OPEN ACCESS
VOL. 7, NO.21, PP.173-182, MAY 2025

ISSN : 2622-8106 (ONLINE)
DOI: 10.20895/INISTA.V7I2.1834

Journal of Informatics, Information System, Software Engineering and Applications (INISTA)

# Classification of DDoS Attacks based on Network Traffic Patterns Using the k-Nearest Neighbor (k-NN) Algorithm

Muhammad Nur Faiz [*#1], Ratih Hafsarah Maharrani [#2], Laura Sari [#3], Arif Wirawan Muhammad [*4], Abdul Rohman Supriyono [#5]

*# Jurusan Komputer dan Bisnis, Politeknik Negeri Cilacap*
*Address Jalan Dr. Soetemo No.1, Cilacap, Central Java, Indonesia*

[1] faiz@pnc.ac.id
[2] ratih.hafsarah@pnc.ac.id
[3] laurasari@pnc.ac.id
[5] a.rohman.sy@pnc.ac.id

[*] *Informatics Engineering, Telkom University Purwokerto*
*Jl. DI Panjaitan No. 128, Purwokerto, Central Java, Indonesia*
[4] arifnm@telkomuniversity.ac.id

Received on 05-05-2025, revised on 20-05-2025, accepted on 24-05-2025

**Abstract**

Many server attacks disrupt industrial or business operations. Attacks that flood bandwidth with simultaneous requests can overwhelm a system, leading to significant downtime and financial losses. Additionally, breaches that compromise sensitive data can damage a company's reputation and erode customer trust. DDoS attacks, or Distributed Denial of Service attacks, are among the most common types of server attacks. DDoS has been proven to cause server downtime, and one effective way to mitigate this attack is to detect and classify it using a machine learning approach. The K-Nearest Neighbor (KNN) algorithm, a simple yet effective classification method based on similarity measures, is known for its high accuracy. The current research builds upon two stages: the feature extraction stage and the classification stage, with the ultimate goal of improving the accuracy of DDoS identification using the CICDDoS2019 dataset. Based on this premise, the detection accuracy can be improved by enhancing these two stages. At a value of k equal to 3, this study produces an accuracy of 99.73%.

**Keywords:** DDoS, Classification, k-NN, Dataset, CICDDoS2019



***Corresponding Author:***
*Muhammad Nur Faiz
Jurusan Komputer dan Bisnis, Politeknik Negeri Cilacap
Jalan Dr. Soetemo No.1, Cilacap, Central Java, Indonesia.
Email: faiz@pnc.ac.id

## I. INTRODUCTION

Distributed denial-of-service (DDoS) is a type of attack that has been around since the 1990s [1],[2]. In recent years, the number of network-based threats, including the volume and intensity of DDoS, has increased significantly. DDoS attacks generally do not exploit security vulnerabilities in a targeted network system; instead, DDoS attacks aim to disrupt services available in the target network by flooding the target's bandwidth or overloading the processing capacity of the target's server system [3]. On the other hand, the decrease in processing capacity of the network server system is not only caused by DDoS attacks but also by the presence of flash crowds. A flash crowd is not an attack, but rather a situation where there is a sudden and significant increase in network traffic, causing it to become inaccessible for a certain period of time [4].

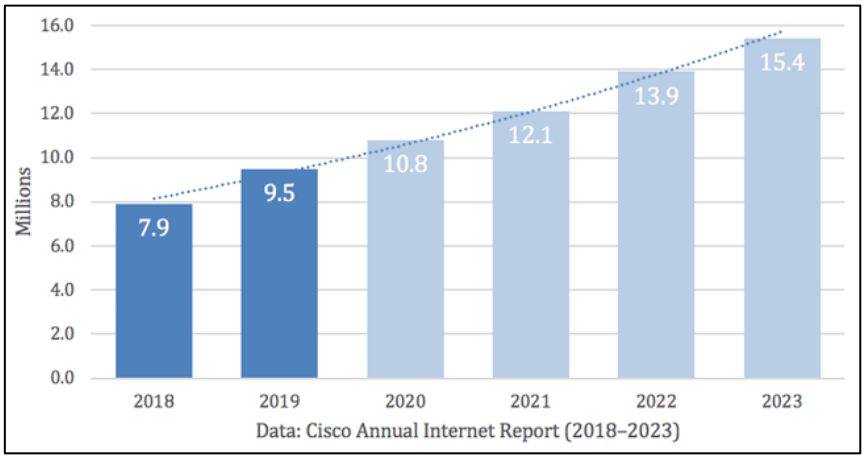



Fig. 1. DDoS Attack Prediction 2018-2023

Based on Figure 1, there is a significant upward trend from 2018 to 2023. This increase indicates that threats to cybersecurity, especially those that disrupt the target network server system by overloading, are becoming increasingly worrying from year to year [5]. DDoS attacks are currently still classified as one of the most popular types of attacks on cybersecurity issues [6]. The impact of this attack generally includes increased bandwidth usage, server resource consumption, disruption to data integrity and availability, and potential threats to the confidentiality of data stored on the server [7],[8]. Given the magnitude of the impact caused, early detection is a crucial key to minimizing the impact of this DDoS attack [9]. One of the primary efforts to minimize DDoS attacks is to implement an Intrusion Detection System (IDS) on the server, which monitors the flow of data packets entering the internal network or vice versa [10],[11]. Detection techniques in IDS continue to develop to deal with various modern techniques and tools attackers use. Currently, many IDSs rely on signature and anomaly-based detection models, which have proven effective in recognizing known attack patterns and deviant network behavior [12]. Although this technique can produce a high false-positive rate, it demonstrates the system's sensitivity in detecting various potential threats [13]. Technically, signature-based IDS and anomaly-based IDS work by monitoring the flow of data packets entering or leaving the internal network, providing a quick response to suspicious activity [14],[15]. When detecting activity that does not match the embedded signature database, the IDS will provide a notification or flag as an initial mitigation step [6],[16]. To improve the accuracy of predictions in identifying attacks, various machine learning-based approaches, such as the KNN algorithm, have also begun to be adopted.

The K-Nearest Neighbor (KNN) algorithm is a non-parametric machine learning method that offers several advantages, particularly due to its flexible parameter settings. Unlike parametric algorithms that rely on several assumptions and fixed parameters, KNN uses parameters that can be dynamically increased according to the complexity and amount of data available. This approach is very useful in situations where the data structure is not known with certainty or tends to change [17],[18]. The KNN algorithm is also lazy learning [19], which means it does not use training data to create a model. In short, the KNN algorithm has no training phase, although if one exists, it is minimal. All training data is used in the testing phase. This makes the training process faster and the testing phase slower, and tends to be 'expensive' in terms of time and memory. In the worst case, KNN takes more time to scan all data points [20]. This process will also require more memory to store training data. KNN is known as one of the simplest algorithms in machine learning for regression and classification tasks. This algorithm follows the principle of “birds of a feather flock together” in determining the class of new data [21], assuming that similar data will be nearby. Therefore, KNN classifies new data based on the similarity or distance to existing data and assigns the data to the most common class among its nearest neighbors [22].

The calculation of the distance between two points in the KNN algorithm uses the Euclidean Distance method, which can be applied to one-dimensional space (1-dimensional space), two-dimensional space (2-dimensional space), or multi-dimensional space (multi-dimensional space) [23]. One-dimensional space means that the distance calculation only uses one independent variable, two-dimensional space involves two independent variables, and multi-dimensional space involves more than two variables. Classification in the KNN algorithm is based on a distance function that measures the difference or similarity between two samples. The standard Euclidean distance $d(x, y)$ between two points can be calculated using the following formula, as shown in Equation (1).

$$\text{dis}\,(x_1, x_2) = \sqrt{\sum_{i=0}^{n}\left(X_{1_i} - X_{2_i}\right)} \quad (1)$$

Describe :
$dis$ : Euclidean distance between two points
$x_1$ : starting point
$x_2$ : end point
$i$ : feature index -i
$n$ : amount or dimension of data

The general architecture of the KNN algorithm is shown in Figure 2. The input layer consists of an input signal vector $x$. Calculations are performed between the input signal vector k and the neighbors $n$ in the hidden layer $y$. The output neuron then sums the linear outputs 0 of the neurons in the hidden layer. The results of the study [24] also showed that the KNN algorithm can provide a high detection rate and a low false positive rate. The approach using KNN in the study [25] also proved to be very effective in detecting DDoS attacks.

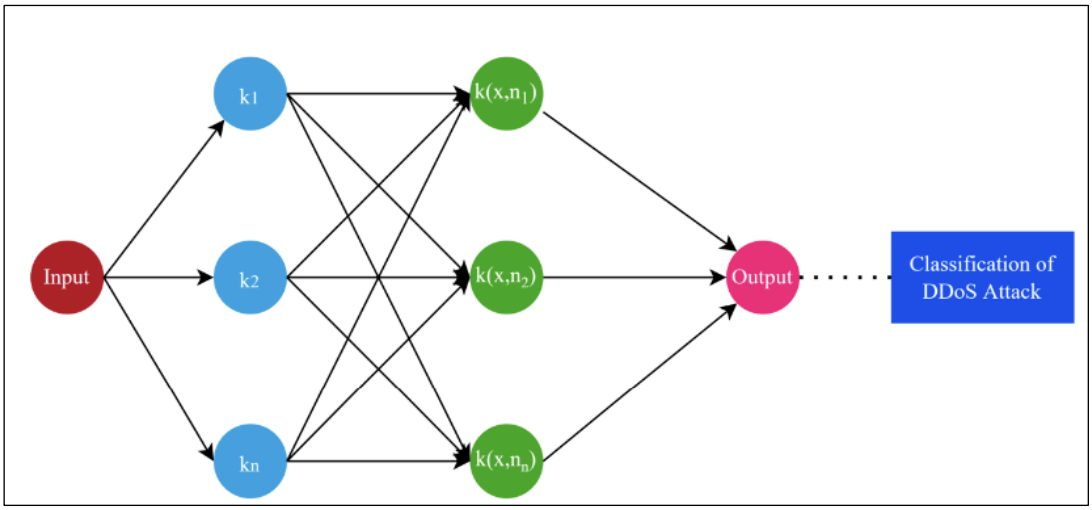



Fig. 2. Block diagram of the KNN model for DDoS attack classification.

Several previous studies on DDoS detection and classification have employed various algorithms aimed at improving the accuracy rate. Research [26] states that feature engineering focuses on obtaining datasets of various dimensions with significant features, utilizing feature selection methods such as recursive feature elimination, the chi-square ($\chi 2$) test, and the information gain score. In experiments conducted on DDoS datasets, the K-Nearest Neighbor algorithm showed the best overall performance, followed by SVM. Meanwhile, low-dimensional datasets with discrete-type features performed better under the Random Forest model compared to high-dimensional datasets with numeric features. The experiments demonstrated that a reduction of approximately 68% in feature space was achievable, with an impact of only about 0.03% on accuracy. Further research [27] developed a deep classification model using data streams to detect slow DoS attacks on HTTP. The classifier evaluated using the CICIDS2017 dataset showed an accuracy of 99.61%.

Additionally, research [28] proposed a new, explainable, and adaptable learning-based DDoS detection and classification method. The method first uses a modified KNN algorithm to detect DDoS attacks, then applies risk-level sorting through fine-grained traffic classification. This method generates a risk profile that provides interpretability for filtering DDoS traffic and does not require retraining the detection model when applied to a new network environment. Users can leverage various prior knowledge to develop the model. This study evaluated the approach in both simulated and real-world environments, demonstrating its effectiveness and efficiency with an accuracy rate of 98.4%.

Meanwhile, research [8] developed a DDoS detection model using the KNN algorithm to improve accuracy. This study will use the CICDDoS2019 dataset, which consists of 50,063,112 logs (where 50,006,249 rows are DDoS attacks and 56,863 rows are normal traffic). The dataset encompasses various DDoS attacks, including NTP, DNS, LDAP, and others. The dataset contains one class attribute and 88

features used to determine whether a packet is genuine or malicious. The application of the KNN algorithm to the data is expected to increase the accuracy in solving classification problems by utilizing a dataset consisting of DDoS attack data packets and normal data packet flows. Therefore, this study aims to develop a DDoS attack classification model based on the KNN algorithm, utilizing preprocessing techniques such as dimension reduction and feature selection to enhance the process of identifying and effectively handling DDoS attacks. The difference with research no. 28 is regarding DDoS detection, namely the use of its Dataset and different accuracy results, it is shown that in research 28 the dataset used CICIDS2017 and the accuracy results were 99.61% while in this study using the CICIDS2019 dataset is more recent because attacks continue to develop and the accuracy results are 99.7%. In addition, the use of features from this study is achieved through feature selection from 88 selected features, specifically 15 features that have a major influence on the accuracy results, thereby impacting the future success of DDoS detection.

## II. RESEARCH METHOD

### A. *Literature Study*

At this stage, tools and methods for the research are prepared based on the synthesis of the literature review results, particularly from previous studies.

### B. *Dataset*

At this stage, data collection is conducted that is relevant to the research being carried out. The dataset is obtained from online sources, namely datasets that have been published on the internet. The dataset obtained is then divided into two parts: one for training the machine learning model and the other for testing or validating the model. This data is divided into two parts in this algorithm, with 70% allocated for training and 30% for testing, because research on [29] error rates shows that these ratios yield lower error rates than the 80/20 and 60/40 ratios.

### C. *Data Preprocessing*

Before building the model, a data preprocessing stage was conducted, which included data cleaning and feature transformation. The data cleaning process aims to improve the quality and reliability of the dataset by removing duplicates, handling missing values, and eliminating outliers. Duplicate records were identified and removed to avoid bias in the classification results. Missing values were addressed using imputation techniques such as mean or mode replacement, depending on the attribute type [30]. Outliers were detected and treated using the interquartile range (IQR) method and z-score analysis to prevent distortion in distance calculations, which is particularly important for the k-NN algorithm [31]

### D. *Feature Selection*

At this stage, an optimal feature selection method is proposed to extract the most relevant attributes for anomaly detection. Feature selection techniques, such as Chi-Square and Information Gain, were employed to identify features that significantly contribute to classification performance. Additionally, a feature weighting strategy was applied to enhance the influence of features that exhibit a strong correlation with the target class during distance calculation in the k-NN algorithm. This approach allows more discriminative features to play a larger role in decision-making. A comparative evaluation was conducted between the weighted-feature k-NN and the baseline k-NN model to assess improvements in accuracy and recall. Although data normalization was not applied in this study, a feature analysis was performed to examine the contribution of individual attributes to the model's predictive performance. See Table I.

TABLE I. SELECTED FEATURES

| No | Feature | No | Feature | No | Feature |
|---|---|---|---|---|---|
| 1 | ct_srv_dst | 6 | dpkts | 11 | dwin |
| 2 | ct_srv_src | 7 | ct_src_ltm | 12 | swin |
| 3 | ct_dst_src_ltm | 8 | ct_dst_sport_ltm | 13 | stopb |
| 4 | ct_src_dport_ltm | 9 | dloss | 14 | dtopb |
| 5 | ct_dst_ltm | 10 | dbytes | 15 | spkts |

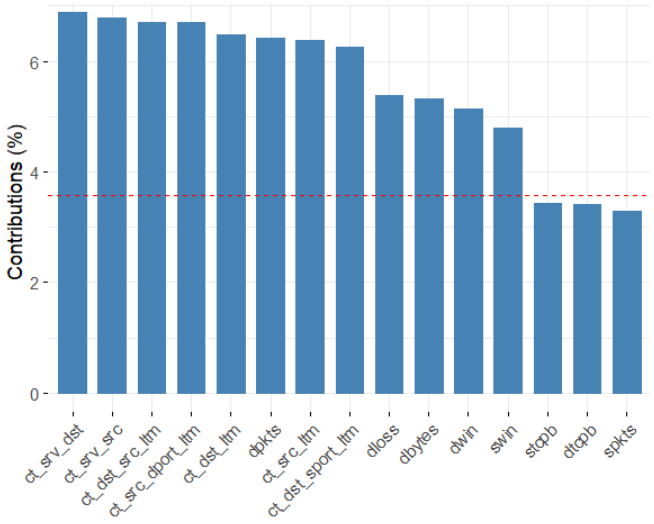



Fig. 3. Selected feature with the biggest impact.

## E. *KNN Algorithm*

This stage involves preparing the parameters that will be used to train and test the machine learning model. The parameters prepared include training time, the number of epochs used, the kernel used, and others. At this stage, training is also conducted on machine learning using previously prepared parameters. At this stage, testing is also conducted on the trained KNN model using the test data that has been prepared and extracted.

The KNN algorithm itself is the simplest supervised Machine Learning (ML) algorithm that utilizes the idea of "feature similarity" to predict the class of a particular data sample. This algorithm identifies sample points based on their neighboring points by calculating the distance between each pair of them. Standard distance metrics used include Euclidean distance, which directly affects classification outcomes. In the KNN algorithm, the parameter k has a significant impact on the model's performance. If the k value is very small, the model may experience over-fitting, as it becomes sensitive to noise in the training data. While choosing a very large k value can result in misclassification, as the algorithm may include too many distant neighbors that do not belong to the same class. Therefore, determining the optimal k is essential and is typically done through empirical testing or cross-validation.

Recent studies, such [30-32] have demonstrated that the KNN algorithm, when applied with proper parameter selection and feature optimization, can achieve robust performance in DDoS detection tasks. These findings highlight the practical relevance and flexibility of KNN in cybersecurity classification problems.

## F. *Confusion Matrix*

At this stage, the classification of normal traffic and attack traffic is performed, and the performance measurement of the detection model is conducted. Classification is carried out using the KNN algorithm. The evaluation of classification performance in Table II utilizes several metrics, including Accuracy, Recall, Precision, and F1-score. Choosing the right evaluation metric is crucial for objectively assessing model performance. Precision (P), Accuracy (A), recall (R), and F1-score (F1) are the main indicators used to calculate the precision value, which represents the proportion of correct positive predictions, with a value range of 0 to 1 [23],[33].

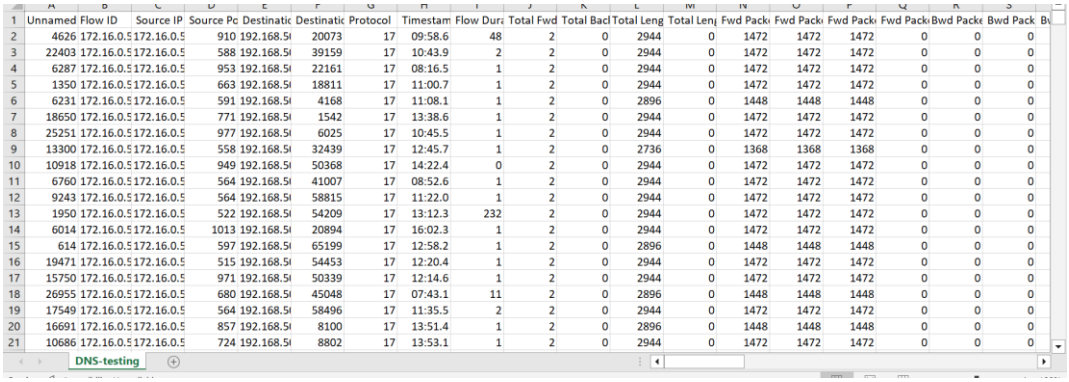
TABLE II. CONFUSION MATRIX

| Matrix | Describe |
|---|---|
| Accuracy | $\text{Accuracy} = [\frac{\text{TP}}{\text{TP} + \text{TN}}] \times 100$ |
| Precision | $\text{Precision} = [\frac{\text{TP}}{\text{TP} + \text{FP}}] \times 100$ |

| Matrix | Describe |
|---|---|
| Recall | $\text{Recall} = [\frac{\text{TP}}{\text{TP} + \text{FN}}] \times 100$ |
| F1-Score | $F1 = 2(\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}})$ |

## III. RESULTS AND DISCUSSION

The results of the KNN study using the CICDDoS2019 dataset with DNS filters are presented in Figure 3.



| Unnamed | Flow ID | Source IP | Source Pc | Destinatic | Destinatic | Protocol | Timestam | Flow Dura | Total Fwd | Total Bacl | Total Leng | Total Leng | Fwd Packe | Fwd Packe | Fwd Packe | Fwd Packe | Bwd Packe | Bwd Packe | Bv |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4626 | 172.16.0.5 | 172.16.0.5 | 910 | 192.168.5 | 20073 | 17 | 09:58.6 | 48 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 22403 | 172.16.0.5 | 172.16.0.5 | 588 | 192.168.5 | 39159 | 17 | 10:43.9 | 2 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 6287 | 172.16.0.5 | 172.16.0.5 | 953 | 192.168.5 | 22161 | 17 | 08:16.5 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 1350 | 172.16.0.5 | 172.16.0.5 | 663 | 192.168.5 | 18811 | 17 | 11:00.7 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 6231 | 172.16.0.5 | 172.16.0.5 | 591 | 192.168.5 | 4168 | 17 | 11:08.1 | 1 | 2 | 0 | 2896 | 0 | 1448 | 1448 | 1448 | 0 | 0 | 0 | |
| 18650 | 172.16.0.5 | 172.16.0.5 | 771 | 192.168.5 | 1542 | 17 | 13:38.6 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 25251 | 172.16.0.5 | 172.16.0.5 | 977 | 192.168.5 | 6025 | 17 | 10:45.5 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 13300 | 172.16.0.5 | 172.16.0.5 | 558 | 192.168.5 | 32439 | 17 | 12:45.7 | 1 | 2 | 0 | 2736 | 0 | 1368 | 1368 | 1368 | 0 | 0 | 0 | |
| 10918 | 172.16.0.5 | 172.16.0.5 | 949 | 192.168.5 | 50368 | 17 | 14:22.4 | 0 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 6760 | 172.16.0.5 | 172.16.0.5 | 564 | 192.168.5 | 41007 | 17 | 08:52.6 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 9243 | 172.16.0.5 | 172.16.0.5 | 564 | 192.168.5 | 58815 | 17 | 11:22.0 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 1950 | 172.16.0.5 | 172.16.0.5 | 522 | 192.168.5 | 54209 | 17 | 13:12.3 | 232 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 6014 | 172.16.0.5 | 172.16.0.5 | 1013 | 192.168.5 | 20894 | 17 | 16:02.3 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 614 | 172.16.0.5 | 172.16.0.5 | 597 | 192.168.5 | 65199 | 17 | 12:58.2 | 1 | 2 | 0 | 2896 | 0 | 1448 | 1448 | 1448 | 0 | 0 | 0 | |
| 19471 | 172.16.0.5 | 172.16.0.5 | 515 | 192.168.5 | 54453 | 17 | 12:20.4 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 15750 | 172.16.0.5 | 172.16.0.5 | 971 | 192.168.5 | 50339 | 17 | 12:14.6 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 26955 | 172.16.0.5 | 172.16.0.5 | 680 | 192.168.5 | 45048 | 17 | 07:43.1 | 11 | 2 | 0 | 2896 | 0 | 1448 | 1448 | 1448 | 0 | 0 | 0 | |
| 17549 | 172.16.0.5 | 172.16.0.5 | 564 | 192.168.5 | 58496 | 17 | 11:35.5 | 2 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |
| 16691 | 172.16.0.5 | 172.16.0.5 | 857 | 192.168.5 | 8100 | 17 | 13:51.4 | 1 | 2 | 0 | 2896 | 0 | 1448 | 1448 | 1448 | 0 | 0 | 0 | |
| 10686 | 172.16.0.5 | 172.16.0.5 | 724 | 192.168.5 | 8802 | 17 | 13:53.1 | 1 | 2 | 0 | 2944 | 0 | 1472 | 1472 | 1472 | 0 | 0 | 0 | |

DNS-testing

Fig. 4. CICDDoS2019 dataset with DNS filter

Figure 4 presents the dataset used in this study, formatted in CSV. The dataset comprises 88 features, which were later simplified by calculating a bias index per block based on selected features identified through the feature selection process. The dataset consists of 30,618 records labeled as DDoS traffic and 3,402 records representing normal traffic. For classification using the k-NN algorithm, the dataset was split into two subsets: 70% for training and 30% for testing. To reduce the potential for sampling bias, data partitioning was performed using a randomized splitting function.

In addition to the k-NN approach, experiments were also conducted using neural network models with different training algorithms. The architecture employed a sigmoid tangent (tansig) transfer function in the hidden layer and a pure linear (purelin) function in the output layer. The neural network was trained using two optimization algorithms available in MATLAB: the Quasi-Newton method (trainlm) and the Resilient Backpropagation method (trainrp). The training process utilized the following parameters: maximum epochs = 10,000, performance goal = 0.01, performance function = Mean Squared Error (MSE), maximum validation failures = 5, minimum gradient = 1.00e-10, and initial mu = 1.00e+10. The training outcomes for both methods are illustrated in Figure 5 and Figure 6, respectively.

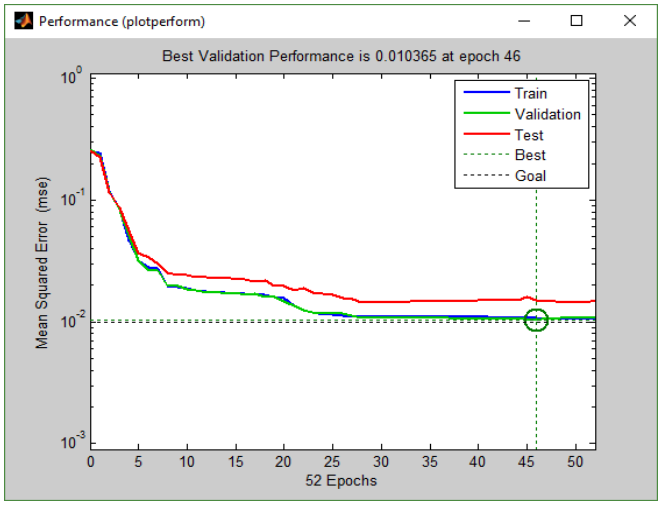



Fig. 5. CICDDoS2019 dataset with DNS filter

In Figure 5 (Quasi-Newton/trainlm), the MSE (Mean Squared Error) curve drops rapidly from nearly 1.0 in the early epochs to approximately 0.1 within the first few iterations. After about epoch 10, the rate of decrease slows but remains steady until reaching the Best Validation Performance of 0.010365 at epoch 46 (marked by the green circle). The training (blue), validation (green), and test (red) curves remain closely aligned without significant divergence after the best epoch, demonstrating that the model generalizes well and does not overfit. This behavior supports the high accuracy values observed (99.6 % on the full dataset and 99.4 % on the selected-features subset).

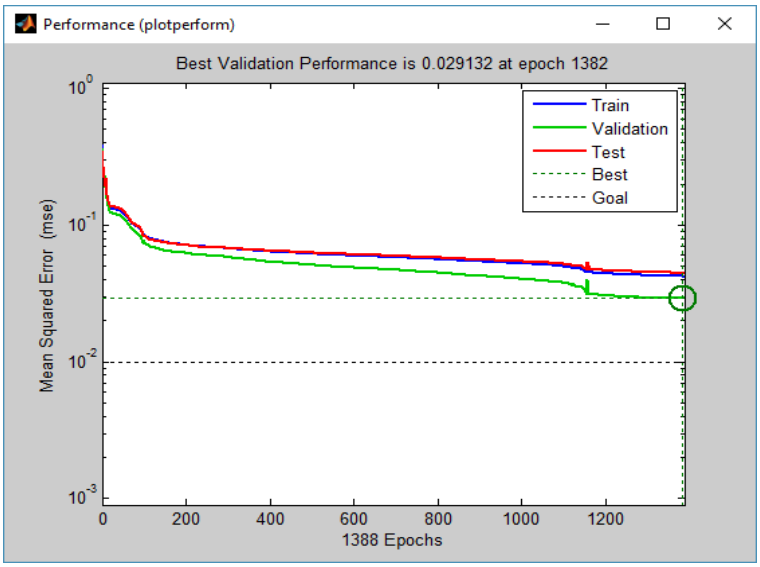



Fig. 6. CICDDoS2019 dataset with DNS filter

In Figure 6 (Resilient-Propagation/trainrp), the MSE also falls quickly from around 1.0 to 0.1 during the initial few hundred epochs. Still, the model requires 1,382 epochs to reach its Best Validation Performance of 0.029132. After this point, the validation and test curves remain stable with no spikes in MSE that would indicate overfitting. Although the final validation MSE (≈ 0.0291) is higher than that achieved by trainlm (≈ 0.0104), the trainrp method still yields slightly higher classification accuracy (99.8% and 99.7%). This indicates that, despite its slower convergence, the more conservative weight updates in trainrp effectively maintain stability and enhance classification performance.

All training results indicate that no overtraining was encountered in the k-NN scheme. The Quasi-Newton training function (trainlm in MATLAB) achieves the highest accuracy of 0.996 (99.6%) on the full CICDDoS2019 dataset and 0.994 (99.4%) on the dataset with selected features, as shown in Figures 5 and 6. Meanwhile, training with the Resilient-Propagation method (trainrp in MATLAB) yields an accuracy of 0.998 (99.8%) on the CICDDoS2019 dataset and 0.997 (99.7%) on the dataset with selected features, confirming that both methods can maximize classification performance without any signs of overfitting.

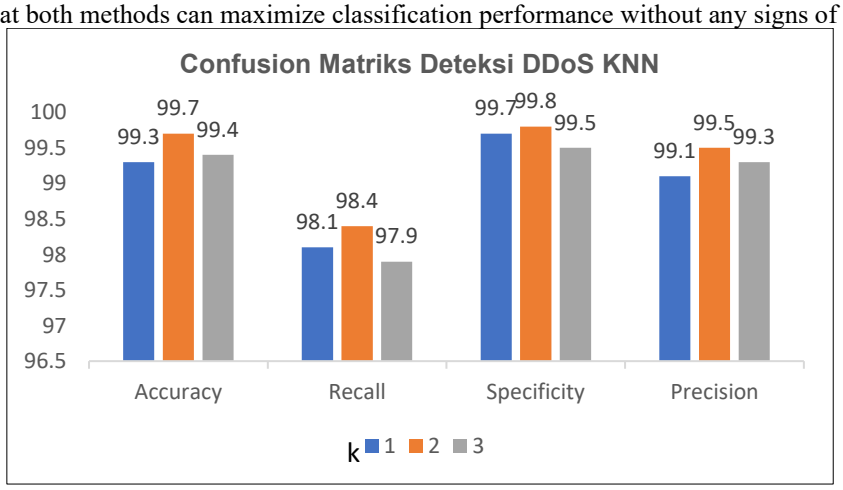



Fig. 7. Example of a Figure

Figure 7 summarizes the results of our KNN experiments using a subset of 15 carefully selected features. When comparing three different values of $k$ ($k = 1, 3, 5$), the configuration with $k = 3$ outperforms the others. Specifically, with $k = 3$, the model achieves an overall accuracy of 99.7 %, correctly classifying nearly all samples in both the DDoS and normal categories. Its recall (measuring the proportion of actual DDoS attacks detected) reaches 98.4%, meaning that very few attacks go unnoticed, while its specificity of 99.8% indicates almost no false alarms on benign traffic. In addition, the precision for $k = 3$ is 99.8 %, demonstrating that nearly every packet flagged as an attack truly is one. These values combine to yield an F1-score of 99.5%, reflecting an excellent balance between detecting attacks and avoiding false positives. For comparison, $k = 1$ yields a lower accuracy of 99.3 %, recall of 98.1 %, specificity of 99.7 %, and precision of 99.1 %, while $k = 5$ reaches 99.4 % accuracy, 97.9 % recall, 99.5 % specificity, and 99.3 % precision. Taken together, these results demonstrate that KNN with $k = 3$ provides a simple yet highly effective approach for classifying DDoS traffic, achieving a very high overall accuracy of 99.7 %.

## IV. CONCLUSION

The study demonstrates that a K-Nearest Neighbors (KNN) classifier using just 15 carefully selected features and $k = 3$ can detect DDoS attacks with outstanding performance (99.7 % accuracy, 99.8 % precision, 98.4 % recall, and a 99.5 % F1-score) outperforming $k = 1$ and $k = 5$. Neural-network models trained with Quasi-Newton (trainlm) and Resilient-Propagation (trainrp) also achieved over 99 % accuracy on both full and reduced-feature datasets without signs of overfitting. These results directly support the claim that feature reduction combined with an optimal $k$ value yields a lightweight yet highly accurate detection framework, as the confusion-matrix metrics and convergence curves corroborate all stated performance figures.

These findings align with previous research asserting that distance-based classifiers benefit enormously from effective feature selection, and they reaffirm that trainlm typically converges faster with lower validation error than trainrp. By showing that a simple KNN model (when tuned with an optimal $k$ and a minimal feature set) can match or exceed more complex architectures, this work advances the field by offering a transparent, resource-efficient baseline for real-time DDoS detection.

## V. ACKNOWLEDGMENT

The author would like to thank the Cilacap State Polytechnic for providing financial support for this research. This research was funded through the Internal DIPA Grant research activity for the 2023 Fiscal Year with contract 069/PL43/HK.07/2023.

## REFERENCES

[1] S. S. Ahmed, "A Study of Machine Learning Algorithms for DDoS Detection," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. VI, pp. 174–178, 2021, doi: 10.22214/ijraset.2021.34922.

[2] A. W. Muhammad, I. Riadi, and S. Sunardi, "Deteksi Serangan DDoS Menggunakan Neural Network dengan Fungsi Fixed Moving Average Window," *JISKA (Jurnal Inform. Sunan Kalijaga)*, vol. 1, no. 3, p. 115, 2017, doi: 10.14421/jiska.2017.13-03.

[3] M. Imthiyas, S. Wani, R. Abdulkhaleq, A. Abdulghafor, A. A. Ibrahim, and A. Hafeez, "DDoS Mitigation : A review of Content Delivery Network and its DDoS Defense techniques," *Int. J. Perceptive Cogn. Comput.*, vol. 6, no. 2, pp. 67–76, 2020.

[4] C. Kamtoso, A. Noertjahyana, and R. Intan, "Kombinasi Metode Partial Rank Correlation dan Flow Correlation Coefficient untuk Membedakan DDoS dengan Flash Crowds," *J. Infra*, vol. 9, no. 1, pp. 116–121, 2019.

[5] Cisco, "Cisco Annual Internet Report (2018–2023)," *Cisco Annual Internet Report*, 2020. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (accessed Jan. 20, 2023).

[6] Muhammad Nur Faiz, Oman Somantri, and Arif Wirawan Muhammad, "Machine Learning-Based Feature Engineering to Detect DDoS Attacks," *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 11, no. 3, pp. 176–182, Aug. 2022, doi: 10.22146/jnteti.v11i3.3423.

[7] M. Aziz, R. Umar, and F. Ridho, "Implemetasi Jaringan Saraf Tiruan untuk Mendeteksi Serangan DDoS pada Forensik Jaringan," *QUERY J. Sist. Inf.*, vol. 03, no. 1, pp. 3–9, 2019, doi: 10.58836/query.v3i1.4423.

[8] D. Kumar, R. K. Pateriya, R. K. Gupta, V. Dehalwar, and A. Sharma, "DDoS Detection using Deep Learning," *Procedia Comput. Sci.*, vol. 218, pp. 2420–2429, 2023, doi:

10.1016/j.procs.2023.01.217.
[9] T. Aytaç, M. A. Aydın, and A. H. Zaim, "Detection DDoS Attacks using Machine Learning Methods," *Electrica*, vol. 20, no. 2, pp. 159–167, 2020, doi: 10.5152/electrica.2020.20049.
[10] A. W. Muhammad, M. N. Faiz, and U. Athiyah, "Pengembangan Perangkat Lunak Untuk Deteksi DDoS Berbasis Neural Network," *Infotekmesin*, vol. 13, no. 02, pp. 301–307, 2022, doi: 10.35970/infotekmesin.v13i2.1396.
[11] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," *Proc. - Int. Carnahan Conf. Secur. Technol.*, vol. 2019-Octob, no. December, 2019, doi: 10.1109/CCST.2019.8888419.
[12] E. M. Bårli, A. Yazidi, E. H. Viedma, and H. Haugerud, "DoS and DDoS mitigation using Variational Autoencoders," *Comput. Networks*, vol. 199, no. June, p. 108399, 2021, doi: 10.1016/j.comnet.2021.108399.
[13] M. N. Faiz, O. Somantri, A. R. Supriyono, and A. W. Muhammad, "Impact of Feature Selection Methods on Machine Learning-based for Detecting DDoS Attacks : Literature Review," *J. Informatics Telecommun. Eng.*, vol. 5, no. 2, pp. 305–314, 2022, doi: 10.31289/jite.v5i2.6112.
[14] N. Bindra and M. Sood, "Evaluating the impact of feature selection methods on the performance of the machine learning models in detecting DDoS attacks," *Rom. J. Inf. Sci. Technol.*, vol. 23, no. 3, pp. 250–261, 2020.
[15] W. A. Prabowo, K. Fauziah, A. S. Nahrowi, M. N. Faiz, and A. W. Muhammad, "Strengthening Network Security: Evaluation of Intrusion Detection and Prevention Systems Tools in Networking Systems," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 9, pp. 1–10, 2023, doi: 10.14569/IJACSA.2023.0140934.
[16] A. Erfan, "DDoS Attack Detection Scheme using Hybrid Ensemble Learning And GA Algorithm for Internet of Things," *J. Archaeol. Egypt/Egyptology*, vol. 18, no. 18, pp. 521–546, 2021.
[17] F. Acito, "k Nearest Neighbors," in *Predictive Analytics with KNIME*, Cham: Springer Nature Switzerland, 2023, pp. 209–227. doi: 10.1007/978-3-031-45630-5_10.
[18] V. B and R. Gangula, "Exploring the Power and Practical Applications of K-Nearest Neighbours (KNN) in Machine Learning," *J. Comput. Allied Intell.*, vol. 2, no. 1, pp. 8–15, Feb. 2024, doi: 10.69996/jcai.2024002.
[19] G. Kaur, P. Gupta, and Y. Kumar, "Detection Mechanism Using Transductive Learning and Support Vectors for Software-Defined Networks," *Int. J. Inf. Retr. Res.*, vol. 12, no. 3, pp. 1–22, 2022, doi: 10.4018/ijirr.300293.
[20] S. S. Priya, M. Sivaram, D. Yuvaraj, and A. Jayanthiladevi, "Machine Learning based DDOS Detection," in *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*, Mar. 2020, pp. 234–237. doi: 10.1109/ESCI48226.2020.9167642.
[21] Mustakim; and G. Oktaviani, "Algoritma K-Nearest Neighbor Classification Sebagai Sistem Prediksi Predikat Prestasi Mahasiswa," *J. Sains, Teknol. dan Ind.*, vol. 13, no. 2, pp. 195–202, 2016.
[22] S. M. Kasongo and Y. Sun, "Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset," *J. Big Data*, vol. 7, no. 1, pp. 1–20, 2020, doi: 10.1186/s40537-020-00379-6.
[23] A. R. Chrismanto, Y. Lukito, and A. Susilo, "Implementasi Distance Weighted K-Nearest Neighbor Untuk Klasifikasi Spam & Non-Spam Pada Komentar Instagram," *J. Edukasi dan Penelit. Inform.*, vol. 6, no. 2, p. 236, 2020, doi: 10.26418/jp.v6i2.39996.
[24] Y. Liao and V. R. Vemuri, "Use of K-Nearest Neighbor classifier for intrusion detection," *Comput. Secur.*, vol. 21, no. 5, pp. 439–448, Oct. 2002, doi: 10.1016/S0167-4048(02)00514-X.
[25] R. M. A. Mohammad, M. K. Alsmadi, I. Almarashdeh, and M. Alzaqebah, "An improved rule induction based denial of service attacks classification model," *Comput. Secur.*, vol. 99, 2020, doi: 10.1016/j.cose.2020.102008.
[26] M. Aamir and S. M. A. Zaidi, "DDoS attack detection with feature engineering and machine learning: the framework and performance evaluation," *Int. J. Inf. Secur.*, vol. 18, no. 6, pp. 761–785, 2019, doi: 10.1007/s10207-019-00434-1.
[27] M. N. and J. B., "A deep learning based HTTP slow DoS classification approach using flow data," *ICT Express*, vol. 7, no. 2, pp. 210–214, Jun. 2021, doi: 10.1016/j.icte.2020.08.005.
[28] Y. Feng and J. Li, *Toward Explainable and Adaptable Detection and Classification of Distributed Denial-of-Service Attacks*, vol. 1271 CCIS, no. March. Springer International Publishing, 2020. doi: 10.1007/978-3-030-59621-7_6.
[29] A. Hakeem and A. Attiah, "Machine Learning-Based Approach for Detecting DDoS Attacks in Software Defined Networks," *Int. J. Comput. Appl.*, vol. 186, no. 43, pp. 1–9, Sep. 2024, doi:

10.5120/ijca2024924031.

[30] M. Tahir, A. Abdullah, N. I. Udzir, and K. A. Kasmiran, "A novel approach for handling missing data to enhance network intrusion detection system," *Cyber Secur. Appl.*, vol. 3, no. March 2024, 2025, doi: 10.1016/j.csa.2024.100063.

[31] A. G. Ayad, N. A. Sakr, and N. A. Hikal, "A hybrid approach for efficient feature selection in anomaly intrusion detection for IoT networks," *J. Supercomput.*, vol. 80, no. 19, pp. 26942–26984, Dec. 2024, doi: 10.1007/s11227-024-06409-x.

[32] Y. Zhou, H. Xia, D. Yu, J. Cheng, and J. Li, "Outlier detection method based on high-density iteration," *Inf. Sci. (Ny).*, vol. 662, p. 120286, Mar. 2024, doi: 10.1016/j.ins.2024.120286.

[33] K. Bouzoubaa, Y. Taher, and B. Nsiri, "Predicting DOS-DDOS Attacks: Review and Evaluation Study of Feature Selection Methods based on Wrapper Process," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 5, pp. 132–145, 2021, doi: 10.14569/IJACSA.2021.0120517.