ISSN 2089-8673 (Print) | ISSN 2548-4265 (Online)
Volume 13, Issue 3, December 2024

NETWORK INTRUSION DETECTION USING TRANSFORMER MODELS AND NATURAL LANGUAGE PROCESSING FOR ENHANCED WEB APPLICATION ATTACK DETECTION

Wowon Priatna[1], Irwan Sembiring[2], Adi Setiawan[3], Iwan Setyawan[4], Joni Warta[5], Tyastuti Sri Lestari[6]

[1,5,6]Informatika, Universitas Bhayangkara Jakarta Raya
[2,3,4]Doktor Ilmu Komputer, Universitas Kristen Satya Wacana

email: wowon.priatna@dsn.ubharajaya.ac.id[1], irwan@uksw.edu[2], adi.setiawan@uksw.edu[3], iwan@uksw.edu[4], joniwarta@dsn.ubharajaya.ac.id[5], tyas@ubharajaya.ac.id[6]

Abstract

The increasing complexity and frequency of web application attacks demand more advanced detection methods than traditional network intrusion detection systems (NIDS), which rely heavily on predefined signatures and rules, limiting their effectiveness against novel threats. This study proposes a novel approach by integrating Transformer models with Natural Language Processing (NLP) techniques to develop an adaptive and intelligent intrusion detection framework. Leveraging the Transformer's capacity to capture long-term dependencies and NLP's ability to process contextual information, the model effectively addresses the dynamic and diverse nature of web application traffic. Using the CSIC 2010 dataset, this study applied comprehensive preprocessing, including tokenization, stemming, lemmatization, and normalization, followed by text representation techniques such as Word2Vec, BERT, and TF-IDF. The Transformer-NLP architecture significantly improved detection performance, achieving 85% accuracy, 95% precision, 83% recall, 84% F1 score, and an AUC of 0.95. Friedman and t-test validations confirmed the robustness and practical significance of the model. Despite these promising results, challenges related to computational complexity, dataset scope, and generalizability to broader network attacks remain. Future research should focus on expanding the dataset, optimizing the model, and exploring broader cybersecurity applications. This study demonstrates a significant advancement in detecting complex web application attacks, reducing false positives, and improving overall security, offering a viable solution to growing cybersecurity challenges.

Keywords: NLP, Intrusion Detection, Transformer, Web Application Attack, Machine Learning

Received: 10-07-2024 | **Revised:** 05-10-2024 | **Accepted:** 09-11-2024
DOI: https://doi.org/10.23887/janapati.v13i3.82462

INTRODUCTION

The security of web applications has become a paramount concern in the current digital era, especially with the rise of attacks targeting vulnerabilities in web applications[1].As technology evolves and internet usage grows, web applications become more vulnerable to attacks like SQL injection, Cross-Site Scripting (XSS), and Denial of Service (DoS). These attacks compromise data integrity, confidentiality, and service availability, with rising frequency and complexity over time [2].

Web applications are often the first point of entry for attackers, exploiting vulnerabilities like SQL injection and Cross-Site Scripting (XSS) to gain unauthorized access or inject malicious scripts. These vulnerabilities highlight the need for robust detection mechanisms specifically tailored to web applications. Therefore, this study focuses on detecting attacks targeting web applications, recognizing this as a critical aspect of maintaining overall network security[3]. Research on network intrusion detection systems (NIDS) has explored various methodologies to counteract these threats[4]. For instance, Research [5] provides a comprehensive overview of existing detection systems specifically designed to monitor web traffic, comparing the capabilities of systems like AppSensor, PHPIDS, ModSecurity, Shadow Daemon, and AQTRONIX WebKnight. Other studies have focused on input validation techniques to prevent intrusions, such as the approach detailed in Research [6], which emphasizes input validation against web application attacks. Additionally, Research [7] has developed an intrusion detection model to mitigate cyber-attacks, data breaches, and identity theft, aiding in effective risk management.

Traditional approaches to network intrusion detection rely heavily on predefined signatures and rules, which limits their effectiveness in detecting new or unknown variants of attacks[8]. This rigidity necessitates more adaptive solutions. A popular approach to overcoming these limitations involves the use of machine learning (ML) and artificial intelligence (AI) to create more intelligent and flexible intrusion detection systems [9]. Machine learning models, such as Random Forest and Support Vector Machines, have been successfully employed to detect anomalies in network traffic[10]. Some studies have advanced this further by combining ensemble learning with NLP-based methods, as indicated in Research [11], to enhance the detection models' effectiveness. However, even these sophisticated methods face challenges in handling the highly dynamic and diverse data generated by web applications. The complexity of web application traffic stems from frequent updates, varying user inputs, and increasingly sophisticated attack vectors, making it difficult for traditional models to adapt in real time[12]. For example, studies have shown that vulnerabilities such as SQL injection and Cross-Site Scripting (XSS) are among the most common attack types, with SQL injection accounting for approximately 65% of web application attacks in 2022, according to OWASP reports[13]. The evolving nature of these vulnerabilities, along with their high frequency, underscores the critical need for more adaptive detection systems capable of handling the sheer volume and variety of data produced by modern web applications.

For instance, research [14] utilizing traditional ML models demonstrated moderate success in detecting known intrusions, but performance degraded significantly when applied to unknown or zero-day attacks. Moreover, approaches based on signature detection or anomaly detection often suffer from high false positive rates, making them impractical for real-world applications. To address these challenges, this study proposes a novel approach that integrates advanced Transformer models with NLP techniques to better capture the complex patterns and contextual information inherent in web application data[15]. While NLP techniques have been widely adopted, the deep integration of NLP with Transformer architectures for web application intrusion detection is a relatively unexplored area, offering a more nuanced detection of web attacks. This combination allows for the detection of complex[16], evolving web threats that are often missed by traditional machine-learning models.

Recent advancements in deep learning, particularly the development of the Transformer model by Vaswani et al., offer a promising solution[17]. The Transformer's ability to capture long-range dependencies in sequential data and process this information efficiently through an attention-based architecture provides a robust framework for addressing the complexities of web application data. The application of Transformer models in network intrusion detection presents new opportunities for developing more adaptive and sophisticated systems capable of identifying a wide range of web attacks[18]. Research has shown that Transformers are particularly effective in analyzing patterns and anomalies within network data, leading to improved detection rates of complex attacks that are often missed by conventional methods[19].

Unlike previous models that focus on static or homogeneous data sets, the proposed research utilizes both Transformer models and NLP techniques to handle the diverse and ever-evolving nature of web application data. This approach differs significantly from existing studies, which often rely on traditional machine learning models or shallow integration of NLP techniques. Our research leverages the Transformer's ability to handle intricate patterns within the data, providing a significant advancement over existing methods. By combining the strengths of NLP in text representation and the deep learning capabilities of Transformers, this study introduces a unique framework that significantly enhances detection performance, particularly for sophisticated web attacks. While earlier studies [11][20][21] employed NLP for enhancing feature extraction in intrusion detection, this research integrates these methods more deeply within a Transformer-based architecture, representing a novel approach to the field.

The novelty of this study lies in its dual integration of NLP techniques and Transformer models for web application intrusion detection, which has not been fully explored in prior research. This combination not only provides a more nuanced approach to understanding the data but also significantly enhances the model's ability to detect sophisticated web attacks. This research contributes to the field by presenting a novel framework that leverages advanced NLP and deep learning techniques to build more resilient intrusion detection systems, potentially reducing false positives and improving overall security[22]. The findings from this study are expected to offer valuable insights and practical implications for future research in cybersecurity, particularly in applying NLP and deep learning to enhance network security.

METHOD

This study aims to develop and analyze a network intrusion detection model based on Transformer methods and Natural Language Processing (NLP) techniques to enhance the security of web applications.

Dataset

The CSIC 2010 dataset, developed by the Spanish Research National Council, contains 61,065 records with 17 attributes, including both normal and malicious web traffic such as SQL injection, Cross-Site Scripting (XSS), and Path Traversal attacks. This dataset's diversity is crucial for training models to recognize both attack patterns and normal behaviors in web traffic, ensuring a robust evaluation of the model's ability to handle real-world scenarios[17]. The dataset's size is sufficient for training deep learning models like Transformers, which require large and diverse datasets to capture complex relationships and generalize well without overfitting. NLP techniques are essential for analyzing the textual nature of web-based attacks. Many attacks, such as SQL injection and XSS, exploit text-based inputs within HTTP requests, making them difficult to detect using traditional methods. NLP allows for deeper analysis of textual data, such as URL parameters and HTTP headers, enabling the model to identify subtle anomalies. The Transformer architecture excels at capturing long-range dependencies, making it adaptable to both known and evolving attack patterns, which is vital for detecting emerging threats in web applications.

Algorithm Selection: Transformer Architecture

In this study, we selected the Transformer architecture due to its ability to effectively process sequential data and capture long-range dependencies[23], which are critical for analyzing web application traffic. Traditional machine learning models, such as Random Forest and Support Vector Machines (SVM), often struggle with the dynamic and unstructured nature of web-based attacks, particularly when analyzing text-based HTTP requests that can be manipulated through attacks like SQL injection or Cross-Site Scripting (XSS)[24]. These conventional algorithms rely heavily on predefined features, making them less effective in detecting new and evolving attack patterns.

The Transformer model addresses these limitations through a self-attention mechanism that highlights key parts of an input sequence, like HTTP headers and URL parameters. This feature enables it to capture extensive dependencies and complex relationships within data, enhancing its ability to identify intricate patterns beyond the reach of traditional models[25][26].

Moreover, Transformers offer significant computational advantages over recurrent models like LSTMs and GRUs, especially in large-scale datasets[27]. Their ability to process data in parallel allows for more efficient training on large-scale datasets, such as the CSIC 2010 dataset, without sacrificing accuracy. This makes Transformers not only faster but also more scalable for real-world applications that involve large and diverse data.

In addition, the integration of NLP techniques with the Transformer model enhances its ability to extract meaningful features from web traffic data[28]. Techniques such as Word2Vec, BERT, and TF-IDF enable the model to better understand textual data and context[29], facilitating more accurate detection of web application attacks that exploit text-based inputs.

Network Intrusion Detection

Network Intrusion Detection Systems (NIDS) are crucial for identifying and mitigating security threats to web applications. Traditional NIDS relies on signature-based and anomaly-based methods. Signature-based systems are adequate for known threats but struggle to detect new attacks, while anomaly-based systems can identify unknown attacks but often have high false favorable rates[30]. Advances in machine learning (ML) and deep learning (DL) have enhanced NIDS capabilities, with convolutional neural networks (CNN) and recurrent neural networks (RNN) demonstrating improved accuracy[31]. However, these models often fail to capture network logs' temporal and contextual dependencies, which is essential for detecting sophisticated web application attacks. Transformer models and Natural Language Processing (NLP) techniques have been introduced to address this. Transformers excel in capturing long-term dependencies and contextual relationships in sequential data[18], while NLP enables effective preprocessing and representation of network logs[11]. This study develops a more robust NIDS for detecting web application attacks by combining Transformer models and NLP, aiming to reduce false positives and improve detection accuracy.

ISSN 2089-8673 (Print) | ISSN 2548-4265 (Online)
Volume 13, Issue 3, December 2024

Transformer

The Transformer is an architectural model that has revolutionized the landscape of natural language processing (NLP) and various other applications. As introduced in "Attention is All You Need," the Transformer relies on the self-attention mechanism to capture relationships between elements in sequential data[17]. The self-attention mechanism allows the model to efficiently consider the entire input context without processing the data sequentially, unlike traditional approaches such as RNNs and LSTM[32]. The core formula in self-attention is shown in equation (1):

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{DK}}\right)\text{V} \qquad (1)$$

The Transformer model consists of multiple encoders and decoders, with each encoder layer comprising a self-attention mechanism and a feed-forward neural network[33]. The encoder generates contextual representations of the input, which are then used by the decoder to produce the output. This approach enables the Transformer to capture long-term dependencies and complex relationships within the data[34].

Transformers have demonstrated their superiority in various NLP tasks, including machine translation, text classification, and language modeling, outperforming previous approaches[17]. Their application in network intrusion detection leverages Transformers and NLP techniques to preprocess network logs and detect attack patterns with high efficiency and improved accuracy. This study will implement the Transformer model to enhance the detection capabilities of web application attacks, utilizing the power of self-attention to capture complex relationships in network data.

Natural Language Processing

Natural Language Processing (NLP) is a branch of artificial intelligence that enables computers to understand and generate human-like text. NLP encompasses sentiment analysis, machine translation, and network log analysis applications. Fundamental NLP techniques include tokenization (breaking text into smaller units), stemming, and lemmatization.

Recent advancements in NLP, such as BERT, use Transformer architecture to capture the bidirectional context in text, significantly improving the performance of NLP tasks. These models have been successfully applied in various domains, including cybersecurity, to process and analyze network logs for anomaly detection. This research leverages NLP techniques to process network logs, converting them into vector representations, and employs Transformer models to detect web application attacks with greater accuracy. Recent advancements in NLP, such as BERT, utilize transformer architecture to capture bidirectional context in text, thereby enhancing the performance of NLP tasks. These models have been successfully applied across various domains, including cybersecurity, to process and analyze network logs for anomaly detection. This study leverages NLP techniques to process network logs, converting them into vector representations, and employs transformer models to more accurately detect web application attacks.

integration of Transformer models with NLP

This study proposes the integration of Transformer models with NLP techniques to detect attacks on web applications through a Network NIDS. The proposed model in this research is illustrated in Figure 1.

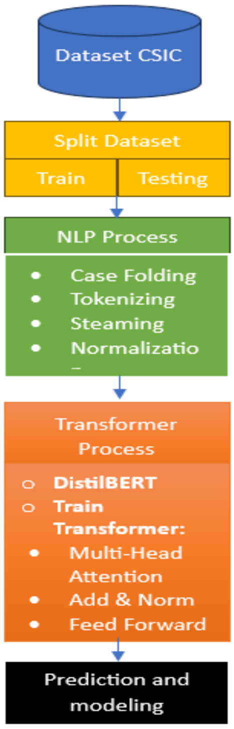

Figure 1. Intrusion Detection Architecture

Based on Figure 1, the steps for intrusion detection are further detailed in Algorithm 1. The initial stage involves initializing parameters for the Transformer and the DistilBERT tokenizer. The NLP preprocessing phase includes case folding, tokenization, stemming, and normalization to ensure that the text data is consistent and formatted

ISSN 2089-8673 (Print) | ISSN 2548-4265 (Online)
Volume 13, Issue 3, December 2024

adequately for analysis. The DistilBERT tokenizer then converts the preprocessed text into appropriate tokens. The following equations are used in the process: Equation (5) for converting logs, including URLs, into lowercase (case folding). Equation (6) for tokenization. Equation (7) for stemming. Equation (8) for normalization

$$lower\ (T) = map(\lambda \mathfrak{x}: \mathfrak{x} \rightarrow lowercase(\mathfrak{x})) \quad (5)$$

$$Tokend = Tokenize(T, delimiter) \quad (6)$$

$$Stem = StemmingAlgoritm(T) \quad (7)$$

$$text \longrightarrow \text{normalized text} \quad (8)$$

Next, the tokenized data is converted into tensors, enabling processing by the Transformer model. The training phase of the Transformer model involves several critical steps, including Multi-Head Attention to capture various aspects of relationships between words, Add & Norm for normalization and residual addition, and Feed Forward layers for further data transformation. After training, the model's performance is evaluated using metrics such as accuracy, recall, F1 score, and AUC to assess its effectiveness in detecting intrusions.

$$output_{residual} = input + Sublayer(input) \quad (9)$$
$$output_{norm} = \frac{output_{residual} - \mu}{\sigma} . \gamma + \beta \quad (10)$$
$$FFN_1(\mathfrak{x}) = ReLU(W_1 x + b_1 \quad (11)$$

Algorithm 1: Transformer NLP Integration

Input: Input: Dataset D={(xi,yi)}
Output: Final model intrusion detection

1. Initialization:
   - Parameters for the Transformer and DistilBERT tokenizer are initialized.
2. NLP Preprocessing:
   - Case Folding
   - Tokenization
   - Steaming
   - Normalization
3. Tokenization
   - The DistilBERT Tokenizer is used to convert text into appropriate tokens:
4. Conversion to Tensors
   - The tokenized data is converted into tensors that the Transformer model can process
5. Train Transformer
   - The Transformer model is trained with the processed data
     a. Multi-Head Attention: use equation (1)
     b. Add & Norm: Normalization and residual addition. Use equations (9) and (10)
     c. Feed Forward. Use equation (11)
6. Model Evaluation
   - The model has evaluated the use of equations (12), (13, (14), (15), (16).
7. Final Model
   - The final model is returned for intrusion detection

Evaluation

The next step in this research is to evaluate the performance of the developed intrusion detection model. The objective of this performance testing is to determine the extent to which the model is suitable for practical use. Several evaluation parameters are utilized, including Accuracy (Ac), Recall (Re), Precision (Pr), F1 Score (F1), and Area Under the Curve (AUC)[21][35]. These parameters provide a comprehensive assessment of the model's effectiveness and reliability in classification. The formulas for each parameter are given in equations (12), (13), (14), (15), and (16)[18]. Table 1 illustrates the prediction of target labels. The next step involves reporting the model's performance using the Receiver Operating Characteristic (ROC) curve to assess the intrusion detection model.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+PN} \quad (12)$$

$$Precision = \frac{TP}{TP+TF} \quad (13)$$

$$Recall = \frac{TP}{TP+TN} \quad (14)$$

$$F1\ Score = \frac{2\ x\ Precision\ x\ recall}{precision+recall} \quad (15)$$
$$AUC = \int_0^1 TPR(FPR)d\ (FPR) \quad (16)$$

Table1. Confusion Matrix

	True Normal	True Anomalous
Predict Normal	TP	FP
Predict Anomalous	TN	TN

Statistical Validation

In this study, statistical validation is performed using the Friedman Test and the

Paired T-test. The Friedman Test, a non-parametric test, is used to compare the performance of multiple classification models on the same dataset[36]. This test examines the null hypothesis that there is no significant difference in the performance of these models. If the Friedman Test results indicate a significant difference, further analysis is conducted using the Paired T-test to identify which pairs of models have significantly different performances[37]. The combination of these two tests provides comprehensive validation, ensuring that the developed model is not only statistically superior but also has practical significance in its application[34].

RESULT AND DISCUSSION

The proposed Transformer-NLP method demonstrates that the Transformer model excels in capturing contextual relationships in network logs, enhancing its ability to detect web application attacks. This success can be attributed to the Transformer's self-attention mechanism, which enables the model to identify intricate attack patterns by focusing on relevant sections of the input data, making it highly effective in distinguishing between normal and anomalous traffic.

Data Processing

At this stage, data cleaning was performed, where three records with missing data were removed, reducing the dataset from 61,065 to 61,062 records. The following process involved reducing the number of variables from 17 to 2, which were relevant to the context of the research. Table 1 shows the dataset after data preprocessing. Table 2 defines the target or label for classification, where 0 represents normal, and 1 represents anomalous.

Table 2. Pre-processing Result Dataset

	URL	Label
1	<s> http :// localhost : 8080 / tienda 1 / publico / vaciar . jsp ? b 2 = vac iar + carr ito http / 1 . 1 </s>	0
2	http://localhost:8080/ ?OpenServer HTTP/1.1	0
61062	http://localhost:8080/tienda1/miembros.Inc HTTP/1.1	1

Text Representation Formation

In this stage, processing is conducted using NLP techniques, including tokenizing, case folding, stemming, and stop word normalization. First, tokenizing: The results of tokenization demonstrate how URLs are broken down into smaller parts that the transformer model can process. This process involves adding unique tokens, handling special characters and symbols, and sub-word tokenization to address words not present in the model's overall vocabulary. Table 3 provides a clear overview of how raw data is transformed into a format suitable for NLP modelling.

Table 3. Tokenization Results

Input Process	Output Process
http://localhost:8080/ tienda1 /publico/vaciar.jsp? B2=Vaciar+carrito HTTP/1.1	<s> http :// localhost : 8080 / tienda 1 / publico / vaciar . jsp ? B 2 **=** Vac iar + carr ito HTTP / 1 . 1 </s>

Next, all characters in the URL are converted to lowercase before tokenization using case folding. Table 4 shows the results of tokenization, demonstrating that all elements in the URL have been converted to lowercase and broken down into smaller tokens. This helps ensure consistency in text processing and makes the model more robust against variations in capitalization.

Table 4. Case Folding Results

Input Process	Output Process
<s> http :// localhost : 8080 / tienda 1 / publico / vaciar . jsp ? B 2 **=** Vac iar + carr ito HTTP / 1 . 1 </s>	<s> http :// localhost : 8080 / tienda 1 / publico / vaciar . jsp ? b 2 = vac iar + carr ito http / 1 . 1 </s>

Table 5. Stemming Results

Input Process	Output Process
<s> http :// localhost : 8080 / tienda 1 / publico / vaciar . jsp ? b 2 = vac iar + carr ito http / 1 . 1 </s>	<s> http :// localhost : 8080 / tienda 1 / publico / vaciar . jsp ? b 2 = vac iar + carr ito http / 1 . 1 </s>

The steaming process does not significantly alter the text in this case because most tokens are part of URLs or symbols. However, words like "vaciar" and "carr" will be processed if there are suffixes that can be removed. Table 5 presents the final results, showing that tokenization and stemming have been applied, although minimal changes occurred due to the specific characteristics of the input text (URLs and symbols).

The stop word removal stage is omitted since most tokens are part of URLs. The normalization process at this stage includes converting all text to lowercase, removing punctuation, and eliminating numbers. Lowercasing ensures consistency, allowing 'HTTP' and 'http' to be treated identically. Punctuation marks, such as periods, slashes, and question marks, are removed to streamline the text. Table 6 presents the results of applying these normalization steps to the sample input.

Table 6. Normalization Results

Input Process	Output Process
<s> http :// localhost : 8080 / tienda 1 / publico / vaciar . jsp ? b 2 = vac iar + carr ito http / 1 . 1 </s>	<s> http :// localhost : 8080 / tienda 1 / publico / vaciar . jsp ? b 2 = vac iar + carr ito http / 1 . 1 </s>

Model Implementation

In this study, the implementation of the Transformer model with the integration of Natural Language Processing (NLP) for network intrusion detection is conducted through several key stages. The first stage is data processing, which includes normalization, batch processing, and splitting the data into training and testing sets with ratios of 70/30, 80/20, and 90/10. In the NLP processing, steps such as case folding, text normalization, tokenization, and stemming are performed to ensure the text is in a consistent format. Tokenization uses the DistilBERT Tokenizer to convert the text into tokens that the Transformer model can process.

As shown in Figure 1, the architecture for network intrusion detection with Transformer and NLP integration is implemented according to Algorithm 1. In the model training stage, DistilBERT, initialized with default parameters, is used to handle the Multi-Head Attention, Add & Norm, and Feed Forward layers. The model is trained using the Adam optimizer with a learning rate of 2e-5 and the Cross-Entropy loss function. Training is conducted over three epochs with a batch size of 8. Model evaluation is performed by measuring metrics such as accuracy, recall, F1 score, and ROC-AUC to ensure the model's performance in detecting network intrusion categories classified as "Normal" and "Anomalous." Evaluation results indicate that the integration of the Transformer model and NLP is effective in detecting web application attacks and significantly contributes to the improvement of the network intrusion detection system's accuracy. The parameters of the Transformer model integrated with NLP are shown in Table 7.

Table 7. Parameter Model

Parameter	Value
Input Shape	Input_dim
NLP Pre-preprocessing	Case Folding, Normalization, Tokenization, Stemming
Tokenization	DistilBERTTokenizer
Multi-Head Attention	Num_heads=8, dim_model=512
Add & Norm	Layer Normalization
Feed Forward	Dense (2048, Activation='ReLU'
Linear Layer	Dense (256, activation='softmax'
Softmax Layer	Dense(num_classes, activation='softmax'
Optimizer	AdamW (learning_rate=2e-5)
Loss Function	Cross-Entropy Loss
Training Parameter	Epoch=3, Batch Size=8
Evaluation Matrix	Accuracy, Recall, F1 Score, AUC

Evaluation

The implemented model is then evaluated to test its performance. Compared to traditional algorithms such as DNN, Random Forest, and SVM, the Transformer-NLP model showed marked improvements in accuracy and AUC. Previous studies using conventional methods often struggled to maintain high detection rates across varied datasets, while the Transformer model's adaptive architecture proved effective in handling diverse attack types, as evidenced by its consistently higher AUC scores across multiple data splits. The evaluation uses equations (12), (14), (15), and (16). The results are shown in Tables 8, 9, and 10. Subsequently, the model's performance is tested using the ROC curves, which are displayed in Figures 2, 3, and 4.

Table 8. Evaluation Using 80-20 Training Split

Algorithm	A_c	R_e	F_1	AUC
DNN	0.76	0.78	0.74	0.83
RF	0.83	0.98	0.82	0.92
DT	0.82	0.93	0.80	0.88
SVM	0.80	0.89	0.72	0.82
KNN	0.81	0.94	0.80	0.90
XGBoost	0.83	0.96	0.82	0.93
NB	0.63	0.33	0.42	0.59
Trans+NLP	0.85	0.95	0.83	0.94

Table 9. Evaluation Using 70-30 Training Split

Algorithm	A_c	R_e	F_1	AUC
DNN	0.79	0.78	0.74	0.83
RF	0.83	0.98	0.82	0.88
DT	0.82	0.93	0.80	0.88
SVM	0.73	0.89	0.72	0.82
KNN	0.81	0.94	0.72	0.90
XGBoost	0.83	0.96	0.82	0.93
NB	0.64	0.33	0.42	0.59
Trans+NLP	0.85	0.95	0.83	0.94

Table 10. Evaluation Using 90-10 Training Split

Algorithm	A_c	R_e	F_1	AUC
DNN	0.77	0.52	0.65	0.85
RF	0.83	0.99	0.83	0.93
DT	0.83	0.94	0.82	0.90
SVM	0.72	0.86	0.72	0.84
KNN	0.80	0.87	0.78	0.89
XGBoost	0.83	0.94	0.82	0.92
NB	0.63	0.30	0.40	0.85
Trans+NLP	0.85	0.95	0.84	0.94

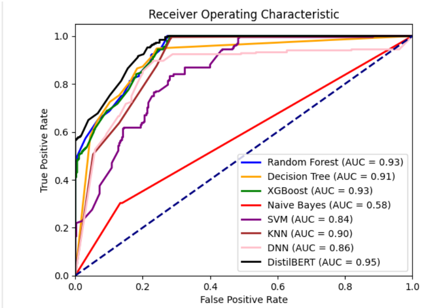

Figure 2. ROC for 90-10 Model

Statical Validation

To test the reliability of the built model, we conducted evaluations using the Friedman test and t-test to compare its performance with other models[36]. We designated the proposed model as the control method in this experiment, and the significance level α for the statistical tests was set at 0.05. Generally, a smaller p-value indicates a significant difference between comparison methods. The results of the Friedman test and t-test are shown in Table 11.

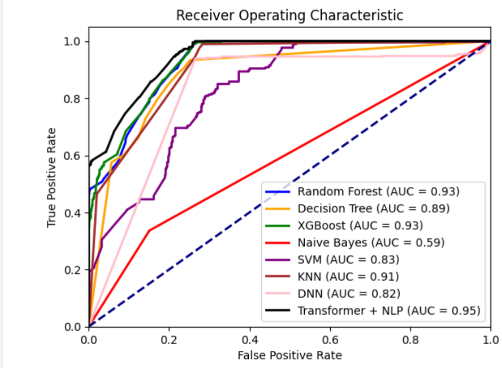

Figure 3. ROC for 80-20 Model

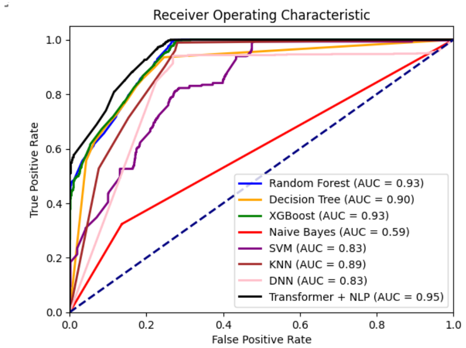

Figure 4. ROC for the 70-30 Model

Parameter Sensivitas

In this section, we examine the impact of the hyperparameter, denoted by λ, on the proposed detection model. This analysis aims to understand how variations in λ influence the model's performance and effectiveness. The study involves adjusting the λ values and observing changes in key performance metrics, such as accuracy, recall, F1 score, and ROC-AUC. The results of this hyperparameter tuning are presented in Table 12, illustrating the relationship between different λ values and the corresponding performance metrics. This detailed evaluation helps in identifying the optimal λ setting for achieving the best detection results.

Table 11. Friedman Test and T-test Results

	DNN	DT	XB	NB	SVM	KNN	RF
Friedman	0.0009	0.005	0.019	2.159	0.0001	0.006	0.04
T-Test	8.705	5.35	3.765	40.785	13.19	5.244	2.99

.Table 12. Impact of Hyperparameter λ on Model Performance

Λ	A_c	R_c	F_1	Auc
1e-05	0.856	0.944	0.843	0.948
2e-05	0.852	0.944	0.840	0.950
3e-05	0.851	0.906	0.841	0.946
5e-05	0.849	0.952	0.838	0.946

ISSN 2089-8673 (Print) | ISSN 2548-4265 (Online)
Volume 13, Issue 3, December 2024

Based on Table 12, although the AUC value remains the same (0.95) for several learning rate values, other metrics such as Accuracy, Recall, and F1 Score vary. The model with a learning rate of 2e-05 shows the highest AUC of 0.9505, indicating slightly better performance compared to other learning rates. Models with learning rates of 1e-05 and 3e-05 exhibit nearly the same AUC values (around 0.9490) but with variations in Accuracy and Recall. The ROC curve, illustrating sensitivity, is shown in Figure 5.

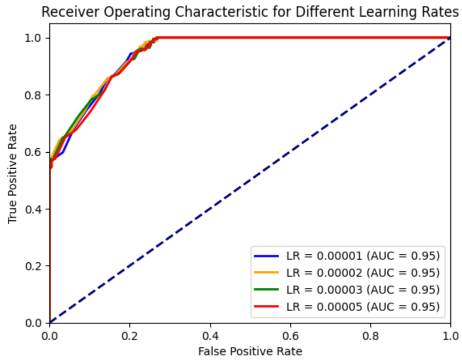

Figure 5. ROC Curve for Sensitivity Analysis of Parameters

Discussion

This study demonstrates that integrating the Transformer model with NLP techniques significantly enhances the performance of NIDS for web applications. The use of the Transformer model, with its self-attention mechanism, allows for capturing complex dependencies in sequential data, such as HTTP requests, which is crucial for detecting intricate attack patterns within dynamic and diverse web traffic. The CSIC 2010 dataset used in this study was processed through several pre-processing steps, including tokenization, stemming, lemmatization, and normalization, to ensure data consistency. Text representation techniques such as Word2Vec, BERT, and TF-IDF were employed to enable the Transformer model to effectively capture contextual relationships in network log data.

The model's performance evaluation demonstrated superior results compared to traditional algorithms like Deep Neural Network (DNN), Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM), k-nearest Neighbor (KNN), XGBoost, and Naive Bayes (NB). The Transformer-NLP model achieved higher accuracy (up to 85%), recall (95%), F1 score (83%), and AUC (0.95) across training/testing splits of 80/20, 70/30, and 90/10. This performance is especially significant when compared to traditional models, which showed lower AUC values, indicating that the Transformer-NLP approach provides a more robust framework for intrusion detection across various scenarios, with the best AUC value of 0.9505 at a learning rate of 2e-05, demonstrating its ability to adapt to different training scenarios. The ROC curve further illustrated the model's superior capability in distinguishing between normal and anomalous traffic, proving more reliable than the other models tested.

Statistical validation using the Friedman test and t-test confirmed the reliability and practical significance of the proposed model. Hyperparameter sensitivity analysis indicated that variations in the λ value impacted the model's performance, with a learning rate of 2e-05 providing the optimal results. These findings suggest that the proposed Transformer-NLP model is not only effective in improving detection accuracy but also offers a robust framework for reducing false positives, enhancing the overall security posture of web applications in response to increasingly sophisticated cyber threats.

Additionally, the model effectively detects complex attack patterns, especially in text-based inputs like SQL injection and XSS, enhancing web application security, and preventing unauthorized access and malicious data manipulation. The Transformer-NLP model’s unique integration of NLP for preprocessing and the self-attention mechanism significantly reduces false positive rates. This reduction enhances both efficiency and reliability in real-world scenarios, as it minimizes unnecessary alerts and focuses security resources on genuine threats. By improving precision and recall, this model presents a more reliable solution for continuous, real-time web application monitoring, minimizing unnecessary alerts and enabling security teams to focus on genuine threats. This improvement in detection accuracy directly bolsters the resilience of web applications against evolving attack methods, helping to maintain data integrity, confidentiality, and availability.

However, this study has certain limitations. First, the CSIC 2010 dataset, while useful for evaluating web application security, may not fully capture the range of modern web application attack techniques, potentially limiting the model's applicability to newer or more varied threats. Second, the computational demands of both Transformer models and NLP preprocessing may pose challenges for practical deployment, particularly in environments with constrained resources. Additionally, while this study focused on optimizing performance metrics such as accuracy and AUC, it did not extensively address potential overfitting, which can be a concern with complex models trained on relatively limited datasets. Future research should explore the use of larger, more diverse datasets and further refine the model to balance computational efficiency with detection capability.

CONCLUSION

This study demonstrates that integrating the Transformer model with NLP techniques significantly improves NIDS performance for web applications by capturing complex contextual relationships in network log data. The Transformer-NLP model outperformed traditional algorithms, including DNN, RF, DT, SVM, KNN, XGBoost, and NB, across key metrics (accuracy, recall, F1 score, and AUC), addressing a crucial gap in current NIDS methods. Statistical validation using the Friedman and t-tests further supports the model's robustness and practical effectiveness, especially in handling the dynamic nature of web traffic.

However, limitations remain. The CSIC 2010 dataset may not fully reflect modern web application threats, which could affect generalizability. Additionally, the model's high computational demands pose challenges for real-world deployment. This study also did not deeply explore overfitting, which could impact performance given the dataset size. Future work should examine strategies such as regularization and cross-validation to enhance model robustness, along with architectural optimizations to improve computational efficiency for practical deployment in constrained environments.

REFERENCES

[1] A. A. Bouramdane, "Cyberattacks in Smart Grids: Challenges and Solving the Multi-Criteria Decision-Making for Cybersecurity Options, Including Ones That Incorporate Artificial Intelligence, Using an Analytical Hierarchy Process," *J. Cybersecurity Priv.*, vol. 3, no. 4, pp. 662–705, 2023, doi: 10.3390/jcp3040031.

[2] J. A. Dharma and Rino, "Network Attack Detection Using Intrusion Detection System Utilizing Snort Based on Telegram," *bit-Tech*, vol. 6, no. 2, pp. 118–126, 2023, doi: 10.32877/bt.v6i2.943.

[3] O. J. Falana, I. O. Ebo, C. O. Tinubu, O. A. Adejimi, and A. Ntuk, "Detection of Cross-Site Scripting Attacks using Dynamic Analysis and Fuzzy Inference System," *2020 Int. Conf. Math. Comput. Eng. Comput. Sci. ICMCECS 2020*, 2020, doi: 10.1109/ICMCECS47690.2020.240871.

[4] P. Dini, A. Elhanashi, A. Begni, S. Saponara, Q. Zheng, and K. Gasmi, "Overview on Intrusion Detection Systems Design Exploiting Machine Learning for Networking Cybersecurity," *Appl. Sci.*, vol. 13, no. 13, 2023, doi: 10.3390/app13137507.

[5] N. Agarwal and S. Z. Hussain, "A Closer Look at Intrusion Detection System for Web Applications," *Secur. Commun. Networks*, vol. 2018, 2018, doi: 10.1155/2018/9601357.

[6] Y. J. Park and J. C. Park, "Web Application Intrusion Detection System for input validation attack," *Proc. - 3rd Int. Conf. Converg. Hybrid Inf. Technol. ICCIT 2008*, vol. 2, pp. 498–504, 2008, doi: 10.1109/ICCIT.2008.338.

[7] S. Sasipriya, L. R. Madhan Kumar, R. Raghuram Krishnan, and K. Naveen Kumar, "Intrusion Detection System in Web Applications (IDSWA)," *Proc. - 5th Int. Conf. Intell. Comput. Control Syst. ICICCS 2021*, no. Iciccs, pp. 311–314, 2021, doi: 10.1109/ICICCS51141.2021.9432086.

[8] M. Verkerken, L. D'hooge, T. Wauters, B. Volckaert, and F. De Turck, "Towards Model Generalization for Intrusion Detection: Unsupervised Machine Learning Techniques," *J. Netw. Syst. Manag.*, vol. 30, no. 1, pp. 1–25, 2022, doi: 10.1007/s10922-021-09615-7.

[9] L. Ashiku and C. Dagli, "Network Intrusion Detection System using Deep Learning," *Procedia Comput. Sci.*, vol. 185, no. June, pp. 239–247, 2021, doi: 10.1016/j.procs.2021.05.025.

[10] R. Sudiyarno, A. Setyanto, and E. T. Luthfi, "Peningkatan Performa Pendeteksian Anomali Menggunakan Ensemble Learning dan Feature Selection Anomaly Detection Performance Improvement Using Ensemble Learning and Feature Selection," *Citec J.*, vol. 7, no. 1, pp. 1–9, 2020.

[11] S. Das, M. Ashrafuzzaman, F. T. Sheldon, and S. Shiva, "Network Intrusion Detection using Natural Language Processing and Ensemble Machine Learning," *2020 IEEE Symp. Ser. Comput. Intell. SSCI 2020*, no. Ml, pp. 829–835, 2020, doi: 10.1109/SSCI47803.2020.9308268.

[12] R. Sujatha, A. Teja, P. Naveen, and J. M. Chatterjee, "Web Application for Traffic

Monitoring and Guidance," vol. 10, no. 4, pp. 1–14, 2020, doi: 10.33168/JSMS.2020.0403.
[13] J. R. Tadhani, V. Vekariya, V. Sorathiya, S. Alshathri, and W. El Shafai, "Securing web applications against XSS and SQLi attacks using a novel deep learning approach," *Sci. Rep.*, pp. 1–17, 2024, doi: 10.1038/s41598-023-48845-4.
[14] T. Sowmya and M. A. E. A, "Measurement : Sensors A comprehensive review of AI based intrusion detection system," *Meas. Sensors*, vol. 28, no. May, p. 100827, 2023, doi: 10.1016/j.measen.2023.100827.
[15] J. Campino, "Unleashing the transformers : NLP models detect AI writing in education," *J. Comput. Educ.*, no. 0123456789, 2024, doi: 10.1007/s40692-024-00325-y.
[16] N. Patwardhan, S. Marrone, and C. Sansone, "Transformers in the Real World : A Survey on NLP Applications," 2023.
[17] Z. Long, H. Yan, G. Shen, X. Zhang, H. He, and L. Cheng, "A Transformer-based network intrusion detection approach for cloud security," *J. Cloud Comput.*, vol. 13, no. 1, 2024, doi: 10.1186/s13677-023-00574-9.
[18] Y. Liu and L. Wu, "Intrusion Detection Model Based on Improved Transformer," *Appl. Sci.*, vol. 13, no. 10, 2023, doi: 10.3390/app13106251.
[19] J. Kim, H. Kang, and P. Kang, "Time-series anomaly detection with stacked Transformer representations and 1D convolutional network," *Eng. Appl. Artif. Intell.*, vol. 120, no. November 2022, p. 105964, 2023, doi: 10.1016/j.engappai.2023.105964.
[20] N. Montes, G. Betarte, R. Martínez, and A. Pardo, "Web Application Attacks Detection Using Deep Learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12702 LNCS, pp. 227–236, 2021, doi: 10.1007/978-3-030-93420-0_22.
[21] A. D. Y. SURYADI, "Pengembangan Intrusion Detection System (IDS) Berbasis Machine Learning," vol. 13, no. 3, pp. 189–195, 2022, [Online]. Available: https://repository.mercubuana.ac.id/63488/.
[22] A. Nurdin, B. Anggo Seno Aji, A. Bustamin, and Z. Abidin, "Perbandingan Kinerja Word Embedding Word2Vec, Glove, Dan Fasttext Pada Klasifikasi Teks," *J. Tekno Kompak*, vol. 14, no. 2, p. 74, 2020, doi: 10.33365/jtk.v14i2.732.
[23] S. R. Choi and M. Lee, "Transformer Architecture and Attention Mechanisms in Genome Data Analysis: A Comprehensive Review," *Biology (Basel).*, vol. 12, no. 7, 2023, doi: 10.3390/biology12071033.
[24] H. Salih Abdullah and A. Mohsin Abdulazeez, "Detection of SQL Injection Attacks Based on Supervised Machine Learning Algorithms: A Review," *Int. J. Informatics, Inf. Syst. Comput. Eng.*, vol. 5, no. 2, pp. 152–165, 2024, doi: 10.34010/injiiscom.v5i2.12731.
[25] H. Wang and W. Li, "DDosTC: A transformer-based network attack detection hybrid mechanism in SDN," *Sensors*, vol. 21, no. 15, 2021, doi: 10.3390/s21155047.
[26] Z. Gao, Y. Shi, and S. Li, "Self-attention and long-range relationship capture network for underwater object detection," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 36, no. 2, p. 101971, 2024, doi: 10.1016/j.jksuci.2024.101971.
[27] H. Kheddar, "Transformers and Large Language Models for Efficient Intrusion Detection Systems : A Comprehensive Survey," pp. 1–34.
[28] H. Zhang and M. O. Shafiq, "Survey of transformers and towards ensemble learning using transformers for natural language processing," *J. Big Data*, 2024, doi: 10.1186/s40537-023-00842-0.
[29] D. E. Cahyani and I. Patasik, "Performance comparison of TF-IDF and Word2Vec models for emotion text classification," vol. 10, no. 5, pp. 2780–2788, 2021, doi: 10.11591/eei.v10i5.3157.
[30] G. Zachos, I. Essop, G. Mantas, K. Porfyrakis, J. C. Ribeiro, and J. Rodriguez, "An Anomaly-Based Intrusion Detection System for Internet of Medical Things Networks," *Electronics*, no. June, pp. 1–25, 2021.
[31] A. Aldallal, "Toward Efficient Intrusion Detection System Using Hybrid Deep Learning Approach," *Sysmmetry*, 2022.
[32] A. Chandra, L. Tünnermann, T. Löfstedt, and R. Gratz, "Transformer-based deep learning for predicting protein properties in the life sciences," *Elife*, vol. 12, pp. 1–25, 2023, doi: 10.7554/eLife.82819.
[33] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI Open*, vol. 3, no. October, pp. 111–132, 2022, doi:

10.1016/j.aiopen.2022.10.001.

[34] R. Cao, J. Wang, M. Mao, G. Liu, and C. Jiang, "Feature-wise attention based boosting ensemble method for fraud detection," *Eng. Appl. Artif. Intell.*, vol. 126, no. PC, p. 106975, 2023, doi: 10.1016/j.engappai.2023.106975.

[35] T. S. Lestari, I. Ismaniah, and W. Priatna, "Particle Swarm Optimization for Optimizing Public Service Satisfaction Level Classification," *J. Nas. Pendidik. Tek. Inform.*, vol. 13, no. 1, pp. 147–155, 2024, doi: 10.23887/janapati.v13i1.69612.

[36] J. Liu and Y. Xu, "T-Friedman Test: A New Statistical Test for Multiple Comparison with an Adjustable Conservativeness Measure," *Int. J. Comput. Intell. Syst.*, vol. 15, no. 1, pp. 1–19, 2022, doi: 10.1007/s44196-022-00083-8.

[37] W. Priatna, H. Dwi Purnomo, A. Iriani, I. Sembiring, and T. Wellem, "Optimizing Multilayer Perceptron with Cost-Sensitive Learning for Addressing Class Imbalance in Credit Card Fraud Detection," *Resti*, vol. 8, no. 4, pp. 19–25, 2024.