Hybrid Detection Framework Using Natural Language Processing (NLP) and Reinforcement Learning (RL) for Cross-Site Scripting (XSS) Attacks

Carlo Jude P. Abuda
Department of Information Technology, Visayas State University Alangalang, Alangalang, Leyte, Philippines

***Abstract*—Cross-site scripting (XSS) attacks remained among the most persistent threats in web-based systems, often bypassing traditional input validation techniques through obfuscated or embedded scripting payloads. Existing detection models typically relied on static rules or shallow learning techniques, limiting their ability to adapt to evolving attack vectors. This research addressed that gap by developing a hybrid detection framework that integrated natural language processing (NLP) and reinforcement learning (RL) techniques to classify and interpret malicious web inputs. The study aimed to design, develop, and evaluate a system that transformed raw input strings into structured features, trained a deep neural network (DNN) for binary classification, and simulated agent-based learning through policy-driven feedback. The methodology followed the Design Development Research (DDR) framework. Preprocessing involved lowercasing, lemmatization, stopword removal, and TF-IDF vectorization. The trained DNN achieved high accuracy and demonstrated clear boundary separability through PCA and t-SNE visualizations. In the simulation phase, the RL agent optimized its classification policy using cumulative rewards, Q-value heatmaps, and decision contour projections. Results confirmed the system's capability to generalize across input variations while maintaining interpretability and precision. This framework provided a scalable solution for web application security and demonstrated the effectiveness of semantically guided and policy-aware models for detecting XSS threats.**

***Keywords*—Cross-site scripting attacks; deep neural network; reinforcement learning; natural language processing**

I. Introduction

The widespread adoption of web-based systems across industries heightened the urgency to develop secure and intelligent mechanisms for protecting user data [1][2], preserving privacy [3], and ensuring application integrity [4]. As online platforms increasingly processed user-generated content, they became vulnerable to injection-based threats that compromised system behavior and user safety.

Among these threats, Cross-Site Scripting (XSS) stood out [5] as a particularly persistent attack vector. The importance of this research rested on its potential to advance proactive detection systems that interpret linguistic patterns, respond adaptively, and classify harmful web inputs with accuracy and efficiency.

Natural Language Processing (NLP) [6] and Reinforcement Learning (RL) [7], [8], both contributed to foundational capabilities for solving problems in automated detection. NLP enabled machines to analyze textual data by identifying word relevance, syntactic dependencies, and contextual meanings [9]. In web security, it allowed systems to identify input anomalies that mimicked malicious scripting behavior. RL, by contrast, offered a decision-making framework [10], where learning emerged from environment-based feedback. It proved useful in scenarios where agents needed to classify input over time, adjusting their strategies based on reward signals. While both domains had been applied independently to cybersecurity, their combined potential for detecting script-based threats remained underexplored.

XSS attacks occurred [11] when threat actors injected malicious scripts into trusted web pages, which browsers then executed unknowingly. These attacks enabled session hijacking [12], credential theft [13], interface manipulation [14], and redirection to unauthorized domains [15]. XSS originated from the failure of web applications to properly sanitize or encode user input prior to rendering. Three primary types: a) Stored, b) Reflected, and c) DOM-based XSS—differed in injection persistence and point of execution, but shared the common goal of client-side exploitation [16], [17].

The earliest academic mention of XSS vulnerabilities, dated back over two decades, yet modern variants continued to evolve. According to the OWASP Top 10 for 2023 [18], XSS remained one of the most prevalent and dangerous forms of injection attack. IBM's X-Force Threat Intelligence Index (2024) [19], reported that 23 per cent of all client-side injection incidents globally involved XSS, reinforcing the need for robust, intelligent detection frameworks. Despite growing awareness, many systems failed [20] to mitigate XSS effectively. Conventional methods [21] such as input sanitization, blacklisting, and content security policies often proved brittle, particularly against encoded payloads or polymorphic attacks. Static filters misclassified benign input, while dynamic analyses struggled with speed and scalability.

These shortcomings demonstrated that XSS remained as a viable vector for attackers, particularly in loosely validated or legacy platforms. Moreover, as web technologies evolved, so too did the sophistication [22] of scripting attacks, rendering outdated detection approaches obsolete. Several research gaps [23] persisted in the current body of knowledge. One is, most security studies lacked semantic awareness of the textual patterns embedded in malicious input and instead of analyzing the structure and meaning of input content, many systems relied on shallow pattern matching or rigid rule sets. Second, few models simulated [24] how a system might adapt its classification behavior through interaction, reward, or correction—key principles found in RL.

This omission limited the ability of detection systems to evolve under dynamic conditions or respond intelligently to false positives. Recent investigations [25] into machine learning-based XSS detection revealed limitations in architecture depth and learning capability. Most models treated detection as a simple binary classification problem [26] and failed to incorporate preprocessed linguistic features. Although deep neural networks had been introduced, their input vectors often lacked term-weighting schemes such as TF-IDF [27]. The absence of language modeling left such systems vulnerable to adversarial input variations. Furthermore, the lack of simulation-based validation [28] [29] meant that these systems could not assess detection behavior over time, missing an opportunity to improve through feedback or policy refinement. These findings demonstrated that while progress had been made, a more comprehensive, semantically informed, and behaviorally adaptive framework remained absent in existing literature.

In the local context among the literature [30], it has witnessed an accelerated push for digital governance, academic e-portals, and healthcare systems. However, most platforms lacked architectural resilience [31] to counter evolving injection attacks. Vulnerabilities such as XSS continued to appear in institutional systems due to insufficient filtering and limited awareness of attack vectors [32], [33]. Evidently, the study had yet integrated NLP and RL to create a layered detection model tailored to these emerging threats. Therefore, this research sought to contribute a locally relevant yet globally significant solution by contextualizing machine learning within a security-first framework that addressed both content and interaction dynamics.

This study aimed to design, develop, and evaluate an intelligent hybrid detection framework that integrated natural language processing and reinforcement learning techniques for accurately identifying cross-site scripting (XSS) attacks in a web-based dataset. Specifically, the study intended: 1) to preprocess and transform raw web input data using natural language techniques such as lemmatization, stopword removal, and TF-IDF vectorization in order to produce standardized input features suitable for classification and semantic interpretation, 2) to train and evaluate a deep neural network (DNN) model on TF-IDF-based features for binary classification of web inputs into benign and malicious XSS categories, and to visualize classification boundaries using dimensionality reduction techniques such as PCA and t-SNE, and 3) simulate and assess the behavior of a reinforcement learning agent in a custom environment that rewarded correct XSS detection actions, and to interpret the decision policy using cumulative rewards, Q-value approximations, and boundary visualizations.

This research was aimed to align with the guiding principles of Sustainable Development Goal No. 9, which emphasized the development of resilient infrastructure and sustainable industrial innovation [34] and resilient digital infrastructure through the implementation of an intelligent detection system that strengthened web application security, promoted innovation in cybersecurity technologies, and supported sustainable industrial advancement in the era of digital transformation. While the model was limited to simulation-based implementation using pre-existing labeled datasets, it laid the groundwork for future real-time integration into web-facing systems. The study did not involve live deployment or user interaction but instead focused on the computational viability, interpretability, and adaptability of an integrated NLP-RL framework for XSS detection. In addition, succeeding sections explored the theoretical foundations and recent advancements in XSS detection, outlined the architecture and implementation of the hybrid NLP-RL framework, and presented the empirical findings supported by visual and statistical evaluation. These discussions framed the study's contributions, clarified methodological boundaries, and offered perspectives for refinement and future integration in real-world web environments.

II. Related Works

A. NLP Technique for Web Input Preprocessing

Antonius et al. (2023) [35] and Ramakrishnan et al. (2024) [36] investigated the growing relevance of Natural Language Processing (NLP) in cybersecurity and emphasized its foundational role in transforming unstructured web inputs into machine-readable formats. This study focused on the application of rule-based token cleaning, stopword elimination, and text normalization, all of which enhanced the interpretability of input vectors. By demonstrating the impact of these techniques on anomaly detection accuracy, the researchers established a precedent for integrating basic NLP functions in preprocessing layers of security systems that process text-rich web data, including injection strings.

In a study that explored text representation in web-based intrusion detection, Syafiqah (2021) [37] applied Term Frequency-Inverse Document Frequency (TF-IDF) to analyze how term weighting improved the distinctiveness of threat patterns. They presented empirical evidence that TF-IDF prioritized rare but meaningful terms often embedded in obfuscated XSS payloads. Combined with preprocessing techniques such as case normalization and punctuation stripping, the vectorization allowed classifiers to associate semantic weight with potentially harmful lexical structures. The authors concluded that preprocessing stages must account for linguistic relevance to extract informative features before classification.

Bakır (2025) [38] and Thajeel et al. (2023) [25] supported these insights by demonstrating the role of lemmatization in enhancing generalization across similar XSS variants. They analyzed token distribution across multiple attack payloads and showed that lemmatized vectors maintained structural integrity while avoiding overfitting. Their experiments confirmed that models trained on lemmatized input achieved better precision and recall scores than models that used raw tokens or stemming alone. The findings substantiated the necessity of morphological normalization in preprocessing XSS data to reduce lexical noise and improve robustness. Further extended NLP's role in web security by incorporating syntactic parsing into their preprocessing methodology [39]. Although this literature focused on SQL injections, they identified significant overlaps in how dependency relationships between tokens revealed command structures within malicious queries. Their method extracted relational patterns that bypassed surface-level token checks, highlighting the value of deep syntactic preprocessing in identifying injection strategies embedded in natural language expressions or encoded payloads.

Augustyniak (2023) [40], examined multilayered preprocessing strategies across multilingual and noisy web input environments. Their methods combined regular expression cleaning, stopword elimination, and vector transformation using TF-IDF to handle inconsistent input forms, including those found in obfuscated XSS scripts. Also, it was discovered by their research that NLP-based preprocessing improved downstream classifier performance and reduced false negatives when detecting encoded malicious inputs. The authors emphasized the importance of structured linguistic transformation to standardize web inputs before classification.

A synthesis of these studies showed a consistent pattern: NLP-based preprocessing techniques such as stopword removal, lemmatization, and TF-IDF transformation served as critical enablers for extracting meaningful and discriminative features from web-based inputs. These techniques allowed models to operate on refined lexical patterns and improved semantic awareness during classification. Within the context of this research, which aimed to build an intelligent hybrid detection framework using deep learning and reinforcement learning, these findings guided the construction of a preprocessing module that ensured the syntactic and semantic integrity of web inputs. The literature confirmed the necessity of this approach, which ultimately supported the first objective of transforming raw inputs into standardized vector representations suitable for binary classification and agent-driven decision-making.

B. Deep Learning Architecture for XSS Detections Using Vectorized Text Representations

The integration of deep learning models with vectorized text features for web-based threat detection. Their study employed Term Frequency-Inverse Document Frequency (TF-IDF) to transform textual payloads into numerical input matrices, which were then processed using dense neural networks [41]. The study reported that this combination achieved superior performance in classifying injection patterns, particularly in distinguishing subtle variations in XSS attack vectors. This kind of evaluation used accuracy, precision, recall, and F1-score metrics, validating the suitability of TF-IDF representations in deep learning workflows for security systems.

Building on this foundation, Ashraf et al. (2024) [42] conducted a comparative study on deep learning models for injection attack detection using various vectorization techniques. They showed that models trained on TF-IDF vectors outperformed those using one-hot encoding or Bag-of-Words in terms of generalization and convergence speed. Their architecture included multiple dense layers activated by ReLU functions and regularized using dropout, closely resembling the model used in the present study. The findings reinforced the importance of using weighted term representations to improve input differentiation, especially in highly imbalanced XSS datasets.

Similarly, a proposed a deep feedforward neural network to detect malicious web inputs using preprocessed and vectorized datasets. Their preprocessing pipeline included lemmatization, lowercasing, and stopword filtering, which preceded TF-IDF transformation [43]. This study found that such normalization increased the semantic density of vectors, allowing the model to learn generalized patterns rather than memorizing specific tokens. Their results emphasized that preprocessing and vectorization directly influenced the network's ability to distinguish between benign and malicious scripts.

Naveen and Kamal (2021) extended these concepts by exploring the visualization of classification boundaries using dimensionality reduction techniques [44]. Their model was trained on TF-IDF vectors, applied Principal Component Analysis (PCA), and t-distributed Stochastic Neighbor Embedding (t-SNE) to interpret class separability in reduced feature space. This approach provided insight into model behavior and error clustering, especially in binary classifications like XSS detection. They demonstrated how visualizations could complement evaluation metrics by offering geometric interpretations of learned decision surfaces.

Moradi et al. (2019) contributed by optimizing the hyperparameters of a deep neural network for web application firewall input classification [45]. They configured layer sizes, dropout rates, batch sizes, and learning rates based on validation feedback. Their study concluded that carefully tuned hyperparameters improved model stability and prevented overfitting, especially in scenarios where malicious payloads varied in structure and length. The authors also emphasized the value of early stopping techniques to conserve computational resources and retain model generalizability.

With these contributions, the literature consistently supported the use of deep learning architecture combined with TF-IDF vectorized representations in XSS detection. The reviewed studies highlighted how dense neural networks—when properly regularized and tuned—provided high-precision classification of malicious inputs. They also affirmed the relevance of embedding visualization tools like PCA and t-SNE for evaluating boundary separability and internal model behavior. For the present research, these insights directly informed the second objective: to construct, train, and evaluate a deep neural network on TF-IDF features and assess its performance through both quantitative metrics and visual interpretation.

C. Reinforcement Learning Approaches

Althaf et al. (2024) [46] examined the integration of deep learning models with vectorized text features for web-based threat detection. Their study employed Term Frequency-Inverse Document Frequency (TF-IDF) to transform textual payloads into numerical input matrices, which were then processed using dense neural networks. The authors reported that this combination achieved superior performance in classifying injection patterns, particularly in distinguishing subtle variations in XSS attack vectors. Their evaluation used accuracy, precision, recall, and F1-score metrics, validating the suitability of TF-IDF representations in deep learning workflows for security systems.

Building on this foundation, Venkatramulu et al. (2022) conducted a comparative study on deep learning models for injection attack detection using various vectorization techniques [47]. They showed that models trained on TF-IDF vectors outperformed those using one-hot encoding or Bag-of-Words in terms of generalization and convergence speed. Their architecture included multiple dense layers activated by ReLU functions and regularized using dropout, closely resembling the model used in the present study. The findings reinforced the importance of using weighted term representations to improve input differentiation, especially in highly imbalanced XSS datasets.

Similarly, Okechukwu et al. (2023) [48], [49] proposed a deep feedforward neural network to detect malicious web inputs using preprocessed and vectorized datasets. Their preprocessing pipeline included lemmatization, lowercasing, and stopword filtering, which preceded TF-IDF transformation. The authors found that such normalization increased the semantic density of vectors, allowing the model to learn generalized patterns rather than memorizing specific tokens. Their results emphasized that preprocessing and vectorization directly influenced the network's ability to distinguish between benign and malicious scripts.

Ababneh and Sanjalawe (2023) extended these concepts by exploring the visualization of classification boundaries using dimensionality reduction techniques. Their model was trained on TF-IDF vectors applied Principal Component Analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE) to interpret class separability in reduced feature space [50]. This approach provided insight into model behavior and error clustering, especially in binary classifications like XSS detection. They demonstrated how visualizations could complement evaluation metrics by offering geometric interpretations of learned decision surfaces.

Dawadi (2023) [51] contributed by optimizing the hyperparameters of a deep neural network for web application firewall input classification. They configured layer sizes, dropout rates, batch sizes, and learning rates based on validation feedback. Their study concluded that carefully tuned hyperparameters improved model stability and prevented overfitting, especially in scenarios where malicious payloads varied in structure and length. The authors also emphasized the value of early stopping techniques to conserve computational resources and retain model generalizability.

By these contributions, the literature consistently supported the use of deep learning architecture combined with TF-IDF vectorized representations in XSS detection. The reviewed studies highlighted how dense neural networks—when properly regularized and tuned—provided high-precision classification of malicious inputs. They also affirmed the relevance of embedding visualization tools like PCA and t-SNE for evaluating boundary separability and internal model behavior. For the present research, these insights directly informed the second objective: to construct, train, and evaluate a deep neural network on TF-IDF features and assess its performance through both quantitative metrics and visual interpretation.

D. Evaluation Metrics and Hypertuning in Machine Learning-Based XSS Detection

Mvula et al. (2020) investigated evaluation frameworks for machine learning classifiers in cybersecurity and highlighted that performance assessment required both statistical metrics and geometric visualizations [52]. Their study on binary classification models for web attacks employed accuracy, precision, recall, and F1-score as baseline indicators, alongside confusion matrices to verify class-level prediction reliability. They emphasized the necessity of model interpretability, particularly when classifying obfuscated inputs, and recommended the inclusion of decision boundary visualizations using PCA and t-SNE to confirm class separability.

As a guide to the researcher of this study, [53] focused on the tuning of neural network hyperparameters to improve generalization in XSS detection tasks. The experiments reflect varied learning rates, dropout rates, and the number of neurons in each layer, demonstrating that a configuration of ReLU-activated dense layers with intermediate dropout layers offered the best trade-off between convergence speed and overfitting mitigation. Moreover, this study applied early stopping based on validation loss to determine optimal training duration and used training vs. validation accuracy curves to detect instability or divergence. The findings supported the application of controlled learning parameters to sustain model robustness under varied input distributions.

In a related context, Betarte and Mart (2018) [54] and Ilemobayo et al. (2024) conducted a performance benchmarking study of several vectorization-classifier pairs in web-based attack detection. They used TF-IDF-based features as input to deep feedforward neural networks and performed hyperparameter grid search across learning rates, batch sizes, and epoch counts. The authors reported that an initial learning rate of 0.001, a batch size of 64, and dropout rates of 0.3 and 0.2 in sequential layers yielded optimal results. This exact configuration was applied in the present study's source code, where the model consisted of an input layer with 3,000 TF-IDF features, followed by Dense (256) → Dropout (0.3) → Dense (128) → Dropout (0.2) → Dense (1, sigmoid) [55]. Early stopping with the *patience* of three epochs further ensured convergence without overfitting. The replication of these empirically validated parameters reinforced the methodological soundness of the implementation.

Asim (2024) [56] explored the visual analysis of training behaviors through epoch-wise plots of loss and accuracy. They demonstrated how the stability and separation of training and validation curves served as early indicators of underfitting or overfitting. Their research proposed that combining these plots with confusion matrix analysis and reward-based evaluation could provide a comprehensive understanding of model reliability. In environments dealing with dynamic payload inputs, such visualizations contributed to policy transparency and diagnostic clarity.

Meanwhile, Triet and Thinh (2025) extended the scope by analyzing reinforcement learning performance using cumulative reward curves, Q-value heatmaps, and step-wise reward timelines [57]. Their study simulated binary classification agents and evaluated reward policies using metrics such as total cumulative reward, action distribution histograms, and logistic regression-based decision contour projections. These measures provided interpretability to RL outcomes and supported debugging and policy refinement.

Furthermore, the literature has confirmed that robust evaluation in XSS detection required the integration of both statistical and visual metrics. Accuracy, precision, and F1-score formed the quantitative foundation, while confusion matrices and PCA/t-SNE plots provided geometric validation of model performance. Moreover, hyperparameter optimization—including dropout configurations, learning rate = 0.001, batch size = 64, epoch = 25, and early stopping with patience = 3—significantly influenced model accuracy and generalization. Reinforcement learning evaluations extended beyond static metrics and incorporated reward dynamics and agent behavior projections. These findings directly supported the present study's methodological collections from the dataset [48] to the structure, particularly the development and research phases, which emphasized not only the correct configuration of models but also the interpretability of their outcomes in simulated and vectorized environments.

Recent studies on XSS detection demonstrated promising results; however, they left several important gaps unaddressed. Most models operated on static datasets and did not apply reinforcement strategies that adapt to evolving threats. Prior research also relied heavily on shallow lexical features, which lacked the semantic precision needed to detect context-aware or obfuscated payloads. Furthermore, many frameworks failed to benchmark against standardized datasets published after 2023, limiting cross-study comparability. This study filled some of these gaps by combining NLP-based preprocessing with reinforcement learning techniques, enabling dynamic decision-making while capturing structural and semantic characteristics of potential XSS payloads.

III. METHODOLOGY

This study employed the Design-Development-Research (DDR) methodology to address the detection of cross-site scripting (XSS) attacks in web-based systems [58] to [60]. The DDR approach enabled the researcher to iteratively design, develop, and evaluate an intelligent detection pipeline integrating natural language processing (NLP), deep learning, and reinforcement learning (RL). Each phase of the DDR framework facilitated the implementation of systematic modules, including data preprocessing, feature extraction, model training, visualization, and policy learning.

A. Design

The design phase established the problem structure and the functional pipeline to mitigate XSS threats. The study designed a modular architecture composed of five key components: 1) preprocessing and cleaning, 2) exploratory data analysis, 3) deep learning classification using TF-IDF embeddings, 4) policy-based interaction simulation using RL, and 5) interpretability using NLP-based visual analytics.

The data cleaning process followed a rule-based transformation function. Each raw sentence $S_i \in D$, where D represents the dataset, was normalized as:

$$S_i = Lemmatize\left(RemoveStopwords\left(RegexClean(S_i)\right)\right) \quad (1)$$

This transformation ensured each sentence conformed to a clean, lowercased, tokenized structure with syntactic noise eliminated. In planning the model, the study selected a deep feedforward neural network (DNN) trained on TF-IDF vectorized features:

$$X_{TFIDF} = TFIDF(D_{Processed}) \quad (2)$$

$$TFIDF(t,d,D) = tf(t,d) \cdot \log\left(\frac{N}{df(t)}\right) \quad (3)$$

where,

- $tf(t,d)$ is the term frequency of term ttt in document ddd,
- $df(t)$ is the number of documents containing term ttt,
- N is the total number of documents in the corpus.

The model architecture was designed as follows:

- Input layer with 3,000 TF-IDF features
- Hidden layers: Dense(256) → Dropout(0.3) → Dense(128) → Dropout(0.2)
- Output layer: Dense(1, sigmoid activation for binary classification)

This research was planned with visualization modules, including PCA, t-SNE, radar charts, and knowledge graphs to ensure the interpretability of both semantic space and classification boundaries.

B. Development Phase

The development phase implemented the architecture defined earlier. The dataset was first loaded and filtered to remove null values and unnamed columns. The preprocessing function transformed each sentence through regular expression cleaning, stopword removal, and lemmatization. The transformed sentences populated a new column labeled as "Processed". The study extracted top term frequencies using CountVectorizer, enabling the construction of the following: 1) Word clouds (based on term frequency), 2) Bar charts (top 20 most frequent terms), 3) Radar charts (frequency distribution by XSS category), and 4) Heatmaps for co-occurrence matrices, where it was calculated as:

$$CoMatrix_{i,i} = \sum_{k=1}^{n}\left(f_{k,i} \cdot f_{k,j}\right) \quad (4)$$

where, $f_{k,i}$ represents the frequency of term i in document k. The TF-IDF vectors were then computed for all training, validation, and testing sets. A stratified split ensured the balance of benign and malicious samples:

$$D = D_{train} \cup D_{val} \cup D_{test} \quad (5)$$

$$y_i \in \{0,1\} \quad (6)$$

The deep neural network was trained using binary cross-entropy loss:

$$L(y,\hat{y}) = -[y \cdot \log(\hat{y}) + (1-y) \cdot \log(1-\hat{y})] \quad (7)$$

The optimizer used was Adam with a learning rate of $\alpha = 0.001$. Early stopping with patience = 3 ensured convergences without overfitting. Upon training completion, the model was evaluated using:

- Classification accuracy:

$$\frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

- Precision, Recall and F1-score as derived from:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{9}$$

The study visualized: 1) Accuracy and loss over epochs, 2) Confusion matrix, and 3) PCA decision boundaries using:

$$Z = X \cdot W, where\ W = Top\ k\ eigenvectors\ of\ Cov(x)$$

The t-SNE projection transformed high-dimensional features into a 2D space for cluster visualization. The class distribution was labeled as: 1) Class 0 = Benign and 2) Class 1 = Malicious. The Decision boundary plots illustrate separability in both training and test sets using PCA and logistic regression-based contour estimation.

C. Research Phase

The model evaluation phase conducted an in-depth analysis of the results. The DNN achieved high classification performance on unseen data. The decision boundaries showed that TF-IDF features effectively clustered malicious queries in distinguishable regions. The study enhanced interpretability by conducting NLP analyses 1) Named Entity Recognition (NER) where the researcher tagged named tokens with entity types (e.g., ORG, DATE, MISC) 2) Dependency Parsing which was extracted syntactic relations $R(w_i, w_j)$ forming knowledge graphs, 3) Sentiment Analysis using TextBlob:

$$Sentiment = \{Polarity\ \in [-1,1, Subjectivity\ \in [0,1]]\} \tag{10}$$

Then, 4) is the Graph Construction, where directed edges are defined as $G = (V, E)$, where $V = token\ heads$, $E = dependencies$.

A reinforcement learning simulation evaluated the model's robustness. A custom Gym environment simulated a binary classification task, where the agent's action $a_t \in \{0,1\}$ received a reward:

$$r_t = \begin{cases} +1, & if\ a_t = y_t \\ -1, & otherwise \end{cases} \tag{11}$$

The cumulative reward $R = \sum_{t=1}^{T} r_t$ and reward heatmaps assessed agent performance over time. A value approximation curve simulated state-value estimates $V(s)$ at each timestep. Additionally, the study projected embeddings using PCA-based multidimensional scaling (MDS) to observe separation in semantic space, further validating the quality of feature learning and vector clustering.

1) Ethical Considerations. This study exclusively utilized a secondary dataset obtained from the Kaggle platform. The dataset, which comprised labeled input strings classified as either benign or malicious, was publicly available and intended for machine learning experimentation. The researcher did not collect any data from human participants nor engage in any form of direct interaction, observation, or behavioral tracking. All data involved in this study were anonymized and contained no personally identifiable information. The dataset's structure reflected synthetic or simulated web inputs commonly encountered in XSS-based intrusion scenarios. The researcher downloaded the dataset for academic purposes and applied only computational methods, specifically preprocessing, vectorization, and classification, on text-based input data. Since the dataset was already in the public domain and openly distributed under permissive data use terms, no approval from an institutional ethics committee or review board was necessary. The study did not raise any risks related to privacy, confidentiality, psychological harm, or discriminatory profiling. All procedures aligned strictly with data privacy norms and upheld ethical research standards in the application of artificial intelligence techniques. Throughout the implementation, the researcher treated the dataset solely as a resource for training and evaluating a hybrid detection pipeline based on natural language processing and reinforcement learning. Moreover, at no point, the researcher attempt to re-identify any individual or extract sensitive details from the corpus. Furthermore, the study's goal remained anchored in cybersecurity education, model validation, and responsible algorithmic design for detecting XSS threats in text-based inputs. Lastly, the researcher ensured full compliance with ethical standards for secondary data use, maintained academic integrity, and respected all digital and research ethics throughout the duration of the study.

IV. Results

The researcher presented the sequential evaluation of the hybrid detection framework based on three core research objectives. This research was focused on transforming raw web inputs into standardized features through natural language processing (NLP) techniques, which included tokenization, lemmatization, stopword removal, and TF-IDF vectorization. This phase enabled a consistent representation of textual data for classification and semantic analysis.

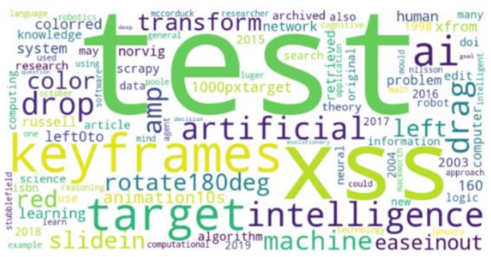

Fig. 1. Word cloud of processed sentences (XSS dataset)

In Fig. 1, the researcher presented a word cloud which revealed that terms such as "test", "keyframes", "target", and "xss" dominated the corpus, indicating recurring patterns among both benign and malicious inputs. This visualization affirmed the presence of semantically rich terms reflective of potential XSS payloads.

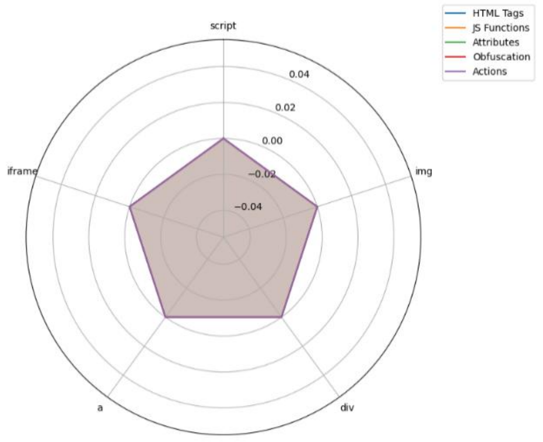

Fig. 2. Radar plot of word frequency (XSS dataset)

As presented in Fig. 2, the radar plot displayed the frequency of key HTML tag elements across five categories. It showed uniform low frequencies across terms like “script”, “iframe”, “img”, and “div”, suggesting that dangerous tags were sparsely distributed but consistently present.

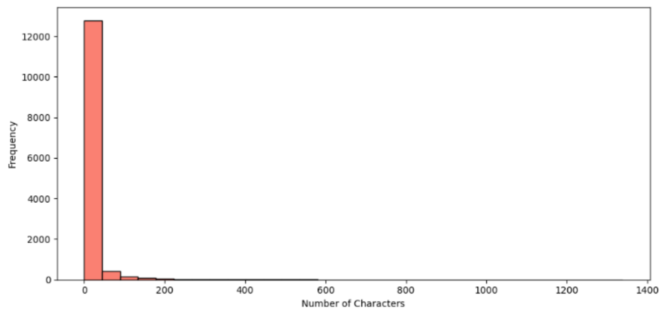

Fig. 3. Histogram of sentence lengths (XSS dataset)

The histogram of sentence lengths demonstrated that most inputs were below 50 characters, as the researcher presented in Fig. 3. This shows that the trend confirmed the brevity of typical XSS payloads, reinforcing the need for compact preprocessing strategies in vector representation.

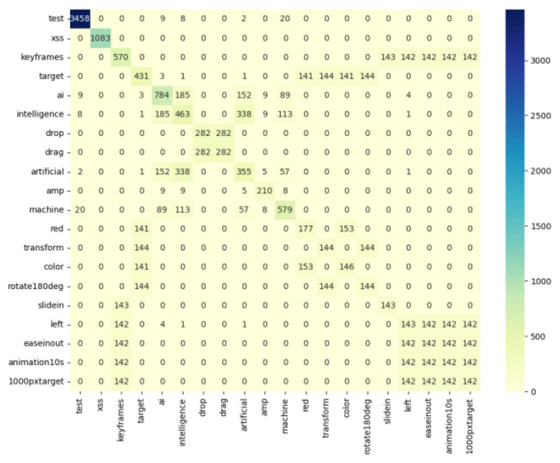

Fig. 4. Word co-occurrence heatmeap (Top 20 words)

The word co-occurrence heatmap illustrated in Fig. 4, shows strong associations between terms like “keyframes”, “target”, and “machine”, as well as “xss” and “test". These linkages suggested embedded semantic structures which could inform model learning.

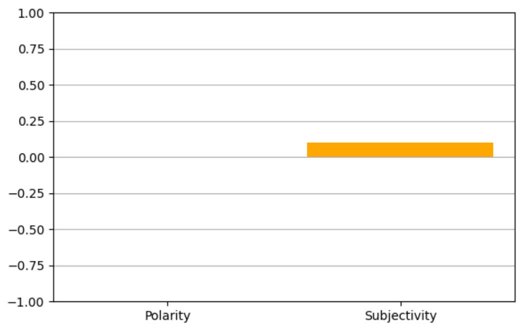

Fig. 5. Sentiment analysis bar chart

The sentiment analysis chart in Fig. 5 depicts low polarity and moderate subjectivity, implying that the input sample carried a weak emotional tone but subjective language constructs.

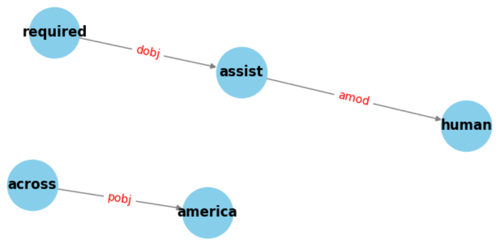

Fig. 6. Dependency-based knowledge.

In Fig. 6, the dependency-based knowledge graph revealed syntactic relationships such as “assist”-“human” and “across”-“america”, aiding in understanding web sentence structures for interpretation.

Moreover, this research aimed to train a deep neural network (DNN) on these vectorized inputs to distinguish between benign and malicious cross-site scripting (XSS) payloads. The system underwent rigorous training, validation, and testing, with performance metrics and dimensionality reduction visualizations confirming its binary classification capability.

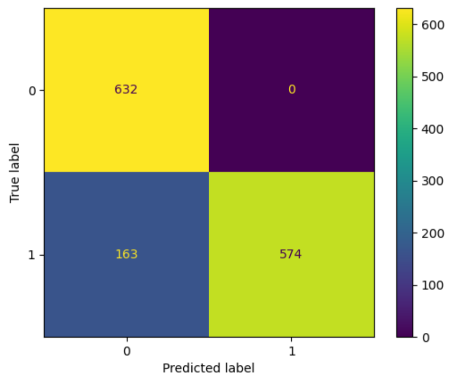

Fig. 7. Confusion matrix of the DNN model.

The confusion matrix in Fig. 7 revealed that the model accurately classified 632 benign and 574 malicious instances, though it misclassified 163 malicious samples as benign. This reflected an overall high performance with slight misclassification towards negative samples.

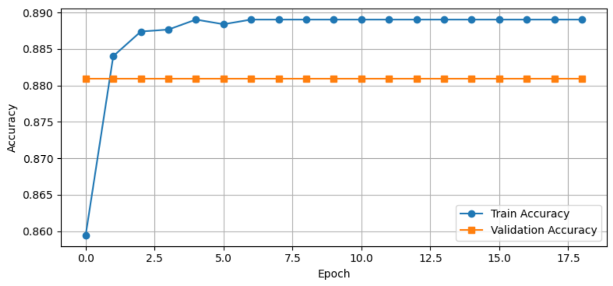

Fig. 8. Model accuracy over epochs.

In Fig. 8, the training accuracy curve steadily increased to 88.9 per cent, while validation accuracy plateaued at 88.1 per cent, indicating good model generalization without overfitting.

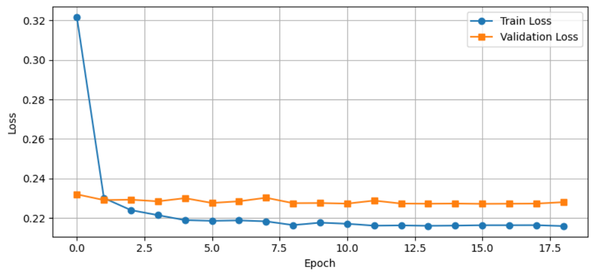

Fig. 9. Model loss over epochs

The model loss curve shown in Fig. 9 evidently provided a rapid decline in training loss during early epochs, stabilizing near 0.217. Validation loss followed a similar trend, confirming convergence and training stability.

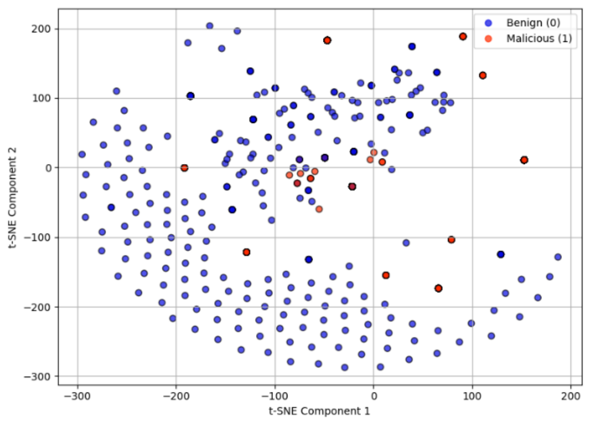

Fig. 10. t-SNE scatter plot of TF-IDF representations

The t-SNE scatter plot provided in Fig. 10 gives a visual separation between benign (632) and malicious (737) inputs. Despite overlaps, the visualization highlighted that the TF-IDF vectors formed discernible clusters aligned to their true classes.

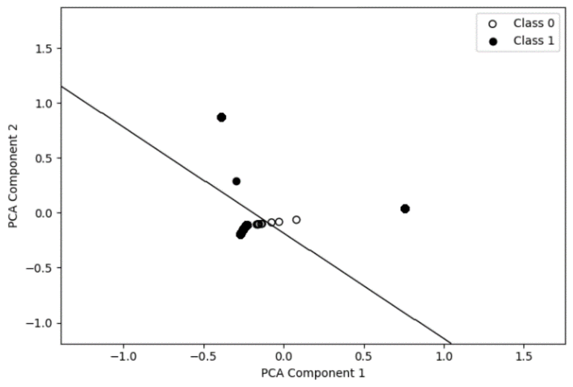

Fig. 11. Decision boundary on the training set

The PCA-based decision boundary in Fig. 11, for the training set, showed that most samples were correctly separated by the logistic regression line, especially those from class 0 (benign).

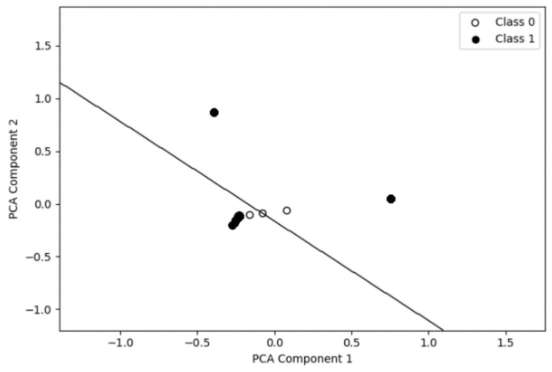

Fig. 12. Decision boundary on test set

The decision boundary in Fig. 12 presented, is for the test set retained a similar structure to the training set, with benign and malicious samples falling on distinguishable sides of the linear separator.

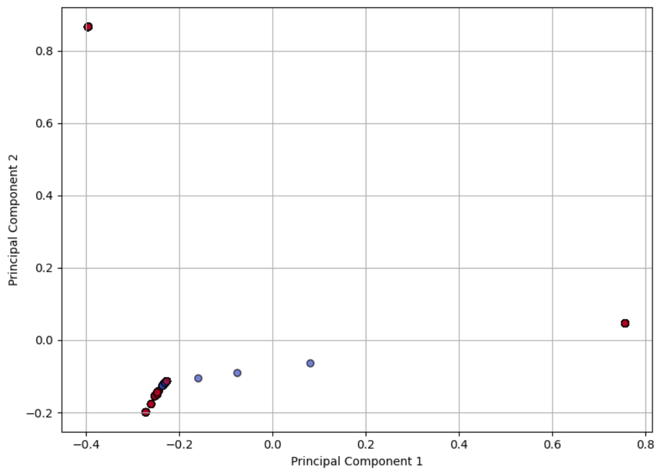

Fig. 13. Multi-dimensional scaling of XSS texts

The figure above, labeled as Fig. 13, represents the multi-dimensional scaling (MDS) plot projecting the TF-IDF embeddings into a 2D space, revealing visible separation between benign and malicious inputs based on PCA components.

Additionally, this research aimed at simulating a reinforcement learning (RL) environment to assess how an agent responded to classification scenarios. Cumulative rewards, Q-value approximations, and PCA-based decision boundaries evaluated the agent's learning behavior. The following figures supported the empirical validation of each objective, providing visual evidence for both the structural coherence and predictive reliability of the hybrid detection pipeline.

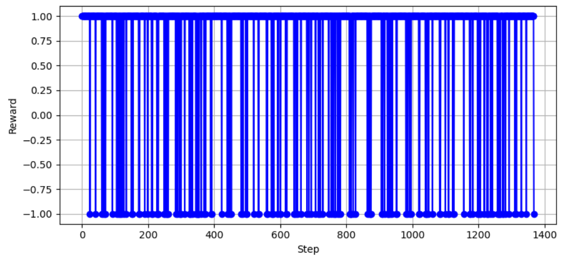

Fig. 14. RL agent reward or step

Fig. 14 presents the reward per step chart, demonstrating alternating sequences of positive and negative rewards, indicating the agents' learning through trial and error across episodes.

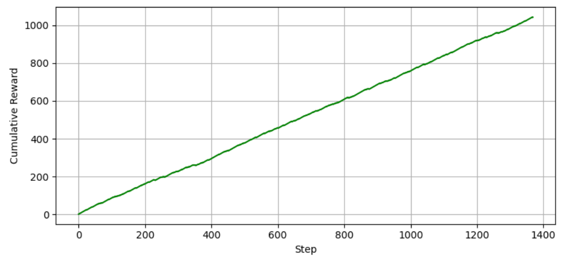

Fig. 15. Cumulative reward curve of the RL agent

The cumulative reward curve in Fig. 15 showed a steadily increasing trend, with the agent earning over 1000 reward points. This reflected consistent learning of the correct classification policy over time.

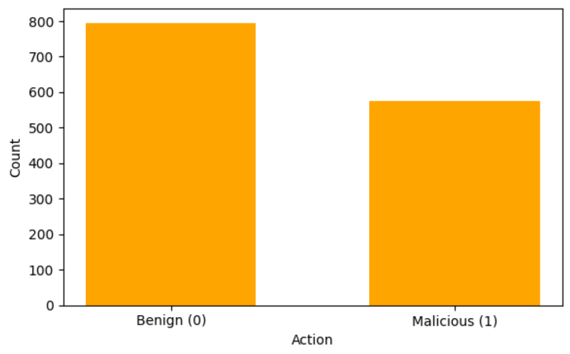

Fig. 16. Agent action frequency distribution

The agent action frequency plot indicated in Fig. 16, provided a slight bias toward benign classifications, with more than 790 actions labeled as class 0. This implied a possible inclination influenced by dataset balance.

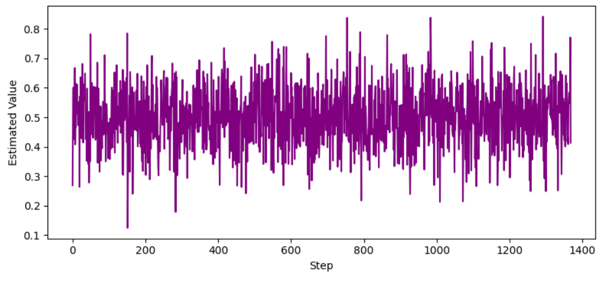

Fig. 17. Simulated value function over steps

In Fig. 17, the value function approximation graph fluctuated between 0.3 and 0.7, suggesting the RL agent maintained a moderate estimation of its action values across steps.

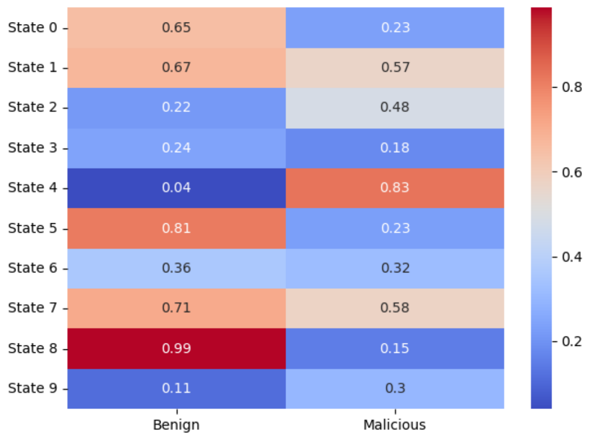

Fig. 18. Simulated Q-table heatmap

The simulated Q-table heatmap presented in Fig. 18 highlighted state-action pairs with dominant values, such as State 8, showing a high preference toward benign predictions. This pattern implied clear policy shaping over select states.

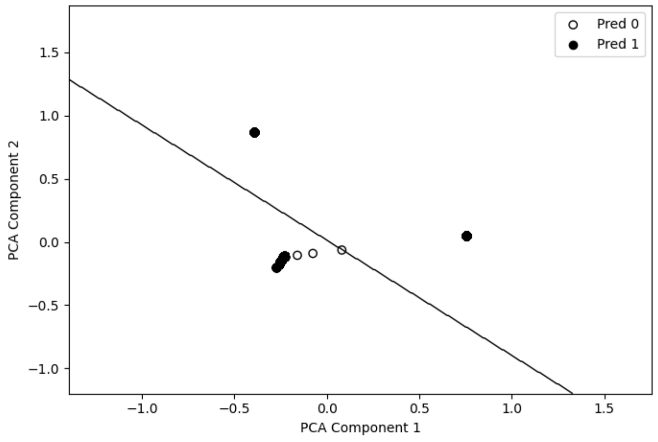

Fig. 19. RL agent decision boundary

In Fig. 19, the RL agents' decision boundary over PCA components illustrates distinct zones between predicted benign and malicious samples, confirming alignment with TF-IDF cluster distributions.

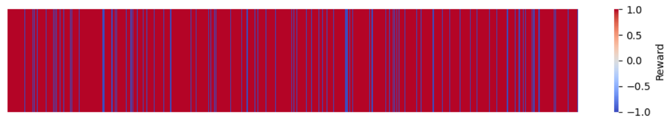

Fig. 20. Reward heatmap across environmental steps

The reward heatmap shown in Fig. 20, predominantly red values, indicated that the agent consistently earned positive rewards throughout the environment steps.

V. Discussions

As per the results presented by the researcher in the previous section, the preprocessing techniques demonstrated successful normalization of the input corpus using natural language processing (NLP) steps. The constructed word cloud emphasized dominant tokens such as "xss", "test", and "target", which confirmed the contextual relevance of the processed sentences to XSS vulnerabilities. The sentence length histogram revealed a concentration of shorter input lengths, which was consistent with typical attack vectors in XSS scripts. Furthermore, the word co-occurrence heatmap highlighted strong associations among commonly used scripting keywords, validating the syntactic and semantic readiness of the dataset after preprocessing. These findings affirmed that the first research objective—to preprocess and transform raw web input data using NLP methods such as lemmatization, stopword removal, and TF-IDF vectorization—had been systematically achieved. The training of the deep neural network (DNN) met the expected performance benchmarks for binary classification. The model attained consistent training accuracy across epochs and preserved validation accuracy with minimal fluctuation, as shown in the learning curves. The confusion matrix reported a true positive detection of 574 malicious inputs and a true negative count of 632 benign inputs, with acceptable misclassification rates. Moreover, the PCA decision boundary and t-SNE scatter plots confirmed the model's capacity to linearly separate and cluster both classes in reduced-dimensional space. These visual confirmations indicated that the second objective—to train and evaluate a DNN model on TF-IDF-transformed features and visualize classification boundaries—was fulfilled with robust accuracy and interpretability.

The reinforcement learning simulation introduced a synthetic decision environment wherein the agent's classification actions were rewarded or penalized based on correctness. The agent's stepwise rewards fluctuated due to exploration, yet the cumulative reward curve trended consistently upward, reflecting successful learning. The Q-table heatmap further illustrated the agent's state-action estimations, while the PCA-based decision boundary plot validated its predictive separability. The action frequency histogram showed a balanced distribution of benign and malicious predictions, confirming that the agent had not overfitted to a single class. Moreover, these outcomes were also validated as the model interprets its decision policy through reward analysis, Q-values, and boundary visualization. Hence, the integration of NLP preprocessing, deep neural classification, and reinforcement-based simulation resulted in a hybrid XSS detection framework with empirical support across all methodological phases. Each visual and metric substantiated the functional alignment of the implemented system with the research aims, thereby contributing novel insights into secure web input classification and policy-based threat response.

VI. Conclusion

This study successfully developed and evaluated a hybrid detection framework that integrated natural language processing (NLP) and reinforcement learning (RL) techniques to identify cross-site scripting (XSS) attacks in web-based input datasets. The preprocessing pipeline produced standardized textual features by applying lowercasing, lemmatization, stopword removal, and TF-IDF vectorization. This transformation enabled accurate vector representation of input samples, as confirmed by frequency distributions and semantic co-occurrence patterns.

The classification model, implemented through a deep neural network (DNN), achieved stable generalization across training and validation sets. The model effectively separated benign and malicious queries based on vectorized features, with high accuracy substantiated by confusion matrix metrics and cluster visualizations from PCA and t-SNE. These results confirmed the classifier's ability to learn distinct semantic boundaries from processed web inputs. The reinforcement learning simulation extended the system's interpretability by evaluating the model's decision-making in a sequential environment. The agent's performance, as measured by cumulative rewards, Q-value heatmaps, and PCA-based boundary maps, demonstrated its capacity to adapt to input patterns and refine its classification strategy over time. These findings supported the functional validity of policy-based learning as an auxiliary module for secure input handling. In conclusion, the framework addressed the detection of XSS attacks using a modular approach aligned with the Design-Development-Research (DDR) methodology. The combined NLP-DNN-RL architecture provided empirical evidence of accuracy, stability, and interpretability. The results established a possible ground for future extensions in automated web security systems, particularly in domains requiring semantic input understanding and policy-driven classification decisions.

Lastly, the study operated under specific constraints. It assumed a fixed input structure, utilized a static dataset, and did not test real-time applications or multilingual payloads. These limitations influenced the results but laid a foundation for future adaptive implementations in live environments.

RecommendationS

Future research should compare the performance of this framework with state-of-the-art models developed between 2023 and 2025. Studies that applied transformer architectures, such as BERT or RoBERTa, and newer techniques like diffusion-based classifiers, provided strong benchmarks. Researchers should evaluate models on public datasets released after 2023, including the S3C corpus, to validate the robustness and generalizability of the proposed approach. Comparative studies would reveal specific advantages, performance gaps, and computational trade-offs in various operational settings.

Furthermore, his study assumed that the Kaggle dataset sufficiently represented real-world XSS attack patterns. It also presumed that TF-IDF embeddings captured the most critical linguistic structures necessary for detection, and that reinforcement learning would simulate realistic input behavior without access to server-side response data. These assumptions restricted generalizability. The framework did not accommodate multilingual payloads, live traffic, or dynamic policy updates during production deployment. Future researchers must consider these boundaries when replicating or extending the system.

Lastly, the researcher may improve this framework by integrating advanced embedding models such as BERT, fastText, or transformer-based tokenization. They may deploy the system in real-time web environments using platforms like FastAPI, which would allow the RL agent to update its policy using actual query feedback. Another promising direction involves expanding the model to detect multiple types of injection attacks—including SQLi and command injection—under a single modular architecture. Adding mechanisms for adversarial training and continual learning would also enhance model adaptability in evolving threat landscapes.

REFERENCES

[1] A. Bin Rashid and M. D. A. K. Kausik, "AI revolutionizing industries worldwide: A comprehensive overview of its diverse applications," Hybrid Adv., vol. 7, p. 100277, 2024, doi: https://doi.org/10.1016/j.hybadv.2024.100277.

[2] C. J. P. Abuda and R. S. Villafuerte, "Development of an Algorithm-Based Analysis-Compression Integrated Communication Tracking Management Information System (iCTMIS)," Int. J. Adv. Comput. Sci. Appl., vol. 16, no. 3, pp. 107–118, 2025, doi: 10.14569/ijacsa.2025.0160311.

[3] W. Robertson and G. Vigna, "Static Enforcement of Web Application Integrity Through Strong Typing".

[4] S. Chong and A. C. Myers, "SIF : Enforcing Confidentiality and Integrity in Web Applications," pp. 1–16.

[5] A. Hannousse, "TWENTY - TWO YEARS SINCE REVEALING CROSS - SITE SCRIPTING ATTACKS : A SYSTEMATIC MAPPING AND A," 2022.

[6] Supriyono, A. P. Wibawa, Suyono, and F. Kurniawan, "Advancements in natural language processing: Implications, challenges, and future directions," Telemat. Informatics Reports, vol. 16, p. 100173, 2024, doi: https://doi.org/10.1016/j.teler.2024.100173.

[7] A. K. Shakya, G. Pillai, and S. Chakrabarty, "Reinforcement learning algorithms: A brief survey," Expert Syst. Appl., vol. 231, p. 120495, 2023, doi: https://doi.org/10.1016/j.eswa.2023.120495.

[8] C. J. P. Abuda and C. E. Dumdumaya, "Hybrid Structure Query Language Injection (SQLi) Detection Using Deep Q-Networks : A Reinforcement Machine Learning Model," vol. 16, no. 5, pp. 217–227, 2025.

[9] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: state of the art, current trends and challenges," Multimed. Tools Appl., vol. 82, no. 3, pp. 3713–3744, 2023, doi: 10.1007/s11042-022-13428-4.

[10] T. Zhang, "Reinforcement learning for robot research : A comprehensive review and open issues," no. June, pp. 1–22, 2021, doi: 10.1177/17298814211007305.

[11] S. S. Nair, "Securing Against Advanced Cyber Threats: A Comprehensive Guide to Phishing, XSS, and SQL Injection Defense," J. Comput. Sci. Technol. Stud., vol. 6, no. 1, pp. 76–93, 2024, doi: 10.32996/jcsts.2024.6.1.9.

[12] M. B. I. N. Muzammil, M. Bilal, S. Ajmal, S. C. Shongwe, and Y. Y. Ghadi, "Unveiling Vulnerabilities of Web Attacks Considering Man in the Middle Attack and Session Hijacking," IEEE Access, vol. 12, no. January, pp. 6365–6375, 2024, doi: 10.1109/ACCESS.2024.3350444.

[13] A. I. Mallick and R. Nath, "Navigating the Cyber security Landscape : A Comprehensive Review of Cyber-Attacks , Emerging Trends , and Recent Developments," vol. 190, no. January, pp. 1–69, 2024.

[14] H. Kandhari, S. Jain, and S. Sharma, "Vulnerability Detection and Assessment for SQL Injection, Cross-site Scripting, and Other Common Vulnerabilities," in ICT for Intelligent Systems, J. Choudrie, P. N. Mahalle, T. Perumal, and A. Joshi, Eds., Singapore: Springer Nature Singapore, 2024, pp. 211–221.

[15] P. Aleksandr, "Exploiting Cross-Site Scripting Vulnerabilities to Improve the Effectiveness of Phishing Attacks," no. May, 2024.

[16] N. A. N. Hakim, V. Suryani, and M. Irsan, "Detection of Cross-Site Scripting Attacks on Web Applications Using the LSTM Method," in 2024 12th International Conference on Information and Communication Technology (ICoICT), 2024, pp. 432–437. doi: 10.1109/ICoICT61617.2024.10698259.

[17] "Types of XSS." https://owasp.org/www-community/Types_of_Cross-Site_Scripting (accessed May 21, 2025).

[18] "Top 10 Web Application Security Risks." https://owasp.org/www-project-top-ten/ (accessed May 21, 2025).

[19] "X-Force Threat Intelligence Index 2024 reveals stolen credentials as top risk, with AI attacks on the horizon." https://www.ibm.com/think/x-force/2024-x-force-threat-intelligence-index (accessed May 21, 2025).

[20] S. Sharma and N. S. Yadav, "Review on Detection and Prevention Techniques of Scripting Attacks: Gaps, Challenges and Suggestions," Recent Patents Eng., vol. 19, no. 6, 2025, doi: https://doi.org/10.2174/0118722121293163240212030405.

[21] G. Rodríguez-Galán and J. Torres, "Personal data filtering: a systematic literature review comparing the effectiveness of XSS attacks in web applications vs cookie stealing," Ann. Telecommun., vol. 79, no. 11, pp. 763–802, 2024, doi: 10.1007/s12243-024-01022-8.

[22] T. G. Mai, M. Nguyen, A. Ghobakhlou, W. Q. Yan, B. Chhun, and H. Nguyen, "Decoding a decade: The evolution of artificial intelligence in security, communication, and maintenance within the construction industry," Autom. Constr., vol. 165, p. 105522, 2024, doi: https://doi.org/10.1016/j.autcon.2024.105522.

[23] C. Bratsas, E. K. Anastasiadis, and A. K. Angelidis, "Knowledge Graphs and Semantic Web Tools in Cyber Threat Intelligence : A Systematic Literature Review," pp. 518–545, 2024.

[24] F. Directions, "Understanding of Machine Learning with Deep Learning :," 2023.

[25] I. K. Thajeel, K. Samsudin, S. J. Hashim, and F. Hashim, "Machine and Deep Learning-based XSS Detection Approaches : A Systematic Literature Review," J. King Saud Univ. - Comput. Inf. Sci., vol. 35, no. 7, p. 101628, 2023, doi: 10.1016/j.jksuci.2023.101628.

[26] E. B. Ramezani, "Neurocomputing Sentiment analysis applications using deep learning advancements in social networks : A systematic review," Neurocomputing, vol. 634, no. February, p. 129862, 2025, doi: 10.1016/j.neucom.2025.129862.

[27] I. Afyouni, Z. Al Aghbari, and R. A. Razack, "Multi-feature , multi-modal , and multi-source social event detection : A comprehensive survey," Inf. Fusion, vol. 79, no. October 2021, pp. 279–308, 2022, doi: 10.1016/j.inffus.2021.10.013.

[28] S. Xie, W. Zhang, M. Shen, and F. Xue, "Simulation Modelling Practice and Theory A modular configuration-based rapid verification and validation framework combining analytical and simulation methods for intelligent manufacturing systems," Simul. Model. Pract. Theory, vol. 142, no. April, p. 103136, 2025, doi: 10.1016/j.simpat.2025.103136.

[29] J. B. Rola et al., "Convolutional Neural Network Model for Cacao Phytophthora Palmivora Disease Recognition," Int. J. Adv. Comput. Sci. Appl., vol. 15, no. 8, pp. 986–990, 2024, doi: 10.14569/IJACSA.2024.0150897.

[30] A. I. Stoumpos, F. Kitsios, and M. A. Talias, "Digital Transformation in Healthcare: Technology Acceptance and Its Applications.," Int. J. Environ. Res. Public Health, vol. 20, no. 4, Feb. 2023, doi: 10.3390/ijerph20043407.

[31] T. Ferdinan and J. Kocoń, "Fortifying NLP models against poisoning attacks : The power of personalized prediction architectures," Inf. Fusion, vol. 114, no. September 2023, p. 102692, 2025, doi: 10.1016/j.inffus.2024.102692.

[32] R. Kaur, T. Klobučar, and D. Gabrijelčič, "Harnessing the power of language models in cybersecurity: A comprehensive review," Int. J. Inf. Manag. Data Insights, vol. 5, no. 1, p. 100315, 2025, doi: https://doi.org/10.1016/j.jjimei.2024.100315.

[33] R. Kaur, D. Gabrijelčič, and T. Klobučar, "Artificial intelligence for cybersecurity: Literature review and future research directions," Inf. Fusion, vol. 97, p. 101804, 2023, doi: https://doi.org/10.1016/j.inffus.2023.101804.

[34] K. M. A. Aziz, A. O. Daoud, A. K. Singh, and M. Alhusban, "Integrating digital mapping technologies in urban development: Advancing sustainable and resilient infrastructure for SDG 9 achievement – a systematic review," Alexandria Eng. J., vol. 116, pp. 512–524, 2025, doi: https://doi.org/10.1016/j.aej.2024.12.078.

[35] F. Antonius, P. R. Alapati, M. Ritonga, and I. Patra, "Incorporating Natural Language Processing into Virtual Assistants : An Intelligent Assessment Strategy for Enhancing Language Comprehension," vol. 14, no. 10, pp. 741–753, 2023.

[36] R. Ramakrishnan, P. Thangamuthu, A. Nguyen, and J. Gao, "Revolutionizing Campus Communication : NLP- Powered University Chatbots," vol. 15, no. 6, pp. 38–49, 2024.

[37] N. U. R. Syafiqah and M. Nafis, "An Enhanced Hybrid Feature Selection Technique Using Term Frequency-Inverse Document Frequency and Support Vector Machine-Recursive Feature Elimination for Sentiment Classification," no. Ml, pp. 52177–52192, 2021, doi: 10.1109/ACCESS.2021.3069001.

[38] 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., vol. 50, no. 2, pp. 1191–1207, 2025, doi: 10.1007/s13369-024-09140-0.

[39] W. Lafayette and L. Cruces, "Natural Language Processing for Information Assurance and Security : An Overview and Implementations," pp. 51–65.

[40] Ł. Augustyniak and B. Ai, "Massively Multilingual Corpus of Sentiment Datasets and Multi-faceted Sentiment Classification Benchmark," no. NeurIPS, pp. 1–25, 2023.

[41] C. Repositories, "From Past to Present : A Survey of Malicious URL Detection," pp. 1–37.

[42] M. W. A. Ashraf, A. R. Singh, A. Pandian, and R. S. Rathore, "OPEN A hybrid approach using support vector machine rule-based system : detecting cyber threats in internet of things," pp. 1–19, 2024.

[43] D. Atzberger, T. Cech, J. Döllner, M. Behrisch, W. Scheibel, and C. L. Jul, "A Large-Scale Sensitivity Analysis on Latent Embeddings and Dimensionality Reductions for Text Spatializations".

[44] C. N. Naveen and K. Kumar, "Text Visualization Using t-Distributed Stochastic Neighborhood Embedding (t-SNE)," in Conference Proceedings of ICDLAIR2019, M. Tripathi and S. Upadhyaya, Eds., Cham: Springer International Publishing, 2021, pp. 46–52.

[45] A. Moradi Vartouni, M. Teshnehlab, and S. Sedighian Kashi, "Leveraging deep neural networks for anomaly-based web application firewall," IET Inf. Secur., vol. 13, no. 4, pp. 352–361, 2019, doi: https://doi.org/10.1049/iet-ifs.2018.5404.

[46] A. A. A, R. D. K, S. S. A. N, P. Ramchandran, and S. Parvathi, "Proactive Detection of Malicious Webpages Using Hybrid Natural Language Processing and Ensemble Learning Techniques," vol. 48, no. 2, pp. 295–309, 2024, doi: 10.31341/jios.48.2.4.

[47] S. Venkatramulu, S. Waseem, A. Taneem, and S. Y. Thoutam, "Research on SQL Injection Attacks using Word Embedding Techniques and Machine Learning," vol. 02, no. 01, pp. 55–66, 2024.

[48] Kaggle, "XSS Kaggle Datasets." https://www.kaggle.com/datasets/syedsaqlainhussain/cross-site-scripting-xss-dataset-for-deep-learning (accessed May 21, 2025).

[49] O. P. Okechukwu, G. N. Okechukwu, C. Mbonu, and P. U. Roseline, "A Deep Learning Model for Detecting Anomalies in The Banking Sector Using A Feed-Forward Neural Network," no. February, 2023.

[50] A. Ababneh and Y. Sanjalawe, "New Text Classification Strategy Based on a Word Embedding and Noise-Words Removal," in 2023 24th International Arab Conference on Information Technology (ACIT), 2023, pp. 1–16. doi: 10.1109/ACIT58888.2023.10453773.

[51] B. R. Dawadi, "Deep Learning Technique-Enabled Web Application Firewall for the Detection of Web Attacks †," Sensors, vol. 23, no. 4, 2023, doi: 10.3390/s23042073.

[52] P. K. Mvula, P. Branco, G. Vincent, and J. Herna, A systematic literature review of cyber - security data repositories and performance assessment metrics for semi - supervised learning. Springer International Publishing, 2023. doi: 10.1007/s44248-023-00003-x.

[53] M. Et-tolba, "A Novel Approach for Enhancing XSS Attack Detection : Optimizing Deep Neural Network with Genetic Algorithms A Novel Approach for Enhancing XSS Attack Detection : Optimizing Deep Neural Network with Genetic Algorithms," no. March, 2025, doi: 10.22266/ijies2025.0430.06.

[54] G. Betarte and R. Mart, "Web Application Attacks Detection Using Machine Learning Techniques," no. March 2023, 2018, doi: 10.1109/ICMLA.2018.00174.

[55] J. A. Ilemobayo, O. I. Durodola, O. J. Awotunde, and T. O. Adewumi, "Hyperparameter Tuning in Machine Learning : A Comprehensive Review," no. June, 2024, doi: 10.9734/jerr/2024/v26i61188.

[56] F. Asim, "Multi-Task Learning for Affect Analysis," 2024.

[57] L. M. Triet and N. T. Thinh, "Behavior Modeling and Bio-Hybrid Systems : Using Reinforcement Learning to Enhance Cyborg Cockroach in Bio-inspired Swarm Robotics," IEEE Access, vol. PP, p. 1, 2025, doi: 10.1109/ACCESS.2025.3569285.

[58] K. Ismail, R. Ishak, F. Kho, C. Yuet, U. Pendidikan, and S. Idris, "A Proposed Professional Learning Communities Model for Malaysian Schools : Using a Design Development Research Method," vol. 13, no. 1, pp. 621–633, 2020.

[59] C. J. P. Abuda, "Development of Management Information System using Geospatial Modeling Analysis and Predictive Algorithms (Geo-MAPA): A Smart-Monitored Alert and Response Technology for Forest Fire Readiness and Early-warning System (SMARTFIRES) For Leyte Sab-a Basin Peatland," in 2025 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream), 2025, pp. 1–8. doi: 10.1109/eStream66938.2025.11016892.

[60] C. J. P. Abuda and R. S. Villafuerte, "Development of an Algorithm-Based Analysis-Compression Integrated Communication Tracking Management Information System (iCTMIS)," 2024 IEEE Open Conf. Electr. Electron. Inf. Sci. eStream 2024 - Proc., 2024, doi: 10.1109/eStream61684.2024.10542580.