positives were lowered to 0.13%, improving system dependability.
– Emphasizing computer efficiency, some studies sought to achieve high performance while preserving accurate qualities important for real-time systems.

- Weaknesses

– Many studies lacked real-world testing for their models in actual systems, which limits their practical applicability.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

– Certain models used intricate hybrid techniques, which can affect execution speed in systems with limited resources.

- Research Gaps:

By analyzing previous studies, several research gaps remain:

- The limited application of the proposed models in real-world environments raises questions about their practical applicability.
- The poor handling of data imbalances, especially with the limited number of samples for rare attacks.
- The lack of integration of explainable AI techniques to understand model decisions and improve their reliability.
- The need to develop models capable of operating in real time without sacrificing accuracy.

7.2 Suggestions for Future Work

With this critical analysis, we present a thorough view of how research in this area has grown, as well as recommendations on how to improve the development of more efficient and effective ways for detecting and mitigating XSS attacks.

This study suggests the need for new techniques that balance accuracy with computational efficiency, such as:

- Deploying language processing technology to promote the capacity for recognizing hidden attack patterns.
- Developing hybrid deep learning models that achieve high accuracy while maintaining acceptable processing speed.
- Integrating text-processing techniques to augment understanding of hidden attack patterns.
- Validating the model in practical scenarios to verify its real-world implementation in modern security systems.

The contribution of this research is to provide a comprehensive and structured review of the latest machine learning-based XSS detection techniques during the period 2021–2025, incorporating quantitative comparison across standardized metrics, critical analysis of limitations, and identification of clear research gaps that guide future studies.

8. Conclusion

Cross-site scripting (XSS) is a web application security threat that is quite dangerous, as it can lead to data stealing, hijacking of accounts, and spreading malware. Even though much-enhanced security technology has not been enough to render these attacks outmoded; hence, fresh ideas on detection and prevention should be explored. New techniques with the use of artificial intelligence and machine learning have been shown to be very effective in securing web applications, outperforming the conventional methods in accuracy and attack detection speed. However, it is essential to develop and improve methods to overcome hindrances. The development of systems and technologies incorporating multiple methods to identify threats and attacks will be linked with improved results. Therefore, developing hybrid technologies that incorporate multiple strategies will enhance detection and prevention rates, boosting the performance of detecting and preventing attacks and thus web security. Finally, increasing security consciousness among users and developers is necessary to ensure a secure online environment against growing cyber-attacks.

References

[1] Ö. Aslan, S. S. Aktuğ, M. Ozkan-Okay, A. A. Yilmaz, and E. Akin, "A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions," *Electronics*, vol. 12, no. 6, p. 1333, 2023.

[2] "CWE Top 25 Most Dangerous Software Weaknesses," *MITRE Corporation, SANS Institute*. [Online]. Available: https://www.sans.org/top25-software-errors/. [Accessed: Mar. 10, 2025].

[3] "The OWASP Top 10 Project," *OWASP Foundation*. [Online]. Available: https://owasp.org/www-project-top-ten/. [Accessed: Mar. 11, 2025].

[4] "The OWASP Top 10 Project (GitHub Repository)," *OWASP Foundation*. [Online]. Available: https://github.com/owasp-top/owasp-top-2017. [Accessed: Mar. 11, 2025].

[5] L. Y. Chang and N. Coppel, "Building cyber security awareness in a developing country: Lessons from Myanmar," *Comput. Secur.*, vol. 97, p. 101959, 2020.

[6] O. Ogbanufe, "Enhancing end-user roles in information security: Exploring the setting, situation, and identity," *Comput. Secur.*, vol. 108, p. 102340, 2021.

[7] R. Alhamyani and M. Alshammari, "Machine learning-driven detection of cross-site scripting attacks," *Information*, vol. 15, no. 7, p. 420, 2024.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025

[8] C. Islam, M. A. Babar, R. Croft, and H. Janicke, "SmartValidator: A framework for automatic identification and classification of cyber threat data," *J. Netw. Comput. Appl.*, p. 103370, 2022, doi: 10.1016/j.jnca.2022.103370.

[9] R. M. Wibowo and A. Sulaksono, "Web vulnerability through cross-site scripting (XSS) detection with OWASP security shepherd," *Indones. J. Inf. Syst.*, vol. 3, no. 2, pp. 149–159, 2021.

[10] Crișan *et al.*, "Detecting malicious URLs based on machine learning algorithms and word embeddings," in *Proc. 2020 IEEE 16th Int. Conf. Intell. Comput. Commun. Process. (ICCP)*, Cluj-Napoca, Romania, 2020, pp. 187–193.

[11] G. E. Rodríguez, J. G. Torres, P. Flores, and D. E. Benavides, "Cross-site scripting (XSS) attacks and mitigation: A survey," *Comput. Netw.*, vol. 166, p. 106960, 2020.

[12] S. J. Weamie, "Cross-site scripting attacks and defensive techniques: A comprehensive survey," *Int. J. Commun. Netw. Syst. Sci.*, vol. 15, no. 8, pp. 126–148, 2022.

[13] E. Barlas, X. Du, and J. C. Davis, "Exploiting input sanitization for regex denial of service," in *Proc. 44th Int. Conf. Softw. Eng.*, 2022, pp. 883–895.

[14] X. Wang, Y. Xu, and Z. Sun, "A hybrid dynamic testing technology source code XSS vulnerability detection method," in *2023 IEEE Smart World Congr. (SWC)*, 2023, pp. 1–6.

[15] H. Sarker, "Cyberlearning: Effectiveness analysis of machine learning security modeling to detect cyber-anomalies and multi-attacks," *Internet Things*, vol. 14, p. 100393, 2021.

[16] N. Kshetri *et al.*, "algoXSSF: Detection and analysis of cross-site request forgery (XSRF) and cross-site scripting (XSS) attacks via machine learning algorithms," in *Proc. 12th Int. Symp. Digit. Forensics Secur. (ISDFS)*, 2024, pp. 1–8.

[17] V. K. Malviya, S. Rai, and A. Gupta, "Development of web browser prototype with embedded classification capability for mitigating Cross-Site Scripting attacks," *Appl. Soft Comput.*, vol. 102, p. 106873, 2021.

[18] Odun-Ayo *et al.*, "An implementation of real-time detection of cross-site scripting attacks on cloud-based web applications using deep learning," *Bull. Electr. Eng. Inform*, vol. 10, no. 5, pp. 2442–2453, 2021.

[19] H. Yan *et al.*, "Cross-site scripting attack detection based on a modified convolution neural network," *Front. Comput. Neurosci.*, vol. 16, p. 981739, 2022.

[20] C. Gupta, R. K. Singh, and A. K. Mohapatra, "GeneMiner: A classification approach for detection of XSS attacks on web services," *Comput. Intell. Neurosci.*, vol. 2022, Art. no. 3675821, 2022.

[21] X. Li *et al.*, "An LSTM based cross-site scripting attack detection scheme for cloud computing environments," *J. Cloud Comput.*, vol. 12, no. 1, p. 118, 2023.

[22] Odeh and A. A. Taleb, "XSSer: Hybrid deep learning for enhanced cross-site scripting detection," *Bull. Electr. Eng. Inform*, vol. 13, no. 5, pp. 3317–3325, 2024.

[23] R. Bakır and H. Bakır, "Swift detection of XSS attacks: Enhancing XSS attack detection by leveraging hybrid semantic embeddings and AI techniques," *Arab. J. Sci. Eng.*, pp. 1–17, 2024.

[24] G. H. Luu *et al.*, "XSShield: A novel dataset and lightweight hybrid deep learning model for XSS attack detection," *Results Eng.*, vol. 24, p. 103363, 2024.

[25] O. Okusi, "Cyber security techniques for detecting and preventing cross-site scripting attacks," *World J. Innov. Mod. Technol.*, vol. 8, no. 2, pp. 71–89, 2024.

[26] N. U. Bacha *et al.*, "Deploying hybrid ensemble machine learning techniques for effective cross-site scripting (XSS) attack detection," *Comput. Mater. Continua*, vol. 81, no. 1, 2024.

[27] Z. Hu, J. Zhang, and H. Yang, "XSS Attack Detection Based on Multisource Semantic Feature Fusion," *Electronics*, vol. 14, no. 6, p. 1174, 2025.

[28] E. Oshoiribhor and J.-O. Adetokunbo, "XSS-Net: An Intelligent Machine Learning Model for Detecting Cross-Site Scripting (XSS) Attack in Web Application," 2025.

DOI: https://doi.org/10.54216/JCIM.170210
Received: April 12, 2025 Revised: June 28, 2025 Accepted: August 22, 2025