Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

Optimizing Early Network Intrusion Detection: A Comparison of LSTM and LinearSVC with SMOTE on Imbalanced Data

Khabib Adi Nugroho*[1], Taqwa Hariguna[2], Azhari Shouni Barkah[3]

[1,2]Magister of Computer Science, Faculty of Computer Science, Universitas Amikom Purwokerto, Indonesia
[3]Informatics, Faculty of Computer Science, Universitas Amikom Purwokerto, Indonesia

Email: [1]23ma41d018@students.amikompurwokerto.ac.id

Received : Apr 28, 2025; Revised : Jun 24, 2025; Accepted : Jun 24, 2025; Published : Dec 22, 2025

Abstract

This study aims to improve network intrusion detection systems (IDS) by addressing class imbalance in the CICIDS 2017 dataset. It compares the effectiveness of Long Short-Term Memory (LSTM) networks and Linear Support Vector Classifier (LinearSVC) in detecting intrusions, with a focus on the impact of Synthetic Minority Over-sampling Technique (SMOTE) for balancing the dataset. The dataset was preprocessed by removing irrelevant features, handling missing values, and applying Min-Max normalization. SMOTE was applied to balance the training dataset. Results showed that LSTM outperformed LinearSVC, especially in recall and F1-score, after applying SMOTE. This research highlights the benefits of combining LSTM with SMOTE to address class imbalance in IDS and emphasizes the importance of temporal sequence models like LSTM for detecting network intrusions. Future work could involve using the full dataset, exploring advanced feature engineering, and implementing more complex architectures to further enhance performance. This research underscores the critical need for improving network security by addressing the challenges of class imbalance in intrusion detection systems, which is vital for ensuring the real-time identification and mitigation of sophisticated cyber threats in the ever-evolving landscape of network security.

Keywords : *Intrusion Detection System, LinearSVC, Network Traffic, S, SMOTE*

1. INTRODUCTION

The rapid advancement of Information and Communication Technology (ICT) has become the foundation of modern society, transforming nearly every sector of life, including business, government, and social services. ICT infrastructure has become the backbone that supports daily activities, from communication and digital transactions to secure processing of sensitive data. As economies and societies increasingly transition towards digitalization, dependence on network infrastructure is rising. From e-governance systems to digital commerce and beyond, this infrastructure is critical for enabling and streamlining operations across various sectors [1], [2]. For example, the FinTech sector has seen a surge, with mobile payment services and digital wallets providing new business opportunities, particularly within the digital finance industry [3], [4]. With the increasing integration of digital platforms into every aspect of our lives, securing network infrastructure has become a top priority. As reliance on digital services grows, the frequency and sophistication of cyber threats such as hacking, malware, and Distributed Denial of Service (DDoS) attacks are rising. These threats target not only private entities but also governments, industries, and other critical sectors, with potentially severe impacts. The consequences of these attacks can be serious, ranging from financial losses and reputational damage to the leakage of sensitive data [2]. Malware, for example, continues to be a serious threat, despite advances in cybersecurity technology. It disrupts business operations and causes substantial

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

financial losses [5]. Similarly, DDoS attacks increasingly target network infrastructure, flooding resources and rendering services inaccessible, making detection and response crucial to mitigating this risk [6].

Early intrusion detection is critical to preventing attacks from escalating and minimizing the potential damages caused by cyber threats. IDS serve as the primary line of defense in identifying malicious activity within network systems. The effectiveness of an IDS lies in its ability to detect intrusions early, allowing organizations to take timely actions before significant damage is done to sensitive data and critical infrastructure [7]. Research shows that integrating machine learning and Artificial Intelligence (AI) into IDS can enhance their accuracy and ability to detect new and emerging threats [8]. Neural network technologies, for instance, have proven effective in distinguishing between normal network behavior and malicious activity, allowing systems to adapt to new attack vectors [7]. An efficient IDS framework not only prevents major damage but also strengthens the security posture of an organization by ensuring that detected threats can be promptly addressed. By continuously monitoring network behavior in real-time, these systems provide early warnings that help organizations implement corrective actions in a timely manner [9]. Additionally, building a strong early detection system involves more than just anomaly detection; it requires the implementation of advanced monitoring techniques to assess patterns that signal potential breaches. These systems are essential for maintaining the integrity of networks by reducing the likelihood of ongoing or widespread damage [10], [11]. Early detection methodologies, including clustering algorithms and anomaly detection, play a significant role in providing a comprehensive understanding of network behavior and its vulnerabilities to threats [12]. Therefore, developing strategies that prioritize early detection and effective threat management is crucial for safeguarding sensitive information and ensuring operational continuity.

As digital platforms continue to proliferate, ensuring the security of network systems has become paramount. With growing dependence on digital infrastructure and the evolving complexity of cyber threats, robust network security measures, including effective early detection systems, are essential. Detecting intrusions early allows organizations to reduce threats before they escalate into significant damage, thereby maintaining data integrity and operational continuity. The evolution of IDS technology, particularly through machine learning and AI, has proven to be a powerful tool in enhancing security systems' ability to adapt to ever-changing threats, emphasizing the importance of proactive defense strategies to safeguard the digital world [13]. Understanding common types of network attacks is crucial for developing effective cybersecurity strategies, as these attacks can have a significant impact on data integrity and the functionality of network infrastructure as a whole. Some of the most frequent attacks include malware, phishing, Man-in-the-Middle (MITM) attacks, Denial of Service (DoS), and intrusion attempts, each with its own set of threats and consequences [14]. Malware refers to malicious software designed to disrupt, damage, or gain unauthorized access to computer systems. This includes viruses, worms, Trojans, and ransomware. Malware attacks can lead to unauthorized access to data, data corruption, and even substantial financial losses for organizations. For instance, ransomware can block access to important data until a ransom is paid, severely disrupting business operations [15], [16]. Phishing is an attack involving deceptive communications, such as emails that appear to come from legitimate organizations, with the aim of stealing sensitive information like passwords or personal data. This type of attack often leads to identity theft and unauthorized access to sensitive assets [15]. MITM attacks occur when an attacker intercepts and forwards messages between two parties without their knowledge. These attacks exploit vulnerabilities in unsecured networks, such as public Wi-Fi, leading to the theft of sensitive data, including authentication credentials [17], [18]. DoS attacks aim to make network services unavailable by overwhelming them with excessive traffic, making the service inaccessible to legitimate users. Distributed Denial of Service (DDoS) attacks are even more severe, involving multiple compromised systems targeting the same victim, amplifying the attack and

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

complicating mitigation efforts [19], [20]. The impact of these attacks can cause significant business disruption and damage an organization's reputation. Intrusion attempts involve unauthorized efforts to access or manipulate network resources. Attackers use various techniques, such as exploiting vulnerabilities in network software or hardware. Successful intrusions can result in data theft or destruction of critical systems [21], [22].

The impact of cyber-attacks extends beyond the immediate targets, affecting stakeholders and customers as well [23]. Financial consequences can include recovery costs, regulatory fines, and loss of business due to reputational damage. Additionally, successful attacks can lead to data breaches, exposing sensitive personal information and amplifying existing risks. Damage to network infrastructure may result in extended service outages and erode customer trust, making it increasingly difficult to maintain smooth operations [24]. One of the main challenges in developing effective IDS is the issue of class imbalance in training data. Real-world network traffic datasets typically contain a significantly higher proportion of normal traffic compared to malicious activity, which causes bias in detection systems [25]. This imbalance results in poor performance, particularly when it comes to detecting rare yet critical attacks. Traditional IDS methods often fail to address this issue effectively, often missing less frequent but highly impactful intrusions. To overcome this challenge, this study investigates the use of LSTM networks combined with SMOTE for handling class imbalance. LSTM, known for its ability to analyze sequential data, is highly effective at capturing long-term dependencies and identifying subtle anomalies in network traffic patterns. SMOTE, on the other hand, generates synthetic samples to balance the dataset, thereby enhancing the model's capacity to detect intrusions across both normal and attack traffic. A comparison is also made with LinearSVC, a commonly used machine learning model. The primary objective of this research is to improve IDS performance through the use of LSTM and SMOTE, achieving more accurate and timely detection of network intrusions. By addressing the issue of class imbalance, the study aims to reinforce network security measures and provide more robust defenses against the growing sophistication of cyber threats.

One of the most innovative aspects of this study is the way in which it addresses the problem of class imbalance in network intrusion detection systems by combining LSTM networks and SMOTE. While earlier research has investigated the usage of LSTM and SMOTE in isolation, the current investigation is the first of its kind to combine the two methodologies in order to improve the identification of uncommon attack patterns in network traffic. LSTM's capacity to capture temporal dependencies and SMOTE's capacity to balance the dataset are leveraged in this research to give a unique method for boosting the performance of IDS. This technique is particularly useful for recognising attacks that are generally under-represented in datasets that are imbalanced. This particular mix of approaches provides a novel approach to addressing the issues that are provided by cyber attacks that are always growing and dynamic. As a result, it constitutes a significant contribution to the field of network security.

2. METHOD

Figure 1 presents the flowchart of the research methodology, outlining the key steps from data collection, preprocessing, and handling class imbalance with SMOTE, to model training and evaluation using LSTM and LinearSVC, ultimately aiming to improve network intrusion detection performance.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

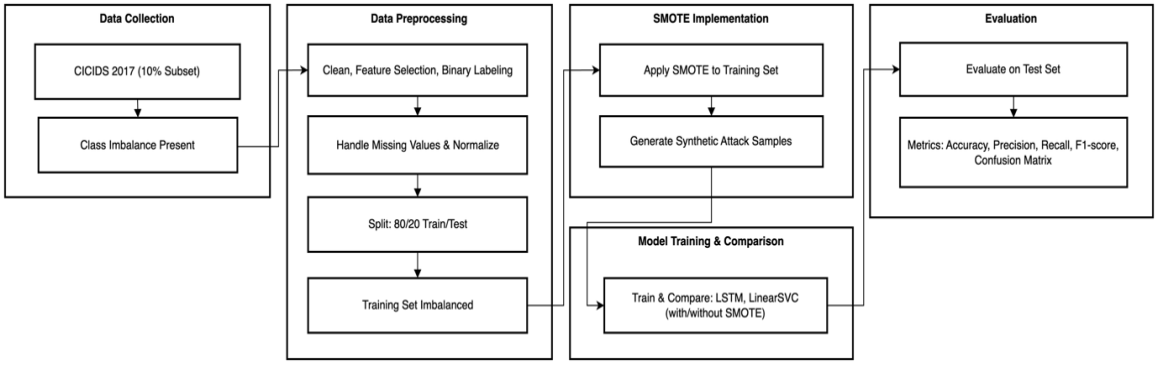


Figure 1. Research Method Flowchart

2.1. Data Collection

In this study, the focus was on utilizing the CICIDS 2017 dataset for the development and evaluation of a LSTM-based IDS. The CICIDS 2017 dataset, widely recognized in network security research, is publicly available and was collected by the Canadian Institute for Cybersecurity (CIC). It contains diverse network traffic data, which includes both normal traffic and a wide variety of cyber-attacks, such as DoS, DDoS, Port Scanning, Botnet activity, and more. The dataset also includes traffic data from several protocols like HTTP, FTP, DNS, and ICMP, making it highly representative of real-world network environments. The data was collected through controlled experiments and network simulations to mimic typical attack scenarios and everyday traffic in enterprise-level networks.

For this study, a 10% subset of the dataset, referred to as dataset_10_percent.csv, was specifically selected for training and evaluation purposes. This sample, containing approximately 252,000 records, was chosen to reduce processing time while still retaining the diversity and integrity of the original dataset. While the CICIDS 2017 dataset contains more than 2 million records, the subset allows for quicker experimentation, making it feasible to test different models and techniques within the scope of computational limitations. Despite using only a fraction of the dataset, the 10% subset remains diverse enough to pose a comprehensive challenge for the intrusion detection system, as it includes both normal traffic and several types of attacks, allowing the models to effectively differentiate between legitimate and malicious activities.

The CICIDS 2017 dataset's class imbalance is problematic. Real-world datasets sometimes have skewed class distributions, with regular traffic instances outnumbering assault instances. Due to its bias towards regular traffic, the model struggles to learn and classify assault patterns. SMOTE is used to balance class distribution and improve model performance, especially in detecting rare and complicated attacks. CICIDS 2017 is appropriate for training machine learning models because it captures the complexity of network traffic and the issues of class imbalance, representing the real-world scenario where most network traffic is benign and only a small percentage is malicious.

2.2. Data Preprocessing

In this study, data preprocessing was a crucial step in preparing the dataset for training the LSTM-based IDS. The dataset used for this study was the CICIDS 2017 dataset, a publicly available and widely recognized dataset in network security research, which contains a variety of network traffic, including both normal traffic and several types of cyber-attacks such as DoS, DDoS, Port Scanning, and Botnet activity, among others. Given the large size and computational requirements of the CICIDS 2017 dataset, a 10% subset of the data, approximately 252,000 records, was used for this study to balance processing efficiency with dataset diversity. This subset retained the integrity of the original dataset's distribution of both normal and malicious network activities.

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

The first step in data preprocessing involved loading the dataset into a Pandas DataFrame, a widely used data structure in Python that is efficient for data manipulation and exploration [26], [27]. After loading, the dataset underwent several cleaning steps, starting with removing leading or trailing whitespace from column names to ensure consistency. 52 relevant features were selected for the intrusion detection model based on prior research and domain expertise, with the aim of focusing on features that could help distinguish between normal and attack traffic. The target variable, labeled "Attack Type", was converted into a binary classification system, where normal traffic was labeled 0, and all attack types were labeled 1. This conversion simplified the model's learning process by allowing it to focus on identifying attack traffic in the dataset.

One of the main challenges in the preprocessing phase was dealing with missing or invalid values. The feature columns were converted to numeric values using Pandas' to_numeric function, which automatically coerced any non-numeric data into NaN values. Rows where the target variable "Attack Type" contained NaN values were removed to ensure the integrity of the target variable. Additionally, infinite values (such as those resulting from division by zero in certain features) were replaced with NaN, and remaining missing values were imputed using the mean of each respective feature. This step ensured that the model retained as much data as possible while dealing with missing or invalid values [28].

After addressing missing values, the dataset underwent feature scaling. Given that network traffic data often contains features with vastly different ranges—such as session duration measured in seconds and the number of packets transferred in the thousands—it was essential to apply Min-Max normalization. This method scaled all features to a consistent range between 0 and 1, ensuring that no single feature would dominate the learning process due to its larger scale. Normalization is particularly important for machine learning models like LSTM, which are sensitive to the magnitude of input features. By normalizing the data, all features were treated equally, allowing the model to learn more effectively from each feature based on its relative importance [29]. The formula for Min-Max Scaling is showed in (1) as follow:

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \quad (1)$$

where X is the original value of the feature, X_{min} is the minimum value of the feature, X_{max} is the maximum value of the feature, and X_{norm} is the normalized value of the feature.

Once the data was preprocessed and normalized, it was split into training and testing sets using an 80/20 ratio, with 80% of the data used for training the model and 20% reserved for testing. This division ensured that the model had enough data to learn from while also being evaluated on unseen data. The stratified sampling technique was used for this split, which ensured that both the training and testing sets maintained the same class distribution as the original dataset. This step is particularly crucial for imbalanced datasets, as it ensures that both normal and attack traffic are proportionally represented in both sets, giving the model a fair chance to learn from both classes. However, the class distribution in the training set was heavily imbalanced, with normal traffic vastly outnumbering attack traffic. This imbalance presented a significant challenge for the model, as it could lead the model to favor predicting normal traffic, potentially overlooking the less frequent but critical attack traffic.

2.3. SMOTE Implementation

Class imbalance poses a critical challenge in network intrusion detection, as the overwhelming prevalence of normal traffic compared to attack instances can bias the classifier toward predicting the majority class, leading to high accuracy for normal traffic but poor detection of rare attacks. This issue results in higher false negatives and missed attack detections. To address this, the SMOTE is applied,

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

generating synthetic examples for the minority (attack) class, which helps balance the dataset. SMOTE works by interpolating new minority samples between existing ones, enriching the feature space and mitigating overfitting. Studies have demonstrated SMOTE's effectiveness in improving intrusion detection system performance. Research [30] showed that combining SMOTE with other techniques enhances the detection of rare attack patterns, while [31] applied SMOTE and Borderline-SMOTE in industrial control systems, improving detection and reducing false negatives. Further research by [32], [33] affirmed that SMOTE leads to better model performance by rebalancing training data, allowing the model to learn more discriminative features. Additionally, SMOTE has been shown to enhance the learning process of various machine learning architectures. Research [34] successfully integrated SMOTE into a hybrid CNN-BiGRU model, significantly improving intrusion detection in imbalanced datasets. These studies collectively highlight SMOTE's crucial role in improving intrusion detection system performance by addressing class imbalance and enhancing detection accuracy for rare attack types.

SMOTE generates synthetic data by choosing random data points from the minority class and finding their nearest neighbors. A synthetic data point X_{new} is created using the formula (2) as follow

$$X_{\text{new}} = X_{\text{minority}} + \lambda \cdot \left(X_{\text{neighbor}} - X_{\text{minority}}\right) \quad (2)$$

where X_{minority} is the original minority class data point, X_{neighbor} is one of the k-nearest neighbors of X_{minority}, λ is a random number between 0 and 1, which scales the difference between the minority data point and its neighbor to generate the new data point. This helps balance the dataset by generating synthetic samples for the minority class (attacks), ensuring that the model is exposed to more attack instances, leading to better performance in detecting rare intrusions.

2.4. LSTM Model

The LSTM network, a specialized type of Recurrent Neural Network (RNN), is particularly effective at processing sequential data, making it highly suitable for intrusion detection systems that handle time-series network traffic. Cyber-attacks typically unfold over time as sequences of events or gradual changes in network behavior, and the ability of LSTM to capture long-term temporal dependencies allows it to recognize such patterns effectively [35], [36]. By modeling these sequential dependencies, LSTM is able to detect subtle anomalies that may occur over extended time intervals, which is crucial for safeguarding network infrastructures [36], [37], [38]. The model architecture is composed of several key layers: an input layer, an LSTM layer with 50 units, a Dropout layer, and an output layer. The LSTM layer plays a pivotal role in capturing temporal dependencies by leveraging internal gating mechanisms, which include the forget gate, input gate, output gate, and cell state. These gates regulate the flow of information at each time step, ensuring that important data is retained and irrelevant data is discarded, thus enabling the model to learn complex temporal patterns over time [36], [38]. To prevent overfitting during training, a Dropout layer with a rate of 0.2 is incorporated, which randomly omits a portion of neurons in each training iteration [39]. The final output layer uses a sigmoid activation function, producing a probability score between 0 and 1, which classifies network traffic as either normal or attack traffic [35].

The effectiveness of LSTM in this context lies in its meticulous design to handle sequential data through its gating mechanisms. The forget gate discards irrelevant information, the input gate updates the cell state with new data, and the output gate determines the contribution of the current state to the final prediction. This structure enables the model to maintain a long-term memory of network events, which is critical for identifying patterns indicative of potential threats in network traffic [40], [41]. This capacity to model temporal dynamics makes LSTM particularly powerful for tasks such as anomaly

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

detection in network intrusion systems, where identifying complex attack patterns is key to timely detection.

The forget gate (f_t) is responsible for determining which parts of the previous cell state (C_{t-1}) should be discarded. This gate takes the previous hidden state (h_{t-1}) and the current input (x_t) as inputs, applies a weighted sum with a bias term, and then passes it through the sigmoid activation function. The sigmoid function squashes the result into a value between 0 and 1, where 0 means "completely forget" and 1 means "completely retain". The forget gate's output can be expressed in (3) as follow

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big) \quad (3)$$

Here, σ is the sigmoid activation function, W_f is the weight matrix for the forget gate, and b_f is the bias term. The forget gate's output controls how much of the previous memory should be kept for the current time step.

The input gate (i_t) as stated in formula (4), determines which new information from the current input (x_t) should be added to the cell state. Like the forget gate, the input gate also receives the previous hidden state (h_{t-1}) and the current input (x_t), applies a weighted sum, and passes it through the sigmoid function. The output of this gate is a value between 0 and 1, where 0 means "do not update the state" and 1 means "fully update the state".

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (4)$$

This controls the flow of information into the cell state, helping the network decide which new information should be stored.

The cell state (C_t) is a critical component of the LSTM, as it acts as the "memory" of the network. It is updated at each time step based on the forget and input gates. The cell state is updated by combining the previous cell state (C_{t-1}) scaled by the forget gate output (f_t), and the candidate cell state ($\widetilde{C_t}$), which is a potential new memory content scaled by the input gate (i_t). The candidate cell state ($\widetilde{C_t}$) is calculated by passing a weighted sum of the previous hidden state (h_{t-1}) and the current input (x_t) through the hyperbolic tangent function (tanh), which squashes the output to a range between -1 and 1, and mathematically formulated in (5) and (6) as follow

$$\widetilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (5)$$

then, the cell state is updated as:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C_t} \quad (6)$$

This update mechanism ensures that the LSTM can "forget" irrelevant information and "remember" important information from past time steps.

The output gate (o_t) is responsible for deciding what part of the cell state should be exposed to the next layer of the network or the next time step. The output gate receives the previous hidden state (h_{t-1}) and the current input (x_t) as inputs, applies a weighted sum, and passes it through the sigmoid function. The output gate formulated in (7) as follow

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (7)$$

The result of the output gate is then multiplied by the hyperbolic tangent of the current cell state ($\tanh(C_t)$), which produces the final hidden state (h_t) for the current time step. The final output of the LSTM is given by formula (8) as follow

$$h_t = o_t \cdot \tanh(C_t) \quad (8)$$

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

This output is used as the input for the next time step or as the final prediction in the case of sequence prediction tasks.

The LSTM's architecture of gates allows it to effectively capture long-term dependencies in sequential data, such as network traffic patterns. The forget and input gates provide a mechanism for the model to selectively retain or discard information from previous time steps based on its relevance to the current context. The cell state acts as a long-term memory store, while the output gate ensures that relevant information is passed on to the next time step. By processing network traffic as a sequence of time steps, LSTM can learn complex patterns and dependencies that are critical for detecting intrusions or anomalies in real-time network environments. Through these gating mechanisms, LSTM models can adapt to dynamic changes in network traffic, making them particularly effective for IDS, where identifying emerging and evolving attack patterns is crucial for cybersecurity. By learning from the temporal structure of the traffic, LSTM models can improve the detection of rare and complex network intrusions that may otherwise go unnoticed by traditional methods.

The Adam optimizer is a highly effective optimization algorithm widely used in deep learning due to its combination of momentum and RMSProp, making it particularly suitable for training models with large datasets and numerous parameters. Unlike traditional stochastic gradient descent (SGD), Adam computes adaptive learning rates for each parameter by utilizing both the first moment (mean) and second moment (variance) estimates of the gradients, which allows the model to efficiently update parameters during training [42]. By integrating these two strategies, Adam accelerates convergence and improves stability by smoothing gradient updates and dynamically adjusting the step size based on the squared gradients, which is beneficial for large and complex datasets [42]. The use of moment estimates enhances robustness by addressing issues such as sparse gradients and noisy updates, helping the optimizer avoid local minima or saddle points that can hinder training in deep neural networks. Its ease of implementation and minimal hyperparameter tuning requirement contribute to its widespread adoption across various applications, from computer vision to natural language processing [43], further solidifying its role as a default choice in deep learning frameworks. The first moment estimate (m_t) is an exponentially weighted average of the gradients, and it is updated at each time step using the formula:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \quad (9)$$

where β_1 controls the decay rate of past gradients, and g_t represents the gradient at time step (t). The second moment estimate (v_t) tracks the squared gradients and is updated using :

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \quad (10)$$

where β_2 controls the decay rate of squared gradients. To correct the bias introduced by the initial moment estimates, bias-corrected versions of m_t and v_t are calculated:

$$\widehat{m_t} = \frac{m_t}{1-\beta_1^t}, \quad \widehat{v_t} = \frac{v_t}{1-\beta_2^t} \quad (11)$$

Finally, the parameters of the model (θ_t) are updated using the following equation:

$$\theta_t = \theta_{t-1} - \alpha \cdot \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}+\epsilon} \quad (12)$$

where α is the learning rate, and ϵ is a small constant added to avoid division by zero. This update rule allows Adam to adjust the learning rate for each parameter based on its gradient and variance, helping the model converge more efficiently.

For the loss function, the model uses binary cross-entropy, a standard loss function for binary classification tasks. It measures the difference between the true labels and the predicted probabilities. The formula for binary cross-entropy is:

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

$$L = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)] \quad (13)$$

where L is the loss to be minimized, N is the number of samples, y_i is the true label for sample i (0 for normal traffic, 1 for attack), and p_i is the predicted probability for sample i being an attack. The loss function penalizes the model when its predictions are far from the true labels, and this encourages the model to learn better predictions over time. Through minimizing this loss, the LSTM model improves its ability to distinguish between normal traffic and attacks, which is critical for accurate network intrusion detection.

In addition to the LSTM model, a traditional machine learning model, LinearSVC, was also implemented for comparison purposes. LinearSVC is a variant of the SVM that works well with linearly separable data. It was chosen to demonstrate how a simpler, non-sequential algorithm compares to the LSTM in handling network intrusion detection tasks. The LinearSVC model was implemented using Scikit-learn's LinearSVC class with default hyperparameters. This includes the squared hinge loss, L2 penalty, and dual='auto', which allows the model to handle both dense and sparse datasets effectively. LinearSVC is generally faster than other SVM variants (like SVC with RBF kernel) and is suitable for large datasets where training time is a concern. It operates by finding the hyperplane that best separates the data into two classes (normal traffic and attack traffic) and works well when there is a clear margin of separation between the classes.

The models were trained and evaluated using different datasets and algorithms to explore how LSTM networks and LinearSVC handle imbalanced datasets versus those augmented with SMOTE. These four configurations were chosen to investigate the impact of both SMOTE (which aims to balance class distributions) and the type of algorithm used on the accuracy and effectiveness of detecting network intrusions. The first model configuration trained was the LSTM on the original (imbalanced) dataset. In this configuration, the LSTM model was trained on the original training data, where the class imbalance was not addressed. The imbalance was notable, as normal traffic instances vastly outnumbered attack traffic, which led to a skewed distribution. In this setup, the model learned to prioritize predicting normal traffic, and as expected, the detection of attack instances was suboptimal. This configuration served as a baseline, providing insight into how an LSTM model performs on imbalanced data and how it struggles to detect rare attack patterns, which is a common issue in IDS that use large, unbalanced datasets.

The second configuration involved training the LSTM on the SMOTE-augmented (balanced) dataset. After applying SMOTE, the class imbalance was addressed by generating synthetic samples for the minority class (attack traffic). This balancing process allowed the LSTM model to be trained on a dataset that contained an equal number of instances for normal traffic and attack traffic, offering a more representative sample of the data. The model could now learn to detect both classes more effectively, as it was no longer biased toward predicting the majority class. This configuration was expected to perform better than the first, as SMOTE helped mitigate the issue of imbalanced classes by allowing the model to better generalize attack patterns. The third model configuration used LinearSVC on the original (imbalanced) dataset. In this case, LinearSVC, a more traditional machine learning model based on a linear kernel, was applied to the same imbalanced dataset. LinearSVC is often used for classification tasks and can be much faster than complex models like LSTM, especially for smaller datasets. However, like LSTM, LinearSVC is also prone to class imbalance, and the model was likely to favor normal traffic predictions over attacks. This configuration served as a comparison to evaluate how LinearSVC, which typically performs well on linear decision boundaries, could handle the same imbalanced data and how it compared to the performance of the LSTM model on the same dataset. The final configuration involved training LinearSVC on the SMOTE-augmented (balanced) dataset. By applying SMOTE to the LinearSVC training data, the class imbalance was addressed, and the dataset was balanced, allowing

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

the model to have an equal representation of normal and attack traffic. This configuration was expected to improve the LinearSVC's performance, especially in detecting rare attacks. By comparing the results of this configuration with those from the LinearSVC on the imbalanced dataset, the study could assess how the inclusion of SMOTE influences the performance of a more traditional machine learning algorithm.

2.5. Evaluation Metrics

Once trained, all models are evaluated on the 20% hold-out test set, which was separated before the training phase to ensure an unbiased evaluation. The performance of each model is assessed using several key metrics. Accuracy is a fundamental metric that gives the overall correctness of the model's predictions, and is calculated as the ratio of correct predictions to the total number of predictions:

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{14}$$

where TP (True Positives) represents correctly identified attack instances, TN (True Negatives) indicates correctly identified normal traffic, FP (False Positives) are incorrectly identified attacks as normal traffic, and FN (False Negatives) are normal traffic incorrectly identified as attacks. While accuracy provides a general measure, precision and recall offer more detailed insights into the model's performance, particularly for imbalanced datasets.

Precision focuses on the proportion of true positives among all instances that were predicted as positive:

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{15}$$

This metric indicates how many of the predicted attacks are actually legitimate. Recall, on the other hand, measures the proportion of actual positives that were correctly identified by the model:

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{16}$$

This metric is crucial for understanding how well the model identifies all instances of attacks, even those that are rare. The F1-score combines precision and recall into a single metric by calculating their harmonic mean:

$$F1\text{-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision+Recall}} \tag{17}$$

This provides a balanced measure of the model's performance, especially when there is an imbalance between precision and recall. Finally, the confusion matrix is used to provide a more granular breakdown of the model's performance, showing the number of true positives, true negatives, false positives, and false negatives. This matrix helps to pinpoint the model's strengths and weaknesses, providing critical insights into where it is succeeding and where improvements are needed. By using these metrics, the model's performance can be thoroughly evaluated and refined to improve its ability to accurately detect network intrusions across both normal and attack traffic.

3. RESULT

3.1. Performance of Models on Imbalanced Data

This section examines the performance of two models, LSTM (Imbalanced) and LinearSVC (Imbalanced), trained on a significantly imbalanced dataset. The efficacy of these models is assessed utilising standard classification metrics: Accuracy, Precision, Recall, and F1-Score. These metrics assist

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

in assessing the models' overall efficacy regarding accurate predictions, the ratio of pertinent instances collected, and the capacity to differentiate between normal traffic and attacks.

The LSTM (Imbalanced) model achieves a high accuracy of 96.23%, which suggests that it correctly predicts a significant portion of the instances in the dataset. However, accuracy alone does not provide a complete picture, especially when dealing with imbalanced datasets where the minority class (attacks) is underrepresented. In this case, LSTM (Imbalanced) shows a Precision of 0.8802 for attack detection, indicating that when the model predicts an attack, it is correct approximately 88% of the time. This is a positive indicator of the model's ability to avoid false positives (incorrectly classifying normal traffic as attacks). The Recall for attack detection is 0.8991, which is fairly high, suggesting that LSTM is able to detect 90% of attack instances correctly. This shows that the model is quite capable of identifying the rare instances of attacks within a sea of normal traffic. The F1-Score of 0.8895 reflects a good balance between Precision and Recall, which is crucial in intrusion detection systems where both false positives and false negatives can have significant consequences. In this case, the model performs well in terms of both detecting attacks and minimizing false positives, making LSTM (Imbalanced) a strong contender for intrusion detection. However, it is important to note that the model still faces challenges inherent to imbalanced datasets. Confusion matrices for LSTM (Imbalanced) reveal that while the model has high True Positives (TP) for normal traffic, it also has a notable number of False Negatives (FN) for attack traffic. This means the model sometimes fails to detect certain attack instances, especially when the normal traffic instances overwhelmingly dominate the dataset. This challenge is typical of imbalanced datasets, where the model may develop a bias toward predicting the majority class (normal traffic).

The LinearSVC (Imbalanced) model shows a slightly lower accuracy (94.41%) compared to LSTM (Imbalanced), which indicates that LinearSVC does not perform as well overall. Precision for LinearSVC (Imbalanced) is 0.8646, which is lower than LSTM's precision, suggesting that when the model predicts an attack, it is less likely to be correct. While this value is still reasonable, it indicates a higher rate of false positives compared to LSTM. The Recall for LinearSVC (Imbalanced) is 0.7933, significantly lower than the Recall of 0.8991 observed in LSTM (Imbalanced). This shows that LinearSVC misses a larger proportion of attack instances, meaning the model fails to identify some attacks, leading to a higher number of False Negatives (FN). This lower recall indicates that LinearSVC (Imbalanced) is less effective at detecting rare attack events than LSTM (Imbalanced). The F1-Score of LinearSVC (Imbalanced) is 0.8274, which is also lower than LSTM's F1-Score. This value reflects the trade-off between Precision and Recall in LinearSVC (Imbalanced). While the Precision is decent, the significant drop in Recall leads to a poorer balance between these two metrics. In the confusion matrix for LinearSVC (Imbalanced), we observe a similar trend to LSTM (Imbalanced), where there are high True Positives (TP) for normal traffic but a significant number of False Negatives (FN) for attacks. This highlights the difficulty of detecting the minority class (attacks) when the dataset is imbalanced, even though LinearSVC is performing reasonably well with respect to the majority class.

The LSTM (Imbalanced) model outperforms LinearSVC (Imbalanced) in terms of Recall and F1-Score, making it more effective at detecting rare attack instances in the dataset. LSTM (Imbalanced) also achieves a high Precision and maintains a well-balanced performance between Precision and Recall, which is critical in network intrusion detection. However, both models face the challenge of imbalanced datasets, as evidenced by the False Negatives for attack detection, highlighting the need for further improvement in detecting minority class instances. In particular, addressing the False Negatives and improving Recall will be important to reduce the risk of missing critical attack instances in a real-world deployment. In the next section, we will evaluate the impact of using SMOTE to handle class imbalance and compare the performance of the models after applying this technique.

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

3.2. Impact of SMOTE on Model Performance

The implementation of SMOTE to rectify class imbalance in the training dataset resulted in alterations in the performance of both LSTM and LinearSVC models, as indicated by the evaluation metrics. The metrics encompass Accuracy, Precision, Recall, and F1-Score, which are essential for assessing the efficacy of IDS, particularly in the context of imbalanced datasets where one class (attack traffic) is under-represented relative to the majority class (normal traffic).

After applying SMOTE to the LSTM (SMOTE) model, the results show a slight improvement in Accuracy, which increased from 96.23% in the imbalanced model to 96.46%. This improvement suggests that the model's overall ability to classify both normal and attack traffic slightly improved after balancing the dataset. However, the most notable improvement comes in Recall, which increased from 89.91% to 94.55%. This substantial increase indicates that SMOTE helped the LSTM model detect a larger proportion of attack instances, addressing the False Negatives issue that was prevalent in the imbalanced dataset. However, this improvement in Recall comes with a slight trade-off in Precision. The precision dropped from 88.02% to 85.92%, meaning the model is now predicting more attacks, but some of those predictions are false positives. False positives occur when the model incorrectly classifies normal traffic as attacks. Despite this drop in precision, the F1-Score (which balances precision and recall) remains high at 0.9003, reflecting a good balance between the model's ability to detect attacks and its ability to minimize false positives. Thus, LSTM (SMOTE) provides a solid improvement in attack detection, but at the cost of introducing some false alarms. This trade-off between Precision and Recall is common when balancing datasets using SMOTE.

In the case of LinearSVC (SMOTE), the impact of SMOTE is more pronounced in terms of Recall improvement. The Recall for attack detection jumps dramatically from 79.33% in the imbalanced model to 96.65% in the SMOTE model. This indicates that SMOTE significantly improved the model's ability to detect attacks, successfully addressing the problem of missing attack instances. However, similar to LSTM (SMOTE), this improvement in recall comes with a noticeable drop in Precision. The Precision dropped from 86.46% to 71.14%, meaning the model now misclassifies a significant amount of normal traffic as attacks. As a result, LinearSVC (SMOTE) has a higher rate of False Positives compared to its performance on the imbalanced dataset. Despite the drop in precision, the F1-Score only slightly decreases from 0.8274 to 0.8195, indicating that the model still strikes a reasonable balance between detecting attacks and avoiding false positives, albeit with some performance degradation. The F1-Score remains a useful metric to assess the trade-off between Recall and Precision, and in this case, LinearSVC (SMOTE) shows an improvement in detecting rare attacks at the cost of falsely classifying normal traffic as attacks.

From the results, it is clear that applying SMOTE improved the Recall for both models, meaning they became better at detecting rare attacks, a key goal in intrusion detection systems. However, this came at the cost of Precision, with both models increasing their False Positive rates. This trade-off is a common challenge when applying SMOTE, as it generates synthetic data for the minority class, leading to the model predicting more instances of attack, some of which may be false positives. The F1-Score, which combines Precision and Recall, shows that LSTM (SMOTE) has a better balance compared to LinearSVC (SMOTE). LSTM (SMOTE) has a higher F1-Score of 0.9003, indicating it maintains a better trade-off between detecting attacks and avoiding false alarms. On the other hand, LinearSVC (SMOTE), despite achieving a higher Recall than LSTM, has a lower F1-Score of 0.8195, indicating it suffers more from the increase in False Positives. This suggests that LSTM (SMOTE) strikes a better balance between Precision and Recall when detecting attacks in the dataset. Applying SMOTE effectively improved the Recall for both models, helping them to identify a larger proportion of attack instances. However, this improvement came with a trade-off in Precision, leading to more false positives. The F1-Score reflects this trade-off, with LSTM (SMOTE) achieving a better overall performance in terms of balancing attack

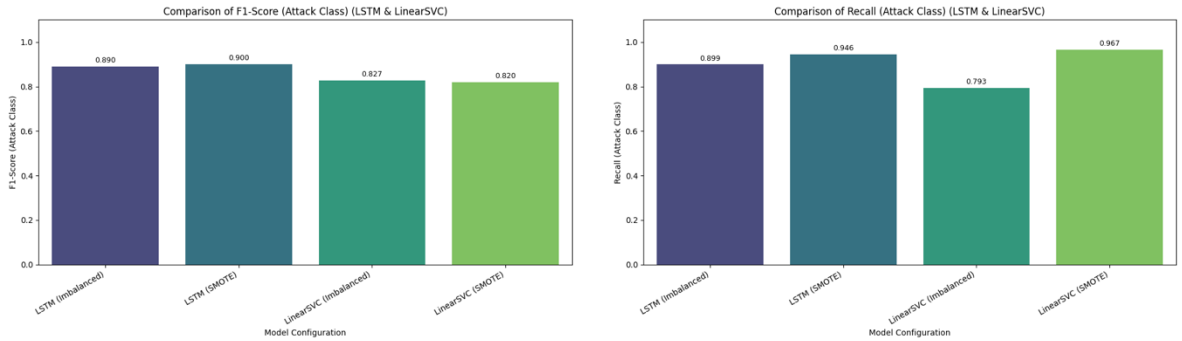
Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

detection and false alarms. While SMOTE improves detection for rare events (attacks), it also emphasizes the ongoing challenge in intrusion detection systems of balancing Recall and Precision. These results underline the need for further refinement in balancing models to minimize false positives while maximizing attack detection, particularly in real-world scenarios where both issues can have serious consequences.

3.3. Comparative Analysis

On the original unbalanced dataset, LSTM (unbalanced) performs noticeably better than LinearSVC (Imbalanced) in a number of important evaluation criteria. With a recall of 89.91%, LSTM outperforms LinearSVC, which has a recall of 79.33%. This suggests that even while regular traffic predominates in an unbalanced dataset, LSTM is more effective at detecting attack traffic. Additionally, LSTM's F1-Score is greater at 0.8895 than LinearSVC's, which is 0.8274. LSTM's higher score indicates that it offers a better balance between accurately detecting attacks and minimising false positives. The F1-Score is a balanced statistic that takes into account both Precision and Recall. This performance can be ascribed to the LSTM's capacity to detect intricate attack patterns in network traffic by capturing long-term dependencies in the sequential data. However, as evidenced by its poor Recall, LinearSVC has trouble identifying attacks. This is probably because models like LinearSVC have a limitation when dealing with sequential data, like network traffic logs, in that they are unable to capture the temporal dependencies that are present in the data.

For the SMOTE-augmented dataset, LSTM (SMOTE) still maintains its advantage over LinearSVC (SMOTE). The F1-Score for LSTM (SMOTE) is 0.9003, which is higher than LinearSVC (SMOTE)'s 0.8195, indicating that LSTM (SMOTE) strikes a better balance between Precision and Recall. While both models show improvement in Recall after applying SMOTE, LSTM (SMOTE) achieves a Recall of 94.55%, significantly better than LinearSVC (SMOTE)'s Recall of 96.65%. This improvement in Recall shows that SMOTE helped both models detect more attacks. However, the increase in Recall comes at a cost for LinearSVC (SMOTE), where Precision drops to 71.14%, down from 86.46% in the imbalanced dataset. This indicates that LinearSVC (SMOTE) is now predicting more attack instances, but it also incorrectly classifies a higher proportion of normal traffic as attacks, resulting in False Positives. In contrast, LSTM (SMOTE) suffers a smaller reduction in Precision (from 88.02% to 85.92%), indicating that it manages to detect attacks while keeping the False Positive rate relatively low compared to LinearSVC (SMOTE).



Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

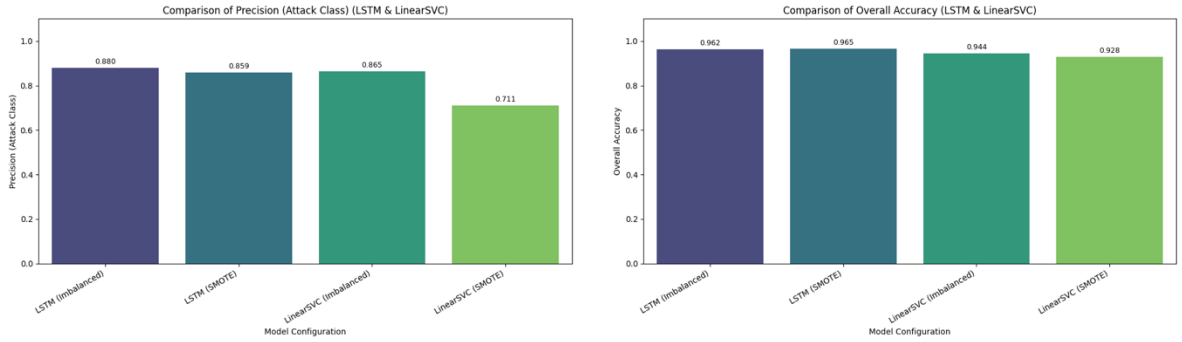


Figure 2. Comparison of a) F1-Score, b) Recall, c) Precision and d) Accuracy

Figure 2 (a-d) visually represent the performance of different machine learning models on various metrics (F1-Score, Recall, Precision, and Accuracy) for the "Attack" class in network intrusion detection. Figure 2.a compares the F1-scores for the "Attack" class across four models: LSTM (Imbalanced), LSTM (SMOTE), LinearSVC (Imbalanced), and LinearSVC (SMOTE). The F1-score combines both precision and recall into a single metric, providing a balanced measure of model performance. From the chart, we can observe that LSTM models (both Imbalanced and SMOTE) outperform the LinearSVC models. The F1-score for LSTM (SMOTE) is the highest, slightly better than LSTM (Imbalanced), indicating that applying SMOTE helps the LSTM model achieve a better balance between detecting attacks and avoiding false positives. On the other hand, the F1-scores for both LinearSVC models are lower, with SMOTE slightly improving the performance compared to the imbalanced data but still lagging behind the LSTM models. Figure 2.b illustrates the recall for the "Attack" class, which measures the model's ability to correctly identify all actual attack instances. Recall for the attack class is particularly important in network intrusion detection because missing an attack (false negative) can have severe consequences. From the chart, it's evident that the recall of LSTM models (Imbalanced and SMOTE) is high, with LSTM (SMOTE) showing the best recall, followed by LSTM (Imbalanced). This suggests that SMOTE helps the LSTM model detect a higher percentage of attacks. In contrast, LinearSVC models show much lower recall, especially the imbalanced version, which indicates that the model misses many attack instances. The SMOTE version of LinearSVC improves recall but still lags behind LSTM.

Figure 2.c compares the precision for the "Attack" class, which evaluates the proportion of true positive attacks among all instances predicted as attacks. Higher precision means that fewer normal instances are misclassified as attacks (false positives). The LSTM models (Imbalanced and SMOTE) both demonstrate high precision, with LSTM (Imbalanced) slightly outperforming LSTM (SMOTE). This suggests that LSTM models are good at correctly identifying attack instances without misclassifying too many normal ones. However, LinearSVC, particularly on SMOTE-augmented data, shows lower precision, indicating that it tends to misclassify more normal traffic as attacks, even though the recall is higher. This suggests a trade-off between recall and precision for the LinearSVC model when using SMOTE. Figure 2.d presents the overall accuracy for each model configuration, which reflects the proportion of correct predictions (both attack and normal traffic) made by the model. The accuracy of the LSTM models (both Imbalanced and SMOTE) is very high, with LSTM (SMOTE) showing a slight improvement over the Imbalanced model. However, the accuracy for LinearSVC models is lower, particularly for the SMOTE version, despite the increased recall. This suggests that while SMOTE improves recall, it may have caused a decrease in overall accuracy due to the model misclassifying more normal traffic as attacks. The LSTM models maintain a high level of accuracy, suggesting they strike a better balance between detecting attacks and correctly classifying normal traffic.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

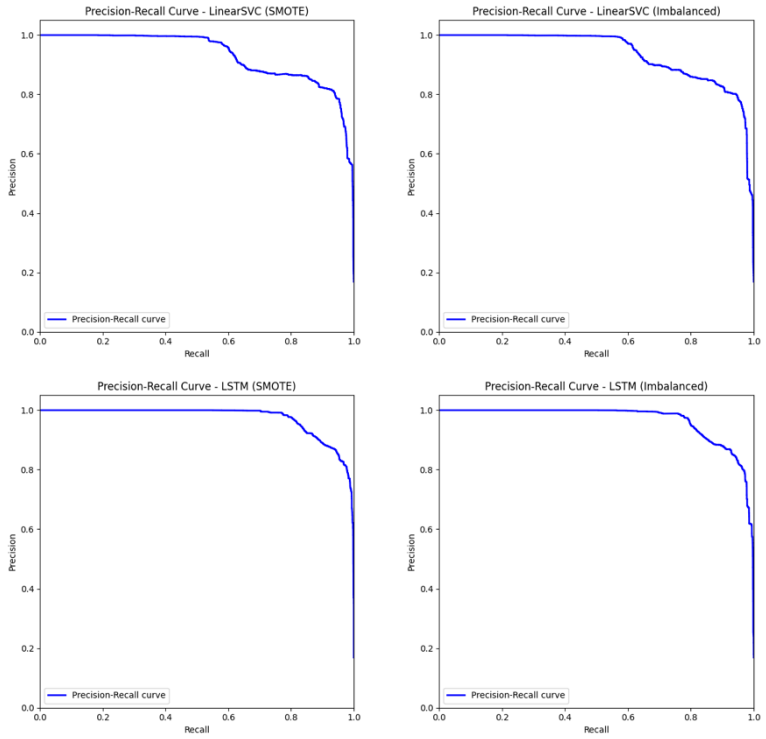


Figure 3. Precision-Recall curves of a) LinearSVC (SMOTE), b) LinearSVC (Imbalance), c) LSTM (SMOTE), d) LSTM (Imbalance)

Figure 3 (a-d) provide valuable insights into how each model performs in terms of detecting network intrusions, particularly for the attack class (the minority class). The Precision-Recall curve is a crucial tool for evaluating models, especially when dealing with imbalanced datasets, where the number of normal traffic instances vastly outweighs the number of attack instances. For LinearSVC with SMOTE (Figure 3.a), the Precision-Recall curve demonstrates the model's tendency to start with very high precision when recall is low, indicating that the model is conservative about predicting attacks. However, as the recall increases, precision drops significantly, which is typical in models trained with SMOTE. SMOTE aims to balance the dataset by creating synthetic examples of the minority class, thus improving recall. However, this results in a higher number of false positives, as the model starts to predict more instances as attacks, even if they are not, which lowers precision. This sharp decline in precision as recall increases is a hallmark of SMOTE's effect on model behavior, pushing the model to detect more attacks at the cost of increasing false positives. The LinearSVC with Imbalanced Data (Figure 3.b) shows a similar trend, with high precision but much lower recall. Since the model is trained on an imbalanced dataset, it tends to be biased towards predicting the majority class (normal traffic), leading to high precision (since it correctly classifies normal traffic as non-attacks) but poor recall for the minority attack class. This illustrates a common problem with imbalanced datasets: the model is not effectively learning to detect rare attack patterns, as it is overly focused on the more prevalent normal traffic class.

The LSTM model trained with SMOTE (Figure 3.c) shows a somewhat better balance between precision and recall compared to LinearSVC, although it still suffers from the precision-recall trade-off. The curve starts with high precision and drops gradually as recall increases. The LSTM model, which is designed to capture temporal patterns in sequential data, benefits from the SMOTE technique by learning better from the augmented attack class data. However, as in the LinearSVC model, the improvement in recall comes at the cost of a decrease in precision, showing that the model is detecting more attacks but also making more false positive predictions. For LSTM with Imbalanced Data (Figure

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

3.d), the Precision-Recall curve mirrors that of the LinearSVC with imbalanced data. Precision is high at low recall levels, meaning the model is cautious about predicting attacks. However, as recall increases, precision drops significantly, highlighting the challenge of class imbalance. The model struggles to detect rare attack instances because it is predominantly trained on the majority class, leading to poor recall. The comparison of these curves underscores the importance of handling class imbalance, which significantly impacts model performance, particularly for minority class detection. The use of SMOTE improves recall, allowing the models to better identify attacks, but it also increases false positives, which is reflected in the drop in precision. Among the models, LSTM generally performs better at balancing the precision-recall trade-off compared to LinearSVC, especially when SMOTE is applied, indicating that LSTM's architecture is more suited for sequential data with temporal dependencies, such as network traffic. The Precision-Recall curves clearly demonstrate the trade-offs between precision and recall for each model. The application of SMOTE helps to address the class imbalance issue by increasing recall but at the expense of precision, as the models become more aggressive in predicting attacks. The LSTM models, particularly when trained with SMOTE, provide a better balance between detecting attacks and reducing false positives compared to the LinearSVC models, confirming that LSTM is better suited for capturing complex patterns in imbalanced datasets.

4. DISCUSSIONS

4.1. Interpretation of Results

The application of the SMOTE had a noticeable impact on the models, particularly in terms of recall. As expected, SMOTE addressed the class imbalance issue by generating synthetic samples for the minority attack class, which allowed both LSTM and LinearSVC models to better identify attack instances. The results showed a marked improvement in recall for both models, indicating that SMOTE helped them recognize more attacks, which were previously underrepresented in the training data. However, this improvement in recall came at the cost of reduced precision, particularly for LinearSVC. This trade-off between recall and precision is expected when using SMOTE, as the model starts to predict more instances as attacks, which may include more false positives. Despite this, SMOTE significantly enhanced the models' ability to detect attacks, which is crucial in real-world intrusion detection scenarios, where missing an attack (false negative) can be much more costly than a false alarm (false positive). The effectiveness of SMOTE was more pronounced in LSTM compared to LinearSVC. LSTM (SMOTE) demonstrated an impressive increase in recall without a severe drop in precision. In contrast, LinearSVC (SMOTE) saw a substantial increase in recall but with a more significant reduction in precision. This suggests that LSTM's architecture, designed to capture temporal patterns in sequential data, was better able to integrate the synthetic data created by SMOTE, improving its attack detection capabilities without overfitting to the synthetic examples. On the other hand, LinearSVC, which works well for linearly separable data, struggled more with the high-dimensional feature space after SMOTE augmentation, leading to a greater trade-off between recall and precision.

When comparing the two models, LSTM and LinearSVC, for this intrusion detection task, LSTM emerged as the more suitable model. LSTM is particularly effective for handling sequential data, like network traffic, where patterns over time are crucial for detecting anomalies. In this study, the LSTM model (SMOTE) performed well in terms of recall, capturing more attack instances compared to the LinearSVC model (SMOTE), which demonstrated a higher tendency to miss out on attack cases due to its inability to model temporal dependencies effectively. The ability of LSTM to handle sequences of data and capture long-term dependencies makes it an ideal candidate for network intrusion detection tasks, where attacks often unfold over time in patterns that need to be detected early. In contrast, LinearSVC performed well on imbalanced data, achieving high precision by correctly classifying the majority class (normal traffic). However, its performance in terms of recall for the attack class was

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

relatively low, even after applying SMOTE. This suggests that LinearSVC, despite its fast training and effectiveness in simpler classification tasks, is less suited for tasks that require understanding complex sequential dependencies. In scenarios where capturing patterns over time is essential—such as detecting network intrusions—LinearSVC might struggle, as it is not inherently designed to handle sequential or time-series data.

The LSTM model demonstrated its potential in the intrusion detection task, especially after applying SMOTE. Although the recall was lower than ideal, the increase in recall after SMOTE suggested that LSTM was able to learn more effectively from the minority attack class, capturing attack patterns even when those instances were underrepresented. The F1-score for LSTM (SMOTE) also improved, highlighting the model's ability to balance the trade-off between precision and recall. LSTM's ability to capture sequential patterns was crucial in this study, as attacks in network traffic often manifest over time through complex and subtle changes in traffic patterns. Even though LSTM may not have fully exploited its temporal modeling capabilities in this simplified setting, its performance—especially in recall and F1-score—indicated that it was better suited for handling sequential data and imbalanced datasets compared to LinearSVC. LinearSVC demonstrated strong performance on the imbalanced dataset, particularly in terms of precision. However, its recall was relatively low, indicating that the model had difficulty detecting the minority attack class, which was expected given the class imbalance in the dataset. After applying SMOTE, LinearSVC achieved a significant boost in recall, but this came at the expense of a noticeable drop in precision. The Precision-Recall curve for LinearSVC (SMOTE) showed that the model became more aggressive in predicting attacks, but many of these predictions were false positives, which led to a lower precision. The performance trade-off seen in LinearSVC was largely due to its linear nature, which struggles when dealing with high-dimensional, synthetic features generated by SMOTE. Unlike LSTM, which can better handle such data by modeling temporal dependencies, LinearSVC was not as effective at leveraging the synthetic data to improve attack detection.

In the context of network intrusion detection, recall and F1-score are far more important metrics than accuracy. The goal of an intrusion detection system is to detect as many attacks as possible (high recall), even if that means generating a few false positives (low precision). The F1-score, which balances precision and recall, is a better indicator of a model's overall ability to detect attacks while minimizing false alarms. Accuracy, on the other hand, is less meaningful in imbalanced datasets because it can be biased towards predicting the majority class (normal traffic). A model with high accuracy may still fail to detect most attacks, leading to a false sense of security. The LSTM model (SMOTE) achieved a favorable balance between recall and precision, as reflected in its higher F1-score compared to LinearSVC after SMOTE, indicating that it is better suited for detecting attacks in imbalanced datasets. The results of this study align well with the chosen methodology, which focused on utilizing SMOTE to handle class imbalance and LSTM for its ability to capture temporal dependencies. The LSTM architecture was particularly effective in capturing the sequential nature of network traffic, which is crucial for detecting complex and evolving network intrusions. The preprocessing steps, including feature selection, handling missing data, and scaling, ensured that the data fed into the model was clean, balanced, and ready for training. The use of SMOTE was particularly important in this study, as it addressed the class imbalance, enabling both models to learn more effectively from the minority class. However, the trade-offs observed in LinearSVC (SMOTE) emphasized that, while SMOTE can improve recall, it may lead to a decline in precision, especially for models that are not inherently designed to handle complex sequential data. The results reinforce the idea that for network intrusion detection, models like LSTM that can capture temporal patterns are more suitable than simpler models like LinearSVC when dealing with imbalanced datasets and time-series data.

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

4.2. Comparison with Existing Approaches

In comparison to traditional signature-based IDS, which rely on predefined attack signatures, the LSTM-based approach offers a distinct advantage, especially in detecting novel or polymorphic attacks. Signature-based systems are highly dependent on having a prior knowledge of the attack signatures, which makes them ineffective against new, unknown, or rapidly evolving attack patterns. This limitation is particularly evident in the face of modern cyber-attacks, which often evolve or change to avoid detection. LSTM, on the other hand, is adept at processing sequential data and capturing temporal dependencies within network traffic, making it more suitable for detecting complex and evolving attack patterns that might otherwise go unnoticed by traditional systems. Despite these advantages, the LSTM model still faces challenges, particularly with rare attack types in imbalanced datasets. In such datasets, the model is prone to bias toward detecting the majority class, which in this case is normal traffic. Some recent hybrid models, combining LSTM with other machine learning algorithms like Random Forest or XGBoost, have shown promise in improving detection performance by leveraging the strengths of multiple models. These hybrid approaches have been found to be effective in improving the recall and precision for the minority class, which is essential for detecting rare or complex attacks [5], [11]. Future work in IDS could explore such hybrid models further to improve detection capabilities, particularly for low-frequency attacks.

4.3. Limitations

The limitations of this study include the use of only a 10% subset of the CICIDS 2017 dataset, which may not fully capture the complexity and variability of the entire dataset, potentially affecting the generalizability of the results. Additionally, the study relied on a predefined set of features, and alternative feature selection methods might lead to different outcomes. The LSTM architecture used was relatively simple, consisting of a single layer with 50 units and a single time-step view; more complex architectures or variations in sequence length could potentially improve performance. Moreover, the models were trained with fixed hyperparameters such as the number of epochs, batch size, and default settings for LinearSVC, without extensive hyperparameter tuning, which could have optimized model performance further. Lastly, the study applied only the basic SMOTE algorithm, whereas other advanced over-sampling techniques or combined approaches, such as Borderline-SMOTE or Adaptive Synthetic Sampling (ADASYN), may yield better results in handling class imbalance.

4.4. Implications for Network Security

This study highlights the significant potential of using LSTM combined with SMOTE for improving network intrusion detection, especially for complex and evolving attack patterns. The ability of LSTM to capture temporal dependencies within network traffic allows it to detect attack patterns that are dynamic and may not fit traditional attack signatures. However, for the system to be fully effective in real-world environments, further refinements are necessary, particularly in improving the recall for detecting attacks. Real-world IDS must be capable of identifying attacks not only accurately (precision) but also comprehensively (recall) to ensure that all possible threats are detected. To further improve the model, continuous training with updated data, incorporating more diverse datasets, and experimenting with hybrid models combining LSTM with other machine learning techniques like Random Forest or XGBoost could be beneficial. Additionally, integrating data from various sources such as user activity logs or IoT traffic could enrich the model's ability to detect a broader range of attack types and make it more adaptable to different network environments. The results from this study suggest that combining LSTM with techniques like SMOTE could provide a more accurate and timely approach to detecting network intrusions, which is crucial for reducing the risk of cyber-attacks and data breaches [36], [44].

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

This study enhances IDS utilising sequential learning by illustrating the efficacy of integrating LSTM networks with SMOTE to tackle the issue of class imbalance in network traffic data. In contrast to conventional IDS techniques that focus mostly on static attack signatures, LSTM's capacity to identify temporal relationships in network data facilitates the detection of dynamic and intricate assault patterns. This study distinguishes itself from prior research by concentrating on the incorporation of SMOTE, which equilibrates class distribution and improves the identification of infrequent yet significant attacks. Although LSTM has demonstrated potential in enhancing the identification of attack patterns, subsequent research may expand upon these results by investigating hybrid models and integrating a broader array of data sources, thereby better optimising the model's recall and flexibility across multiple network contexts.

5. CONCLUSION

In this study, we compared the performance of LSTM and LinearSVC models for network intrusion detection using the CICIDS 2017 dataset. The results highlighted the effectiveness of SMOTE in improving recall, especially for LSTM, which showed a noticeable improvement in detecting attack traffic after applying SMOTE. While LinearSVC demonstrated high precision, its recall was lower, particularly with the SMOTE dataset, where it struggled with balancing between precision and recall. LSTM, on the other hand, had a more favorable F1-score and recall, indicating its suitability for identifying rare attack types, though further improvements in precision are needed. Overall, LSTM showed more promise in capturing temporal dependencies and attack patterns, especially when balanced with SMOTE, but the trade-offs between precision and recall still require careful consideration in practical applications. Looking ahead, several avenues for improvement can be explored in future work. Using the full dataset or larger samples could provide a more comprehensive understanding of model performance, especially for rare attack types. Exploring advanced feature engineering and selection techniques may uncover more discriminative features to improve model accuracy. Additionally, the performance of LSTM could be enhanced by implementing more complex architectures like stacked LSTMs or Bi-LSTMs, or by integrating other deep learning models such as CNN-LSTM or Transformers, which may better capture complex patterns. Extensive hyperparameter optimization could also be performed to fine-tune models for better accuracy. Future research could consider different sampling techniques, such as ADASYN, Borderline-SMOTE, or Tomek Links, to address class imbalance more effectively. Lastly, extending the approach to multi-class classification would allow the identification of specific attack types, providing a more detailed analysis of network threats. This research underscores the urgent need for advanced, adaptable intrusion detection systems that can effectively combat the increasing sophistication of cyber-attacks, emphasizing the importance of leveraging cutting-edge techniques like LSTM and SMOTE to enhance network security in real-time environments.

CONFLICT OF INTEREST

The authors declares that there is no conflict of interest between the authors or with research object in this paper.

REFERENCES

[1] R. L. Okyere, "Efficient Data Hiding Scheme Using Steganography and Cryptography Technique," *Adv. Multidiscip. Sci. Res. J. Publ.*, 2022, doi: 10.22624/aims/digital/v10n4p4.
[2] M. A. Hayudini, "Network Infrastructure Management: Its Importance to the Organization," *Nat. Sci. Eng. Technol. J.*, 2021, doi: 10.37275/nasetjournal.v2i1.15.

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

[3] D.-H. Park, "Virtuality Changes Consumer Preference: The Effect of Transaction Virtuality as Psychological Distance on Consumer Purchase Behavior," *Sustainability*, 2019, doi: 10.3390/su11236618.
[4] A. Priambodo, N. Anwar, and S. Suharno, "Is GRDP a Mediating Factor in Enhancing Local Tax Revenues Due to ICT Development in Indonesia?," *Nurture*, 2024, doi: 10.55951/nurture.v18i3.722.
[5] A. I. A. Alzahrani, M. Ayadi, M. M. Asiri, A. Al-Rasheed, and A. Ksibi, "Detecting the Presence of Malware and Identifying the Type of Cyber Attack Using Deep Learning and VGG-16 Techniques," *Electronics*, 2022, doi: 10.3390/electronics11223665.
[6] M. J. Awan *et al.*, "Real-Time DDoS Attack Detection System Using Big Data Approach," *Sustainability*, 2021, doi: 10.3390/su131910743.
[7] Y. Liu and L. Zhu, "A New Intrusion Detection and Alarm Correlation Technology Based on Neural Network," *Eurasip J. Wirel. Commun. Netw.*, 2019, doi: 10.1186/s13638-019-1419-z.
[8] Y. Liu and Y. Guo, "Towards Real-Time Warning and Defense Strategy AI Planning for Cyber Security Systems Aided by Security Ontology," *Electronics*, 2022, doi: 10.3390/electronics11244128.
[9] H. Wang and X. Li, "Optimization of Network Security Intelligent Early Warning System Based on Image Matching Technology of Partial Differential Equation," *J. Cyber Secur. Mobil.*, 2024, doi: 10.13052/jcsm2245-1439.1336.
[10] Z. Li, "A Neighbor Propagation Clustering Algorithm for Intrusion Detection," *Rev. Intell. Artif.*, 2020, doi: 10.18280/ria.340311.
[11] L. Gong, T. Xu, W. Zhang, X. Li, X. Wang, and W. Pan, "Approach Research on the Techniques for Network Intrusion Detection Based on Data Mining," 2015, doi: 10.2991/asei-15.2015.418.
[12] V. Shah, A. K. Aggarwal, and N. K. Chaubey, "Performance Improvement of Intrusion Detection With Fusion of Multiple Sensors," *Complex Intell. Syst.*, 2016, doi: 10.1007/s40747-016-0033-5.
[13] O. Ambavkar, P. Bharti, A. K. Chaurasiya, R. Chauhan, and M. Palinje, "Review on IDS Based on ML Algorithms," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 10, no. 11, pp. 169–174, 2022, doi: 10.22214/ijraset.2022.47284.
[14] S. H. Oh, J. Kim, J. H. Nah, and J. Park, "Employing Deep Reinforcement Learning to Cyber-Attack Simulation for Enhancing Cybersecurity," *Electronics*, vol. 13, no. 3, p. 555, 2024, doi: 10.3390/electronics13030555.
[15] W. Okori and S. Buteraba, "Cyber Security Exploits and Management in Telecommunication Companies: The Case of Uganda," *J. Comput. Sci. Technol. Stud.*, 2024, doi: 10.32996/jcsts.2024.6.4.10.
[16] A. D. Riyanto, A. M. Wahid, and A. A. Pratiwi, "ANALYSIS OF FACTORS DETERMINING STUDENT SATISFACTION USING DECISION TREE, RANDOM FOREST, SVM, AND NEURAL NETWORKS: A COMPARATIVE STUDY," *J. Tek. Inform. Jutif*, vol. 5, no. 4, Art. no. 4, Jul. 2024, doi: 10.52436/1.jutif.2024.5.4.2188.
[17] A. Wijayanto, I. Riadi, Y. Prayudi, and T. Sudinugraha, "Network Forensics Against Address Resolution Protocol Spoofing Attacks Using Trigger, Acquire, Analysis, Report, Action Method," *Regist. J. Ilm. Teknol. Sist. Inf.*, 2023, doi: 10.26594/register.v8i2.2953.
[18] S. F. Pratama and A. M. Wahid, "Fraudulent Transaction Detection in Online Systems Using Random Forest and Gradient Boosting," *J. Cyber Law*, vol. 1, no. 1, Art. no. 1, Mar. 2025.
[19] H. H. Ibrahim *et al.*, "A Comprehensive Study of Distributed Denial-of-Service Attack With the Detection Techniques," *Int. J. Electr. Comput. Eng. Ijece*, 2020, doi: 10.11591/ijece.v10i4.pp3685-3694.
[20] A. Dogra and Taqdir, "DDOS Attack Detection and Handling Mechanism in WSN," *Int. J. Recent Technol. Eng.*, 2019, doi: 10.35940/ijrte.c5644.098319.
[21] R. Singh and S. Kumar, "A Novel Algorithm to Detect Replay Attack in WLANs," *Int. J. Eng. Adv. Technol.*, 2019, doi: 10.35940/ijeat.a1437.109119.
[22] A. M. Wahid, L. Afuan, and F. S. Utomo, "ENHANCING COLLABORATION DATA MANAGEMENT THROUGH DATA WAREHOUSE DESIGN: MEETING BAN-PT ACCREDITATION AND KERMA REPORTING REQUIREMENTS IN HIGHER

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

EDUCATION," *J. Tek. Inform. Jutif*, vol. 5, no. 6, Art. no. 6, Dec. 2024, doi: 10.52436/1.jutif.2024.5.6.1747.

[23] Ö. Aslan, S. S. Aktuğ, M. Ozkan-Okay, A. A. Yılmaz, and E. Akin, "A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions," *Electronics*, vol. 12, no. 6, p. 1333, 2023, doi: 10.3390/electronics12061333.

[24] Y. Liu and S. Li, "Hybrid Cyber Threats Detection Using Explainable AI in Industrial IoT," 2023, doi: 10.1109/hccs59561.2023.10452621.

[25] N. Gao, L. Gao, Q. Gao, and H. Wang, "An Intrusion Detection Model Based on Deep Belief Networks," 2014, doi: 10.1109/cbd.2014.41.

[26] V. Hnamte, H.-N. Nguyen, J. Hussain, and Y. Kim, "A Novel Two-Stage Deep Learning Model for Network Intrusion Detection: LSTM-AE," *Ieee Access*, 2023, doi: 10.1109/access.2023.3266979.

[27] M. Moukhafi, M. Tantaoui, I. Chana, and A. Bouazi, "Intelligent Intrusion Detection Through Deep Autoencoder and Stacked Long Short-Term Memory," *Int. J. Electr. Comput. Eng. Ijece*, 2024, doi: 10.11591/ijece.v14i3.pp2908-2917.

[28] M. N. Arefin and A. K. Muhammad Masum, "A Probabilistic Approach for Missing Data Imputation," *Complexity*, 2024, doi: 10.1155/2024/4737963.

[29] B. Deore and S. Bhosale, "Hybrid Optimization Enabled Robust CNN-LSTM Technique for Network Intrusion Detection," *Ieee Access*, 2022, doi: 10.1109/access.2022.3183213.

[30] H. M. Kamal and M. Mashaly, "Advanced Hybrid Transformer-CNN Deep Learning Model for Effective Intrusion Detection Systems With Class Imbalance Mitigation Using Resampling Techniques," *Future Internet*, 2024, doi: 10.3390/fi16120481.

[31] Y. Zhang, L. Zhang, and X. Zheng, "Enhanced Intrusion Detection for ICS Using MS1DCNN and Transformer to Tackle Data Imbalance," *Sensors*, 2024, doi: 10.3390/s24247883.

[32] S. Kudithipudi, N. Narisetty, G. R. Kancherla, and B. Bobba, "Evaluating the Efficacy of Resampling Techniques in Addressing Class Imbalance for Network Intrusion Detection Systems Using Support Vector Machines," *Ingénierie Systèmes Inf.*, 2023, doi: 10.18280/isi.280511.

[33] Y. G. Damtew and H. Chen, "SMMO-CoFS: Synthetic Multi-Minority Oversampling With Collaborative Feature Selection for Network Intrusion Detection System," *Int. J. Comput. Intell. Syst.*, 2023, doi: 10.1007/s44196-022-00171-9.

[34] A. Benchama and K. Zebbara, "Fine-Tuning CNN-BiGRU for Intrusion Detection With SMOTE Optimization Using Optuna," *Salud Cienc. Tecnol. - Ser. Conf.*, 2024, doi: 10.56294/sctconf2024968.

[35] A. D. Vibhute *et al.*, "An <scp>LSTM</Scp>-based Novel Near-real-time Multiclass Network Intrusion Detection System for Complex Cloud Environments," *Concurr. Comput. Pract. Exp.*, 2024, doi: 10.1002/cpe.8024.

[36] S. Alsudani and A. Ghazikhani, "Enhancing Intrusion Detection With LSTM Recurrent Neural Network Optimized by Emperor Penguin Algorithm," *Wasit J. Comput. Math. Sci.*, 2023, doi: 10.31185/wjcms.166.

[37] S. F. Pratama and A. M. Wahid, "Mining Public Sentiment and Trends in Social Media Discussions on Indonesian Presidential Candidates Using Support Vector Machines," *J. Digit. Soc.*, vol. 1, no. 2, Art. no. 2, Jun. 2025, doi: 10.63913/jds.v1i2.8.

[38] Y. Li, R. Zhang, P. Zhao, and Y. Wei, "Feature-Attended Federated LSTM for Anomaly Detection in the Financial Internet of Things," *Appl. Sci.*, 2024, doi: 10.3390/app14135555.

[39] T. Arjunan, "Real-Time Detection of Network Traffic Anomalies in Big Data Environments Using Deep Learning Models," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 12, no. 3, pp. 844–850, 2024, doi: 10.22214/ijraset.2024.58946.

[40] Y. Zhang, J. Wang, Y. Chen, H. Yu, and T. Qin, "Adaptive Memory Networks With Self-Supervised Learning for Unsupervised Anomaly Detection," *Ieee Trans. Knowl. Data Eng.*, 2023, doi: 10.1109/tkde.2021.3139916.

[41] Y. Fan, L. Zhang, and K. Li, "AE-BiLSTM: Multivariate Time-Series EMI Anomaly Detection in 5g-R High-Speed Rail Wireless Communications," 2024, doi: 10.1109/iccworkshops59551.2024.10615719.

Vol. 6, No. 6, December 2025, Page. 5349-5370
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.6.4672

[42] Y. Shao *et al.*, "An Improved BGE-Adam Optimization Algorithm Based on Entropy Weighting and Adaptive Gradient Strategy," *Symmetry*, 2024, doi: 10.3390/sym16050623.
[43] D. Shulman, "Optimization Methods in Deep Learning: A Comprehensive Overview," 2023, doi: 10.48550/arxiv.2302.09566.
[44] G. Li and Y. Dai, "Time Series Anomaly Detection Using LSTM and Attention," 2024, doi: 10.1117/12.3035015.