Available online at website: https://jurnal.iaii.or.id/index.php/RESTI

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 10 No. 3 (2025) 617 - 625 e-ISSN: 2580-0760

K-Nearest Neighbor Performance Optimization for Multiclass Imbalance of Intrusion Detection Data Using SMOTE and Distance Variation-Based Parameter Tuning

Hairani Hairani[1], Christopher Michael Lauw[2], Sri Farida Utami[3], Afrig Aminuddin[4], Abu Tholib[5]
[1 2]Department of Computer Science, Faculty of Engineering, Universitas Bumigora, Mataram, Indonesia
[3]Department of Digital Business, Politeknik Sukabumi, Kota Sukabumi, Indonesia
[4]Department of Information System, Faculty of Computer Science, Universitas Amikom Yogyakarta, Yogyakarta, Indonesia
[5]Department of Informatics Engineering, Faculty of Engineering, Universitas Nurul Jadid, Probolinggo, Indonesia

[*]*hairani@universitasbumigora.ac.id*

Abstract

The increasing use of computer networks and internet-based services has made cybersecurity threats more complex. Intrusion Detection Systems (IDS) play a crucial role in identifying network attacks; however, conventional signature- or rule-based approaches are limited in handling novel attacks and dynamically changing attack patterns. Therefore, machine learning approaches are applied to enhance the adaptive capabilities of IDS. Nevertheless, the use of machine learning in IDS still faces a major challenge: data imbalance, where normal traffic significantly outweighs attack traffic. This condition biases models toward the majority class, leading to suboptimal detection of minority attacks. Based on this issue, this study aims to improve the performance of the K-Nearest Neighbor (KNN) method in network attack detection by applying the Synthetic Minority Over-sampling Technique (SMOTE) and parameter tuning. The study employs KNN with parameter tuning and SMOTE to address multiclass data imbalance in network attack detection. Parameter tuning is conducted to determine the optimal value of k and distance functions, including Euclidean, Manhattan, and Cosine Similarity. The results show that KNN with k = 3 and Manhattan distance on SMOTE-balanced data achieves the highest accuracy of 96.51%, outperforming Euclidean and Cosine Similarity distances. These findings conclude that applying SMOTE and appropriately selecting k and distance metrics significantly improve KNN performance in network attack detection and increase overall detection accuracy.

Keywords: intrusion detection system; KNN method; multi class imbalance; SMOTE; tuning parameter

How to Cite: H. Hairani, Christopher Michael Lauw, Sri Farida Utami, Afrig Aminuddin, and Abu Tholib, "K-Nearest Neighbor Performance Optimization for Multiclass Imbalance of Intrusion Detection Data Using SMOTE and Distance Variation-Based Parameter Tuning", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 10, no. 3, pp. 617 - 625, Jun. 2026.
Permalink/DOI: https://doi.org/10.29207/resti.v10i3.7489

Received: January 14, 2026
Accepted: May 24, 2026
Available Online: June 8, 2026

This is an open-access article under the CC BY 4.0 License
Published by Ikatan Ahli Informatika Indonesia

1. Introduction

The increasing reliance on computer networks, internet-based services, and distributed systems has caused cybersecurity threats to become increasingly complex and dynamic [1]. Intrusion Detection Systems (IDS) play an essential role in the early detection of suspicious activities and network attacks [2]. However, conventional IDS approaches based on signatures or rule-based mechanisms have limitations in addressing new attacks (zero-day attacks) and rapidly changing attack patterns [3]. Therefore, machine learning approaches are required to enhance the ability of IDS to learn network traffic patterns adaptively and automatically [4].

Nevertheless, the application of machine learning in IDS faces various challenges, particularly data imbalance, where normal traffic is far more dominant than attack data [5], [6]. Data imbalance, high dimensionality, and overlapping patterns cause the model to be biased towards the majority class, increasing classification errors and decreasing the effectiveness of IDS in detecting minority attack [7], [8].

Various previous studies have utilized machine learning methods to detect intrusions. Study [9] discusses machine learning–based network intrusion detection using random oversampling. Study [10] employs the Random Forest method with 12 features for network attack detection. Study [11] uses Artificial Neural Networks (ANN) combined with sampling techniques

for network attack detection, achieving an accuracy of 87%. Study [12] proposes a hybrid model based on CNN and LSTM for network attack detection.

Most prior studies focus on applying classification methods combined with data balancing techniques, without optimizing or tuning classification models in multiclass imbalanced IDS scenarios. In fact, parameter optimization plays a significant role in improving model performance [13].

In addition, recent studies integrating SMOTE with advanced optimization techniques, such as Bayesian optimization, have shown that parameter tuning plays an important role in improving detection performance [14]. Another recent study highlights the effectiveness of the K-Means-SMOTE oversampling technique in improving the performance of machine learning methods without parameter tuning [15].

This study employs the K-Nearest Neighbor (KNN) method for network attack detection due to its simplicity and ability to classify data based on instance proximity. In addition, the KNN method is selected for its superior performance relative to other classification methods, such as SVM [16]. However, KNN performance is strongly influenced by data distribution, the selection of *k*, and the distance metric used. In imbalanced IDS datasets, KNN tends to be biased toward the majority class, leading to suboptimal attack detection when data imbalance is not properly handled, and parameters are not properly optimized.

Therefore, the research gap in this study lies in the absence of an integrated approach that is simultaneously able to address multiclass data imbalance while optimizing KNN parameters, particularly the k value and distance metric selection, in the context of intrusion detection.

Based on this research gap, the novelty of this study lies in integrating the Synthetic Minority Over-sampling Technique (SMOTE) with parameter tuning in the KNN method, along with a comparative analysis of various k values and distance metrics to improve network attack detection performance. The objective of this study is to determine the optimal k value and distance metrics, including Euclidean, Manhattan, and cosine similarity, for SMOTE-balanced KNN in order to improve network attack detection accuracy.

2. Methods

The research workflow consists of systematic stages that begin with data collection and continue through the evaluation of classification model performance, as illustrated in Figure 1.

The first stage involves collecting IDS data from Kaggle [17], consisting of 28223 instances and 32 attributes, including 31 input attributes and 1 output attribute (the label). The labels are multiclass with six categories: Blackhole, Diversion, Normal, Overflow, PortScan, and TCP-SYN. The acquired dataset serves as the foundation for analysis and modeling in subsequent stages.

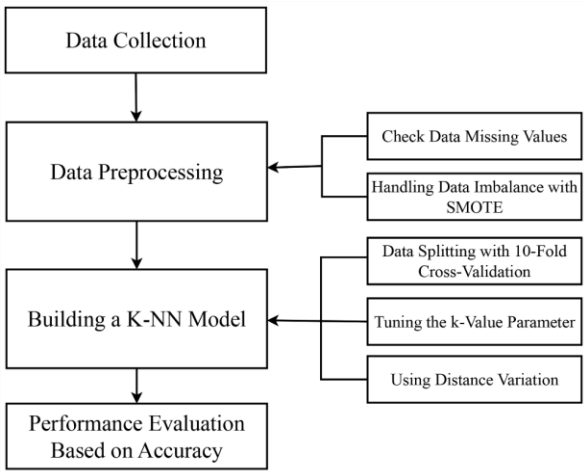


Figure 1. Research Flow

The second stage is data preprocessing, which aims to improve data quality prior to modeling [18]. At this stage, missing value checks are performed to ensure data completeness. Subsequently, the class imbalance is addressed using the SMOTE method to achieve a more balanced class distribution [7]. In the process, SMOTE generates synthetic samples from the minority class by linearly interpolating between the existing minority data and its nearest neighbors, thereby making the class distribution more balanced [19-21]. This process produces new samples located between two minority class instances in the feature space, resulting in a more balanced class distribution and reduced risk of overfitting.

The data are then divided using 5-fold cross-validation to ensure more stable training and testing processes and to reduce bias [22]. In addition, the *k* value in the KNN method is tuned, and different distance metrics, including Manhattan, Euclidean, and Cosine, are used to analyze their effects on model performance.

The third stage is the construction of the KNN model, in which the classification model is built from preprocessed data using predetermined parameters and distance measures. KNN performs classification by assigning the test data point's class based on the majority class among its k nearest neighbors in the feature space [23], [24]. Determining the value of k is an important factor in KNN because low values tend to be more sensitive to noise, while high values can cause underfitting [23].

The KNN process begins by determining the value of k as the number of nearest neighbors used for classification [25]. Next, the distance between the test data and the training data is calculated to select the k nearest neighbors, and then the class of the test data is determined by the majority class among the neighbors.

In KNN, the choice of distance metric can influence model accuracy and stability, particularly for data with

specific characteristics [26]. Commonly used distance metrics in KNN include Euclidean, Manhattan, and Cosine Similarity [27].

In the KNN algorithm, Euclidean distance is widely used to measure the direct distance between two points in a normalized numerical feature space. The Euclidean distance formula is presented in Equation 1 [28], with x as test data, y as training data, and n as the number of attributes.

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (1)$$

The Manhattan distance metric is more robust to outliers than the Euclidean distance because it does not involve squaring. Manhattan distance is often applied to high-dimensional data or data with non-uniform distributions [29]. The Manhattan distance formula is presented in Equation 2 [28].

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i| \quad (2)$$

The Cosine Similarity measure is widely used for high-dimensional, vector-based data, such as text. In KNN, a higher cosine similarity indicates greater similarity. The Cosine Similarity formula is shown in Equation 3 [28].

$$\cos(x,y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \quad (3)$$

The final stage is the evaluation of model performance, which is conducted based on accuracy to measure the model's correctness in performing classification. The confusion matrix is used to evaluate model performance by comparing predicted results with actual labels [30]. For binary classification, the confusion matrix is shown in Table 1. Accuracy, Precision, Recall, and F1-score is used to assess the overall correctness of the model in classifying data using Equations 4-7 [31], [32].

The experimental procedure in this study was designed to ensure valid and reproducible results. The dataset was first processed by handling missing values and balancing multiclass data using SMOTE. Next, a 10-fold cross-validation strategy was used, where the dataset was divided into 10 subsets: 9 for training and 1 for testing at each iteration. This process was repeated ten times, and the final results were obtained by averaging across all iterations. The KNN method was evaluated using various k values and several distance metrics to determine the optimal configuration. Details of the KNN parameter configuration used are presented in Table 2. Model performance was measured using the Accuracy, Precision, Recall, and F1-score metrics, with results reported as the average value across all folds.

Table 1. Confusion Matrix

Actual / Predicted	Positive	Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (4)$$

$$Recall = \frac{TN}{TN+FP} \quad (5)$$

$$Precion = \frac{TP}{TP+FP} \quad (6)$$

$$F1 - Score = \frac{2 * (Precision * Recall)}{Precision + Recall} \quad (7)$$

3. Results and Discussions

This study evaluates the performance of the KNN method for multiclass network attack detection by comparing the original dataset with a SMOTE-balanced dataset. The evaluation is conducted using a confusion matrix, with variations in *k* values (3, 5, and 7) and distance metrics, including Euclidean, Manhattan, and Cosine Similarity. Performance is analyzed by class-wise accuracy for each attack type: Blackhole, Diversion, Normal, Overflow, PortScan, and TCP-SYN.

The study employs an IDS dataset obtained from Kaggle [17]. The dataset consists of 28,223 instances and 32 attributes, including 31 input attributes and 1 output attribute (the class label). The labels are multiclass with six categories: Blackhole, Diversion, Normal, Overflow, PortScan, and TCP-SYN. Sample data from the dataset are presented in Table 2.

The IDS dataset exhibits a significant class imbalance, which can introduce bias into the model's learning process and degrade the performance of the KNN classification method. This imbalance negatively affects the model's ability to recognize minority classes, such as Overflow and PortScan, which have substantially fewer instances than other classes. The class distribution in the IDS dataset is highly uneven, with 8420 Blackhole instances, 5615 Diversion instances, 3773 Normal instances, 1022 Overflow instances, 312 PortScan instances, and 9081 TCP-SYN instances. Such an imbalance causes the KNN model to focus predominantly on the majority class, thereby increasing the risk of misclassifying minority classes.

Table 2. Network Intrusion Detection Dataset

No	Port Number	Received Packets	…	Max Size	Label
1	1	132	…	-1	TCP-SYN
2	2	187	…	-1	TCP-SYN
3	3	235		-1	TCP-SYN
4	4	59		-1	TCP-SYN
5	1	188		-1	TCP-SYN
…	…	…	…	…	…
28219	0	2	…	-1	PortScan
28220	0	42	…	-1	PortScan
28221	0	1	…	-1	PortScan
28222	0	2	…	-1	PortScan
28223	0	74	…	-1	PortScan

To address this issue, this study uses the SMOTE method for data balancing. SMOTE generates synthetic samples for minority classes based on the proximity between data points, thereby enhancing minority class representation without directly duplicating existing data. With a more balanced class distribution, the KNN method can perform classification more equitably and

optimally. The results of dataset balancing using SMOTE are illustrated in Figure 2.

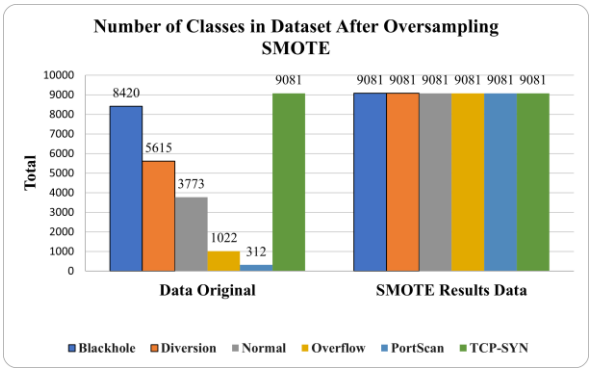


Figure 2. Data Distribution Results Before and After SMOTE

Based on the confusion matrix results and the evaluation metrics in Table 3, k = 3 provided optimal performance across all distance measurement methods: Euclidean, Manhattan, and Cosine Similarity. Increasing k to 5 and 7 tended to decrease recall for minority classes, such as PortScan and Overflow. This occurs because the higher the k value, the more the KNN model is influenced by the majority class, thereby reducing its sensitivity to classes with fewer samples.

With Euclidean Distance, classification performance was relatively stable across most classes, including Blackhole, Diversion, Normal, and TCP-SYN, with F1-scores ranging from 93% to 95% at k = 3. However, performance for the PortScan class remained low and declined as the k value increased. This indicates that Euclidean Distance has limitations in representing the characteristics of minority classes. Conversely, Manhattan Distance demonstrated superior ability to detect minority classes, with the highest PortScan recall of 32.26% at k = 3, while maintaining stable performance for other classes.

Meanwhile, Cosine Similarity shows high consistency in the Normal class, with 100% recall at all k values, and quite good performance in the Overflow class at k = 3. However, as with other methods, the recall value in the PortScan class remains low and decreases as k increases. Overall, the best configuration is obtained at k = 3. Manhattan is recommended for improving minority class detection, Euclidean is superior at maintaining the stability of the majority class, and Cosine is more suitable for data with vector-directional similarity. Therefore, the selection of the distance metric and k value should be adjusted to the classification objective, especially when dealing with imbalanced data.

In terms of distance metrics, Manhattan distance with *k* = 3 yields the best performance on the original data compared to Euclidean and Cosine distances. Furthermore, increasing *k* from 3 to 7 generally decreases accuracy. These findings indicate that the appropriate selection of distance metrics and *k* values can enhance the performance of the KNN method [33], [34].

Based on Table 4, which presents the confusion matrix results of the KNN method on data balanced using SMOTE, classification performance significantly improved across all classes compared to the pre-balanced data. In general, recall and F1-score values for almost all classes were above 90%, with the Normal class consistently achieving 100% recall across all distance variations and k values. This demonstrates that the SMOTE application effectively reduced class imbalance, enabling the KNN model to better recognize patterns in the minority class and minimize bias toward the majority class.

For Euclidean Distance, the best performance was achieved at k = 3, with high F1-scores across all classes, including PortScan (96.65%) and Overflow (95.41%). However, when k values increased to 5 and 7, recall decreased, particularly for the Blackhole and TCP-SYN classes. A similar pattern is also seen for cosine similarity, where k = 3 produces the most optimal performance. In contrast, increasing k values causes a gradual decrease in recall and F1-score, especially for the TCP-SYN class. This indicates that as k increases, the model tends to exhibit a smoothing effect that reduces sensitivity to local data variations.

Meanwhile, Manhattan Distance demonstrated relatively stable and superior performance compared to other distances, especially at k = 3. Recall and F1-score values for minority classes such as Overflow and PortScan reached above 96%, while maintaining consistent performance across other classes. Overall, these results indicate that the optimal combination of SMOTE and KNN is achieved at k = 3, with the Manhattan distance metric providing the best balance between sensitivity to minority classes and overall classification stability. Therefore, selecting k and the distance metric parameters is crucial for optimizing model performance on imbalanced data. These results indicate that the use of SMOTE oversampling can enhance the accuracy of the applied classification method [35], [36].

Table 5 shows that SMOTE consistently improves KNN performance across all combinations of distance metrics and *k* values. In the original dataset, the highest accuracy of 94.29% is achieved using the Manhattan distance with *k* = 3, whereas in the SMOTE-balanced dataset, the Manhattan distance also yields the highest accuracy of 96.51%. This improvement confirms that addressing data imbalance not only enhances minority class performance but also improves overall classification performance [37], [38]. The observed decline in accuracy when *k* is increased to 5 and 9 indicates that excessively large k values can reduce KNN's sensitivity to local patterns that are crucial for distinguishing attack types.

From the perspective of distance metrics, the Manhattan distance consistently provides more stable and superior performance compared to Euclidean and Cosine distances, both on the original dataset and the SMOTE-

balanced dataset [39], [40]. The high-dimensional characteristics of the dataset and the significant variability in attribute values are more appropriately measured using Manhattan distance, as this metric is more robust to extreme differences across features [41].

Table 3. Confusion Matrix Results of the KNN Method with Original Data

Distance	k	Actual / Prediction	Blackhole	Diversion	Normal	Over flow	Port Scan	TCP-SYN	Recall	Precision	F1-score
Euclidean	3	Blackhole	784	8	0	3	0	47	93.11%	95.61%	94.34%
		Diversion	12	536	0	5	0	8	95.54%	95.37%	95.46%
		Normal	0	0	378	0	0	0	100%	100%	100%
		Overflow	8	14	0	64	0	16	62.75%	85.33%	72.32%
		PortScan	2	0	0	0	6	23	19.35%	100%	32.43%
		TCP-SYN	14	4	0	3	0	887	97,69%	90.42%	93.91%
	5	Blackhole	771	9	0	7	1	54	91.57%	94.95%	93.23%
		Diversion	13	536	0	4	0	8	95.54%	94.04%	94.78%
		Normal	0	0	377	0	0	1	99.74%	100%	99.87%
		Overflow	8	18	0	56	0	20	54.90%	81.16%	65.50%
		PortScan	2	0	0	0	5	24	16.13%	62.50%	25.64%
		TCP-SYN	18	7	0	2	2	879	96.81%	89.15%	92.82%
	7	Blackhole	768	10	0	8	1	55	91.21%	94.46%	92.81%
		Diversion	15	533	0	5	0	8	95.01%	93.84%	94.42%
		Normal	0	0	377	0	0	1	99.74%	100%	99.87%
		Overflow	10	17	0	49	0	26	48.04%	73.13%	57.99%
		PortScan	0	0	0	0	4	27	12.90%	80.00%	22.22%
		TCP-SYN	20	8	0	5	0	875	96.37%	88.21%	92.11%
Manhattan	3	Blackhole	786	6	0	3	0	47	93.35%	95.50%	94.41%
		Diversion	9	540	0	5	0	7	96.26%	96.26%	96.26%
		Normal	0	0	378	0	0	0	100%	100%	100%
		Overflow	8	11	0	64	0	19	62.75%	84.21%	71.91%
		PortScan	3	0	0	0	10	18	32.26%	100%	48.78%
		TCP-SYN	17	4	0	4	0	883	97.25%	90.66%	93.84%
	5	Blackhole	776	10	0	4	1	51	92.16%	95.57%	93.83%
		Diversion	11	539	0	3	0	8	96.08%	95.06%	95.57%
		Normal	0	0	378	0	0	0	100%	100%	100%
		Overflow	7	16	0	60	0	19	58.82%	84.51%	69.36%
		PortScan	2	0	0	0	6	23	19.35%	60.00%	29.27%
		TCP-SYN	16	2	0	4	3	883	97.25%	89.74%	93.34%
	7	Blackhole	775	11	0	5	0	51	92.04%	94.98%	93.49%
		Diversion	15	531	0	6	0	9	94.65%	94.48%	94.57%
		Normal	0	0	378	0	0	0	100%	100%	100%
		Overflow	7	16	0	54	0	25	52.94%	78.26%	63.16%
		PortScan	1	0	0	0	3	27	9.68%	100%	17.65%
		TCP-SYN	18	4	0	4	0	882	97.14%	88.73%	92.74%
Cosine Similarity	3	Blackhole	782	11	0	3	0	46	92.87%	95.13%	93.99%
		Diversion	11	537	0	5	0	8	95.72%	95.04%	95.38%
		Normal	0	0	378	0	0	0	100%	100%	100%
		Overflow	6	15	0	66	0	15	64.71%	84.62%	73.33%
		PortScan	4	0	0	0	6	21	19.35%	100%	32.43%
		TCP-SYN	19	2	0	4	0	883	97.25%	90.75%	93.89%
	5	Blackhole	771	10	0	7	0	54	91.57%	95.42%	93.45%
		Diversion	10	539	0	4	0	8	96.08%	93.90%	94.98%
		Normal	0	0	378	0	0	0	100%	100%	100%
		Overflow	6	18	0	58	0	20	56.86%	82.86%	67.44%
		PortScan	2	0	0	0	5	24	16.13%	62.50%	25.64%
		TCP-SYN	19	7	0	1	3	878	96.70%	89.23%	92.81%
	7	Blackhole	767	13	0	8	0	54	91.09%	95.16%	93.08%
		Diversion	11	537	0	5	0	8	95.72%	93.55%	94.63%
		Normal	0	0	378	0	0	0	100%	100%	100%
		Overflow	8	17	0	53	0	24	51.96%	71.62%	60.23%
		PortScan	1	0	0	0	5	25	16.13%	100%	27.78%
		TCP-SYN	19	7	0	8	0	874	96.26%	88.73%	92.34%

In contrast, Euclidean and Cosine distances tend to be more sensitive to changes in *k* and data distribution, leading to greater declines in accuracy. Overall, the results of this study indicate that the combination of SMOTE with the Manhattan distance and a *k* value of 3 is the optimal configuration for improving KNN performance in multiclass network attack detection. These findings highlight the importance of addressing data imbalance and performing parameter tuning when developing Intrusion Detection Systems (IDS) using the KNN method. Furthermore, this study provides empirical evidence for selecting an effective and robust KNN configuration for network attack detection.

Table 4. Confusion Matrix Results of the KNN Method with SMOTE Data

Distance	k	Actual / Prediction	Blackhole	Diversion	Normal	Over flow	Port Scan	TCP-SYN	Recall	Precision	F1-score
Euclidean	3	Blackhole	839	14	0	15	8	32	92.40%	96.88%	94.59%
		Diversion	7	894	0	6	0	1	98.46%	97.28%	97.87%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	5	6	0	873	8	16	96.15%	94.69%	95.41%
		PortScan	1	0	0	1	894	12	98.46%	94.90%	96.65%
		TCP-SYN	14	5	0	27	32	830	91.41%	93.15%	92.27%
	5	Blackhole	829	15	0	18	11	35	91.30%	96.28%	93.73%
		Diversion	9	888	0	9	0	2	97.80%	96.84%	97.32%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	2	6	0	871	9	20	95.93%	93.16%	94.52%
		PortScan	1	0	0	1	892	14	98.24%	93.40%	95.76%
		TCP-SYN	20	8	0	36	43	801	88.22%	91.86%	90.00%
	7	Blackhole	815	19	0	24	15	35	89.76%	96.00%	92.77%
		Diversion	10	884	0	11	0	3	97.36%	96.19%	96.77%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	3	10	0	862	12	21	94.93%	92.09%	93.49%
		PortScan	0	0	0	1	892	15	98.24%	91.96%	94.99%
		TCP-SYN	21	6	0	38	51	792	87.22%	91.45%	89.29%
Manhattan	3	Blackhole	835	15	0	14	8	36	91.96%	96.98%	94.40%
		Diversion	7	896	0	4	0	1	98.68%	97.18%	97.92%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	2	6	0	878	8	14	96.70%	96.17%	96.43%
		PortScan	1	0	0	1	898	8	98.90%	95.33%	97.08%
		TCP-SYN	16	5	0	16	28	843	92.84%	93.46%	93.15%
	5	Blackhole	831	17	0	14	10	36	91.52%	97.08%	94.22%
		Diversion	6	895	0	6	0	1	98.57%	97.07%	97.81%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	2	5	0	879	8	14	96.81%	95.34%	96.07%
		PortScan	0	0	0	1	896	11	98.68%	94.02%	96.29%
		TCP-SYN	17	5	0	22	39	825	90.86%	93.01%	91.92%
	7	Blackhole	820	18	0	16	13	41	90.31%	97.04%	93.55%
		Diversion	7	891	0	8	0	2	98.13%	96.85%	97.48%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	2	7	0	872	10	17	96.04%	94.47%	95.25%
		PortScan	0	0	0	1	894	13	98.46%	92.36%	95.31%
		TCP-SYN	16	4	0	26	51	811	89.32%	91.74%	90.51%
Cosine Similiarity	3	Blackhole	835	15	0	14	9	35	91.96%	96.76%	94.30%
		Diversion	7	895	0	5	0	1	98.57%	97.07%	97.81%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	4	7	0	872	9	16	96.04%	95.20%	95.61%
		PortScan	1	0	0	1	896	10	98.68%	94.71%	96.66%
		TCP-SYN	16	5	0	24	32	831	91.52%	93.06%	92.28%
	5	Blackhole	828	15	0	18	14	33	91.19%	96.50%	93.77%
		Diversion	8	889	0	9	0	2	97.91%	96.42%	97.16%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	3	8	0	869	10	18	95.70%	93.14%	94.41%
		PortScan	1	0	0	1	893	13	98.35%	92.54%	95.36%
		TCP-SYN	18	10	0	36	48	796	87.67%	92.34%	89.94%
	7	Blackhole	814	20	0	23	16	35	89.65%	96.45%	92.92%
		Diversion	8	886	0	11	0	3	97.58%	95.89%	96.72%
		Normal	0	0	908	0	0	0	100%	100%	100%
		Overflow	3	12	0	863	12	18	95.04%	92.00%	93.50%
		PortScan	0	0	0	1	892	15	98.24%	91.68%	94.84%
		TCP-SYN	19	6	0	40	53	790	87.00%	91.75%	89.32%

Overall, the proposed method outperformed both the KNN model without SMOTE and the KNN model without parameter tuning, as evidenced by all evaluation metrics. However, in terms of computational time, the proposed method is slower than the model without SMOTE and the model using distance and cosine similarity metrics. The increase in computation time is due to the addition of minority data via SMOTE and the large number of features in the dataset. This condition is a limitation of this study. Therefore, further research can refine the approach by applying feature selection techniques to integrate only relevant features with SMOTE. This strategy is expected to improve computational time efficiency without compromising model stability and performance.

In the original dataset, high accuracy values accompanied by relatively low recall and F1-score indicate a model bias toward the majority class. This is due to the KNN mechanism, which relies on nearest-neighbor voting. In imbalanced data, the neighborhood of a minority instance tends to be dominated by majority samples. As a result, systematic misclassifications occur in the minority class. Despite high overall accuracy, the model cannot form a representative decision boundary across all classes.

Table 5. Performance Results of the KNN Method with Parameter Tuning

Distance	k	Original Data					Data With SMOTE				
		Accuracy	Recall	Precision	F1-score	Times (s)	Accuracy	Recall	Precision	F1-score	Times (s)
Euclidean	3	94.08%	78.07%	94.46%	85.49%	9	96.15%	96.15%	96.15%	96.15%	36
	5	92.98%	75.78%	86.97%	80.99%	9	95.25%	95.25%	95.26%	95.25%	37
	7	92.35%	73.88%	88.27%	80.44%	10	94.59%	94.59%	94.62%	94.60%	37
Manhattan	3	94.29%	80.31%	94.44%	86.80%	57	96.51%	96.51%	96.52%	96.52%	213
	5	93.62%	76.28%	90.66%	82.85%	57	96.07%	95.46%	95.48%	95.47%	211
	7	92.95%	74.41%	92.74%	82.57%	57	95.37%	95.37%	95.41%	95.39%	213
Cosine	3	93.98%	78.32%	94.26%	85.55%	24	96.13%	96.13%	96.13%	96.13%	90
Similarity	5	92.16%	76.22%	87.32%	81.39%	24	95.14%	95.14%	95.16%	95.15%	88
	7	92.63%	75.19%	91.51%	82.55%	24	94.59%	94.59%	94.63%	94.61%	88

The superior performance of Manhattan distance compared to Euclidean and Cosine Similarity can be explained by its mathematical characteristics. Manhattan distance calculates the absolute difference across each feature dimension, making it more robust to the influence of large variance and more stable in high-dimensional data. Conversely, Euclidean distance tends to magnify the influence of dominant features because it includes a squared component, which can distort the proximity structure among data points. Cosine Similarity emphasizes angular relationships and is less sensitive to magnitude differences, which are important for class separation. Therefore, the Manhattan distance better preserves relevant local structure, especially for sparsely distributed minority samples.

After applying SMOTE, recall and F1-score improved significantly due to a more balanced class distribution, which directly altered the neighborhood composition in KNN. SMOTE generates synthetic samples among minority instances, thereby increasing density in previously sparse areas and reducing the dominance of the majority class in the neighborhood determination process. This allows for the formation of a more stable and representative decision boundary, while reducing fragmentation through interpolation between minority samples. Furthermore, a smaller k value (k=3) proved optimal because it more accurately captured local structure, while larger k values tended to include neighbors from other classes, which could introduce noise. However, this performance improvement was accompanied by a significant increase in computational time due to the greater volume of oversampled data, indicating a trade-off between classification performance and computational efficiency, especially when using the Manhattan distance and large-scale datasets.

Table 6 compares the performance of the proposed method with several previous studies in the context of IDS attack detection, using accuracy and F1-score metrics. Various machine learning and deep learning approaches have been employed in prior research, including ensemble methods (XGBoost and Random Forest), deep learning models (CNN and LSTM), and unsupervised learning based on Autoencoders.

The reported accuracy results vary, ranging from 85.93% to 95%. Study [6] applied XGBoost and achieved an accuracy of 87%, while study [42] using Random Forest attained 90%. Study [43] improved performance to 92% through the use of CNN with data augmentation, highlighting the importance of handling data imbalance. The Autoencoder-based approach in [44] achieved an accuracy of 89%, whereas study [45] employing LSTM obtained an accuracy of 85.93%. Study [46] applied a deep learning approach and reported an accuracy of 95%. Study [47] roposed an AODV model combined with ABC, SVM, and ANN, achieving an accuracy of 94.85% and an F1-score of 94.60%. Study [48] used the Random Forest method and achieved an accuracy of 94.89% with an F1-score of 94.81%. Meanwhile, studi [49] proposed the Random Forest method combined with SMOTE and RFE feature selection, achieving an accuracy of 84.13% and an F1-score of 84.95%.

Table 6. Performance Comparison with Previous Research on Intrusion Detection

Research	Method	Accuracy	F1-score
Zibi et al. [6]	XGBoost	87.00%	-
Ajagbe et al. [42]	Random Foret	90.00%	-
Mohammad, et al. [43]	CNN with Augmentation Data	92.00%	-
Prabowo et al. [44]	Autoencoder	89.00%	-
Kasongo [45]	LSTM	85.93%	-
Lee et al. [46]	Deep Learning	95.00%	
Bharathi et al. [47]	AODV model with ABC, SVM, and ANN	94.85%	94.60%
Liu et al. [48]	Random Forest	94.89%	94.81%
Barkah et al. [49]	Random Forest with SMOTE and RFE	84.13%	84.95%
Lim [50]	CNN	73.38%	-
Proposed Research	SMOTE with KNN based on Manhattan Distance and k=3	96.51%	96.52%

Based on this comparison, the proposed method combining SMOTE with KNN using the Manhattan distance and a *k* value of 3 achieves the best performance, with an accuracy of 96.51% and F1-score of 96.52%. This improvement indicates that addressing data imbalance with SMOTE and selecting the Manhattan distance metric in KNN effectively enhances the model's ability to distinguish between attack patterns and normal traffic. Therefore, the proposed method is not only competitive but also outperforms most previous studies, including more complex deep learning approaches, offering a simpler yet effective solution for network attack detection.

The increase in computation time is due to the addition of minority data via SMOTE and the large number of features in the dataset. This condition is a limitation of this study. Therefore, further research can refine the approach by applying feature selection techniques to integrate only relevant features with SMOTE. This strategy is expected to improve computational time efficiency without compromising model stability and performance.

4. Conclusions

This study demonstrates that the problem of multiclass data imbalance in network attack detection can be effectively addressed by integrating SMOTE with parameter tuning in the KNN method. The novelty of this research lies in its systematic examination of the combined effects of *k*-value and distance-metric variations on KNN performance in imbalanced multiclass Intrusion Detection System (IDS) scenarios, which have received limited attention in previous studies. Experimental results show that $k = 3$ with the Manhattan distance on SMOTE-balanced data yields the best performance, achieving an accuracy of 96.51% and outperforming Euclidean and Cosine Similarity distances. These findings indicate that the Manhattan distance better captures the complex characteristics of network traffic after data balancing. Thus, this study not only improves overall classification performance but also makes a methodological contribution by providing an optimal parameter and distance selection strategy for KNN-based network intrusion detection systems.

However, in terms of computational time, the proposed method is slower than the model without SMOTE and the model using distance and cosine similarity metrics. The increase in computation time is due to the addition of minority data via SMOTE and the large number of features in the dataset. This condition is a limitation of this study. Therefore, further research can refine the approach by applying feature selection techniques to integrate only relevant features with SMOTE. This strategy is expected to improve computational time efficiency without compromising model stability and performance.

In addition, further research is recommended to develop and apply more adaptive data-balancing techniques, such as cluster-based SMOTE or hybrid sampling methods, to reduce class overlap and noise. In addition, performance evaluation can be extended by comparing the optimized KNN approach with other classification algorithms, including Support Vector Machine, Random Forest, Gradient Boosting, and ensemble methods in multiclass IDS scenarios.

References

[1] H. Fawareh and M. Saeed, "The Complex Dynamics of Network Security Threats that Effects Persons and Organizations Security," in *Artificial Intelligence for Sustainable Innovation Management and Risk Management: A Systems (and Network) Perspective*, A. Sarea, A. Echchabi, M. A. Salami, and A. Mahmood, Eds., Cham: Springer Nature Switzerland, 2026, pp. 2093–2113. doi: 10.1007/978-3-031-95310-1_151.

[2] L. Diana, P. Dini, and D. Paolini, "Overview on Intrusion Detection Systems for Computers Networking Security," *Computers*, vol. 14, no. 3, pp. 1–44, 2025, doi: 10.3390/computers14030087.

[3] S. ur Rehman, H. Alhulayyil, T. Alzahrani, H. AlSagri, M. U. Khalid, and V. Gruhn, "Intrusion detection system framework for cyber-physical systems," *Egypt. Informatics J.*, vol. 30, no. June, pp. 1–9, 2025, doi: https://doi.org/10.1016/j.eij.2024.100600.

[4] U. Ahmed *et al.*, "Signature-based intrusion detection using machine learning and deep learning approaches empowered with fuzzy clustering," *Sci. Rep.*, vol. 15, no. 1, pp. 1–33, 2025, doi: 10.1038/s41598-025-85866-7.

[5] Y. Zhang, R. C. Muniyandi, and F. Qamar, "A Review of Deep Learning Applications in Intrusion Detection Systems: Overcoming Challenges in Spatiotemporal Feature Extraction and Data Imbalance," *Appl. Sci.*, vol. 15, no. 3, pp. 1–20, 2025, doi: 10.3390/app15031552.

[6] V. Z. Mohale and I. C. Obagbuwa, "Evaluating machine learning-based intrusion detection systems with explainable AI: enhancing transparency and interpretability," *Front. Comput. Sci.*, vol. 7, no. May, pp. 1–23, 2025, doi: 10.3389/fcomp.2025.1520741.

[7] A. A. Khan, O. Chaudhari, and R. Chandra, "A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation," *Expert Syst. Appl.*, vol. 244, no. November 2023, p. 122778, 2024, doi: 10.1016/j.eswa.2023.122778.

[8] O. Alnasser, J. Al Muhtadi, K. Saleem, and S. Shrestha, "Signature and anomaly based intrusion detection system for secure IoTs and V2G communication," *Alexandria Eng. J.*, vol. 125, no. June, pp. 424–440, 2025, doi: https://doi.org/10.1016/j.aej.2025.03.068.

[9] M. A. Talukder *et al.*, "Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction," *J. Big Data*, vol. 11, no. 1, 2024, doi: 10.1186/s40537-024-00886-w.

[10] C. Lu, Y. Cao, and Z. Wang, "Research on Intrusion Detection Based on an Enhanced Random Forest Algorithm," *Appl. Sci.*, vol. 14, no. 2, 2024, doi: 10.3390/app14020714.

[11] S. Bagui and K. Li, "Resampling imbalanced data for network intrusion detection datasets," *J. Big Data*, vol. 8, no. 1, 2021, doi: 10.1186/s40537-020-00390-x.

[12] M. Sajid *et al.*, "Enhancing intrusion detection: a hybrid machine and deep learning approach," *J. Cloud Comput.*, vol. 13, no. 1, 2024, doi: 10.1186/s13677-024-00685-x.

[13] H. M. Khasanah, A. Aminuddin, F. F. Abdulloh, M. Rahardi, and H. Hairani, "Optimizing mushroom classification through machine learning and hyperparameter tuning," *Eng. Appl. Sci. Res.*, vol. 51, no. 5, pp. 651–660, 2024, doi: 10.14456/easr.2024.61.

[14] M. A. Saare, S. A. Lashari, S. A. Abbas, M. A. Al-Shareeda, M. A. Almaiah, and R. Shehab, "An Interpretable ANN-Based Intrusion Detection System Using SMOTE and Bayesian Optimization," *J. Robot. Control*, vol. 7, no. 1, pp. 3322–3332, Feb. 2026, doi: 10.18196/jrc.v7i1.26715.

[15] H. M. Dinh, W. Zong, and Y. Chow, "Investigating Oversampling Techniques to Mitigate Class Imbalance in Network Intrusion Detection Datasets," *Pragmatic Cybersecurity*, vol. 1, no. 1, pp. 1–24, 2026, doi: https://doi.org/10.53941/pc.2026.100004.

[16] O. P. Barus, F. Billie, Jusin, J. J. Pangaribuan, and A. Maulana, "Obesity Prediction: K-Nearest Neighbor vs. Support Vector Machine," in *2024 2nd International Conference on Technology Innovation and Its Applications (ICTIIA)*, 2024, pp. 1–5. doi: 10.1109/ICTIIA61827.2024.10761704.

[17] Network Intrusion Detection Dataset (UNR-IDD), "Kaggle," https://www.kaggle.com/datasets/mostafanofal/network-intrusion-detection-dataset-unr-idd/data.

[18] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Glob. Transitions Proc.*, vol. 3, no. 1, pp. 91–99, 2022, doi: 10.1016/j.gltp.2022.04.020.

[19] N. V Chawla, K. W. Bowyer, and L. O. Hall, "SMOTE :

Synthetic Minority Over-sampling TEchnique," *J. Artif. Intell. Res.*, vol. 16, pp. 341–378, 2002, doi: https://doi.org/10.1613/jair.953.
[20] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009, doi: 10.1109/TKDE.2008.239.
[21] T. Widiyaningtyas, H. Hairani, D. D. Prasetya, U. Pujianto, and W. Caesarendra, "A Modified SMOTE with Noise Filtering and Manhattan Distance Metric Approach to Address Imbalanced Health Datasets," *Eng. Technol. Appl. Sci. Res.*, vol. 15, no. 4, pp. 25452–25459, Aug. 2025, doi: 10.48084/etasr.11925.
[22] V. Teodorescu and L. Obreja Brașoveanu, "Assessing the Validity of k-Fold Cross-Validation for Model Selection: Evidence from Bankruptcy Prediction Using Random Forest and XGBoost," *Computation*, vol. 13, no. 5, pp. 1–22, 2025, doi: 10.3390/computation13050127.
[23] R. K. Halder, M. N. Uddin, M. A. Uddin, S. Aryal, and A. Khraisat, "Enhancing K-nearest neighbor algorithm: a comprehensive review and performance analysis of modifications," *J. Big Data*, vol. 11, no. 1, pp. 1–55, 2024, doi: 10.1186/s40537-024-00973-y.
[24] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967, doi: 10.1109/TIT.1967.1053964.
[25] A. İ. Çetin and A. H. Büyüklü, "A new approach to K-nearest neighbors distance metrics on sovereign country credit rating," *Kuwait J. Sci.*, vol. 52, no. 1, pp. 1–10, 2025, doi: 10.1016/j.kjs.2024.100324.
[26] R. Ehsani and F. Drabløs, "Robust Distance Measures for kNN Classification of Cancer Data," *Cancer Inform.*, vol. 19, pp. 1–9, 2020, doi: 10.1177/1176935120965542.
[27] R. Buaton and S. Solikhun, "The Application of Numerical Measure Variations in K-Means Clustering for Grouping Data," *MATRIK J. Manajemen, Tek. Inform. dan Rekayasa Komput.*, vol. 23, no. 1, pp. 103–112, 2023, doi: 10.30812/matrik.v23i1.3269.
[28] E. Blanco-Mallo, L. Morán-Fernández, B. Remeseiro, and V. Bolón-Canedo, "Do all roads lead to Rome? Studying distance measures in the context of machine learning," *Pattern Recognit.*, vol. 141, pp. 1–15, Sep. 2023, doi: 10.1016/j.patcog.2023.109646.
[29] I. H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions," *SN Comput. Sci.*, vol. 2, no. 3, p. 160, May 2021, doi: 10.1007/s42979-021-00592-x.
[30] P. Singh, N. Singh, K. K. Singh, and A. Singh, "Chapter 5 - Diagnosing of disease using machine learning," in *Machine Learning and the Internet of Medical Things in Healthcare*, K. K. Singh, M. Elhoseny, A. Singh, and A. A. Elngar, Eds., Academic Press, 2021, pp. 89–111. doi: https://doi.org/10.1016/B978-0-12-821229-5.00003-3.
[31] F. Saeik *et al.*, "Task offloading in Edge and Cloud Computing: A survey on mathematical, artificial intelligence and control theory solutions," *Comput. Networks*, vol. 195, p. 108177, 2021, doi: https://doi.org/10.1016/j.comnet.2021.108177.
[32] H. Hairani, T. Widiyaningtyas, D. D. Prasetya, and A. Aminuddin, "Addressing Imbalance in Health Datasets: A New Method NR-Clustering SMOTE and Distance Metric Modification," *Comput. Mater. Contin.*, vol. 82, no. 2, pp. 2931–2949, 2025, doi: 10.32604/cmc.2024.060837.
[33] S. Zhou, "A Method for Improving KNN Model Performance Based on Surrogate Model Feature Selection," in *Proceedings of the 2nd International Conference on Image Processing, Machine Learning, and Pattern Recognition*, in IPMLP '25. New York, NY, USA: Association for Computing Machinery, 2025, pp. 171–176. doi: 10.1145/3759928.3759956.
[34] B. Wan, Z. Sheng, W. Zhu, and Z. Hu, "Compactness-Weighted KNN Classification Algorithm," *Int. J. Adv. Comput. Sci. Appl.*, vol. 15, no. 9, pp. 229–238, 2024, doi: 10.14569/IJACSA.2024.0150922.
[35] H. Hairani, A. Anggrawan, and D. Priyanto, "Improvement Performance of the Random Forest Method on Unbalanced Diabetes Data Classification Using Smote-Tomek Link," *Int. J. Informatics Vis.*, vol. 7, no. 1, pp. 258–264, 2023, doi: https://dx.doi.org/10.30630/joiv.7.1.1069.
[36] D. Priyanto, H. Hairani, K. Marzuki, and M. Innuddin, "Optimization of Random Forest for Health Data Classification Using PCA and K-Means SMOTE-ENN," *Eng. Technol. Appl. Sci. Res.*, vol. 15, no. 5, pp. 27646–27652, 2025, doi: 10.48084/etasr.12976.
[37] M. M. Ahsan, M. S. Ali, and Z. Siddique, "Enhancing and improving the performance of imbalanced class data using novel GBO and SSG: A comparative analysis," *Neural Networks*, vol. 173, p. 106157, 2024, doi: https://doi.org/10.1016/j.neunet.2024.106157.
[38] W. Chen, K. Yang, Z. Yu, Y. Shi, and C. L. P. Chen, "A survey on imbalanced learning: latest research, applications and future directions," *Artif. Intell. Rev.*, vol. 57, no. 6, pp. 1–51, 2024, doi: 10.1007/s10462-024-10759-6.
[39] R. Suwanda, Z. Syahputra, and E. M. Zamzami, "Analysis of Euclidean Distance and Manhattan Distance in the K-Means Algorithm for Variations Number of Centroid K," *J. Phys. Conf. Ser.*, vol. 1566, no. 1, pp. 1–6, Jun. 2020, doi: 10.1088/1742-6596/1566/1/012058.
[40] J. A. Tolentino and B. D. Gerardo, "Enhanced Manhattan-based Clustering using Fuzzy C-Means Algorithm for High Dimensional Datasets," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 9, no. 3, pp. 766–771, May 2019, doi: 10.18517/ijaseit.9.3.6005.
[41] S. Chormunge and S. Jena, "Metric Based Performance Analysis of Clustering Algorithms for High Dimensional Data," in *2015 Fifth International Conference on Communication Systems and Network Technologies*, 2015, pp. 1060–1064. doi: 10.1109/CSNT.2015.127.
[42] S. A. Ajagbe, J. B. Awotunde, and H. Florez, "Intrusion Detection: A Comparison Study of Machine Learning Models Using Unbalanced Dataset," *SN Comput. Sci.*, vol. 5, no. 8, pp. 1–10, 2024, doi: 10.1007/s42979-024-03369-0.
[43] R. Mohammad, F. Saeed, A. A. Almazroi, F. S. Alsubaei, and A. A. Almazroi, "Enhancing Intrusion Detection Systems Using a Deep Learning and Data Augmentation Approach," *Systems*, vol. 12, no. 3, pp. 1–18, 2024, doi: 10.3390/systems12030079.
[44] A. O. Prabowo *et al.*, "Evaluation of Anomaly-based Network Intrusion Detection Systems with Unclean Training Data for Low-rate Attack Detection," *J. Cubersecurity Priv.*, vol. 6, no. 1, pp. 1–36, 2026, doi: https://doi.org/10.3390/jcp6010014.
[45] S. M. Kasongo, "A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework," *Comput. Commun.*, vol. 199, no. October 2022, pp. 113–125, 2023, doi: 10.1016/j.comcom.2022.12.010.
[46] G. H. Lee, J. Kim, Y. Yoo, and I. Lee, "Scientific Reports Article in Press Early detection of air leakage in IoT-connected compressors using enhanced data sampling with deep learning IN Early Detection of Air Leakage in IoT-Connected IN," *Sci. Rep.*, pp. 1–19, 2025, doi: https://doi.org/10.1038/s41598-025-29522-0.
[47] V. Bharathi, B. Poojitha, D. Ramesh, and P. Babu, "Intelligent Ad-Hoc Networking : Ensuring Confidential and Adaptive Data Transmission with AI," *Int. J. Commun. Networks Inf. Secur.*, vol. 17, no. 4, pp. 296–304, 2025. [Online]. Available: https://www.ijcnis.org/index.php/ijcnis/article/view/8429.
[48] L. Liu, P. Wang, J. Lin, and L. Liu, "Intrusion Detection of Imbalanced Network Traffic Based on Machine Learning and Deep Learning," *IEEE Access*, vol. 9, pp. 7550–7563, 2021, doi: 10.1109/ACCESS.2020.3048198.
[49] A. S. Barkah, S. R. Selamat, Z. Z. Abidin, and R. Wahyudi, "Impact of Data Balancing and Feature Selection on Machine Learning-based Network Intrusion Detection," *JOIV Int. J. Informatics Vis.*, vol. 7, no. 1, pp. 241–248, Feb. 2023, doi: 10.30630/joiv.7.1.1041.
[50] Y. S. Lim, "Empirical Evaluation of SNN for IoT Network Anomaly Detection," *J. Korean Inst. Commun. Inf. Sci.*, vol. 49, no. 11, pp. 1497–1509, 2024, doi: 10.7840/kics.2024.49.11.1497.