

Flooding distributed denial of service detection in software-defined networking using k-means and naïve Bayes

Hicham Yzzogh, Hafssa Benaboud

Intelligent Processing and Security of Systems, Faculty of Sciences, Mohammed V University in Rabat, Rabat, Morocco

Article Info

Article history:

Received May 25, 2024

Revised Sep 6, 2024

Accepted Oct 1, 2024

Keywords:

Flooding distributed denial of service attacks

K-means

Naïve Bayes

SDN datasets

Software-defined networking

ABSTRACT

Software-defined networking (SDN) is a network architecture that enables the separation of the control plane and data plane, facilitating centralized management of the network. While centralized control offers numerous benefits, it also comes with certain drawbacks. Flooding distributed denial of service (DDoS) attacks pose a significant threat in SDN environments. These attacks involve overwhelming a target system with a large volume of packets, aiming to disrupt its functionality. In this paper, we propose a new approach for detecting DDoS attacks based on multiple k-means models and the naïve Bayes algorithm. Our methodology involves training multiple k-means models to cluster each data point within every column of the dataset, where each column represents a feature. This process results in a new dataset with the same shape, containing only clusters, except the column containing the target variable (labels). These clusters are then used as input by naïve Bayes to perform binary classification. We assessed our approach using the InSDN and CIC-DDoS2017 datasets. The results underscore the impressive accuracy of our model, achieving 99.9839% on the InSDN dataset and 99.7030% on the CIC-DDoS2017 dataset. This performance was achieved by optimizing the desired number of clusters.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Hicham Yzzogh

Intelligent Processing and Security of Systems, Faculty of Sciences, Mohammed V University in Rabat

Avenue Ibn Battouta B.P. 1014 RP, Rabat, Morocco

Email: Hicham_yzzogh@um5.ac.ma

1. INTRODUCTION

Software-defined networking (SDN) offers significant advantages for modern network infrastructure, including centralized control and programmability that enhance resource management and optimization. This centralized approach enables dynamic network configuration, simplifying management, boosting scalability, and increasing agility. Additionally, SDN facilitates automation and orchestration, reducing manual tasks such as provisioning and troubleshooting, which leads to improved deployment times and operational efficiency. Its programmable nature fosters innovation, supporting the development of novel network services and applications. Consequently, SDN's capabilities in centralized control, programmability, automation, and innovation establish it as a powerful paradigm that surpasses traditional network architectures in many aspects.

However, SDN is not immune to cybersecurity threats, with flooding distributed denial of service (DDoS) attacks being among the most critical challenges [1]. In SDN environments, the network architecture typically relies on OpenFlow switches for packet forwarding and management [2]. Attackers exploit vulnerabilities in this architecture to launch flooding DDoS attacks, overwhelming the system with illegitimate traffic [3]. These attacks target the network's reliance on OpenFlow switches and controllers by sending a high volume of packets that do not match existing flow table entries. As a result, these packets are

encapsulated into packet-in messages and forwarded to the controller. This excessive traffic can overwhelm the controller's bandwidth, memory, and central processing unit (CPU) resources, causing it to become unresponsive and disrupting network operations for legitimate users. Additionally, switches can experience memory exhaustion from the malicious traffic, impairing their ability to process legitimate traffic effectively.

Machine learning (ML) techniques offer a promising solution for enhancing DDoS defenses in SDN environments. Supervised learning methods, such as support vector machines (SVM), naïve Bayes (NB), and neural networks (NNs), leverage labeled datasets to distinguish between normal network behavior and DDoS patterns. These methods analyze features like packet headers and traffic volume to detect deviations from expected norms. Unsupervised learning approaches, such as clustering algorithms, can identify anomalies without pre-labeled data by analyzing deviations in data structures, thereby adapting to evolving attack strategies. Ensemble learning [4] further improves detection accuracy by combining multiple models, enhancing the effectiveness of ML-based DDoS detection systems.

In this paper, we introduce a novel approach that integrates k-means clustering with naïve Bayes classification for DDoS attack detection in SDN environments. By applying clustering at each feature level prior to classification, our method enhances detection accuracy. We first utilize multiple k-means models to cluster network traffic, followed by classification using naïve Bayes. Experimental results with the InSDN dataset [5] demonstrate that our approach outperforms both the naïve Bayes model and existing methods. Further evaluation using the CICIDS2017 dataset [6], [7] confirms its robustness and potential applicability beyond SDN environments.

While various methods have been proposed for DDoS detection in SDN environments, none utilize clustering at each feature level before classification. For instance, the network detection and prevention agent (NDPA) algorithm [8] dynamically regulates traffic flow to mitigate DDoS attacks while maintaining quality of service (QoS) standards. Arivudainambi *et al.* [9] combine convolutional neural networks (CNNs) with the lion optimization algorithm (LOA), achieving 98.2% accuracy on the NSL-KDD dataset [10]. Another method in [11] introduces a two-tier security framework with a C4.5 decision tree to enhance early DDoS attack detection. Additionally, the proposed method in [12] achieved 95.24% accuracy using SVM by extracting six-tuple characteristic values from switch flow tables. The one proposed in [13] explores an automated DDoS attack detection system integrating P4 programmable capabilities with various machine learning algorithms, including k-nearest neighbors (K-NN), random forest (RF), SVM, and artificial neural networks (ANNs). The study [14] investigates the effectiveness of different machine learning classifiers for transmission control protocol-synchronized (TCPSYN) flood DDoS attacks on SDN controllers, finding all classifiers to be highly effective. Similarly, the study [15] evaluates J48, RF, SVM, and K-NN for internet control message protocol (ICMP) and TCP flood detection, with J48 outperforming the other algorithms. Moreover, Kasim [16] introduces a hybrid deep learning model combining CNN and long short-term memory (LSTM) cells for DNS flood attacks, achieving 99.87% accuracy on the CICIDS2017 dataset. Finally, the study [17] proposes a preemptive security model using logistic regression, decision tree (DT), and K-NN algorithms to detect DDoS attacks before they occur, enhancing detection performance. Our approach incorporates feature-level clustering, enabling a more detailed and accurate analysis of network traffic. This method provides a more effective solution for managing DDoS threats in SDN environments and enhances detection accuracy.

The paper is structured as follows. In section 2, we present our approach to detecting flooding DDoS attacks and describe the experimental datasets. The results of the proposed model and their discussion are presented in section 3. Finally, we conclude the paper and discuss future work in section 4.

2. EXPERIMENTS AND METHOD

In this section, we present a detailed overview of our experimental setup, which includes descriptions of the datasets used, the features selected, and our proposed approach. Furthermore, we outline the evaluation process and metrics used to assess the model's performance. The experiments were conducted on a machine equipped with an Intel® Core™ i7-6820HQ CPU running at 2.70 GHz and 32 GB of RAM, using the Windows 10 operating system.

2.1. Experiment datasets

We utilized the InSDN dataset, specifically tailored for SDN attack analysis, to evaluate our model. This dataset stands out as one of the earliest and most comprehensive collections of attack scenarios devised for SDN environments. Employing an SDN-specific dataset is critical for accurately assessing SDN attack detection methods, as generic datasets may not fully capture the distinct architecture and attack vectors inherent to SDN networks. The InSDN dataset encompasses various flooding DDoS attacks, such as TCP-SYN flood, user datagram protocol (UDP) flood, and internet control message protocol (ICMP) flood

attacks, executed using the Hping3 tool, widely recognized as one of the most prevalent tools for conducting DDoS attacks. To form our experimental InSDN dataset, we combined the normal and ovarian vein syndrome (OVS) datasets in CSV format, resulting in a comprehensive collection of network activities. Our experimental InSDN dataset contains seven classes: normal, brute force attacks (BFAs), botnet attacks, denial-of-service (DoS), DDoS, probes, and web attacks. Our objective is to classify the traffic as either DDoS attack or not, so we labeled all classes other than DDoS as ‘Others’. As a result, our experimental dataset will contain only two classes: others and DDoS with the distribution of instances depicted in Figure 1. Furthermore, we excluded ‘Src IP’, ‘Src Port’, ‘Dst IP’, and ‘Dst Port’ to mitigate the risk of overfitting the model, conducting our experimentation using the dataset identified as InSDN_DDoS_Exp [18].

To further evaluate our model, we utilized the CIC-DDoS2017 dataset. This dataset contains “Friday – WorkingHours – Afternoon – DDoS.pcap_ISCX.csv”, a CSV file that includes attacks carried out using the low orbit ion cannon (LOIC) tool. This tool enables attackers to flood target servers or networks with high volumes of traffic, rendering them inaccessible to legitimate users. The file “Friday – WorkingHours – Afternoon – DDoS.pcap_ISCX.csv” contains a significantly higher number of instances of flooding DDoS attacks compared to BENIGN instances. In real-world scenarios, this distribution may not accurately represent the prevalence of DDoS attacks. Therefore, we removed infinite values from this CSV file, and created our experimental CIC-DDoS2017 dataset, denoted as *CIC-DDoS2017_Exp* [19], with the distribution depicted in Figure 1.

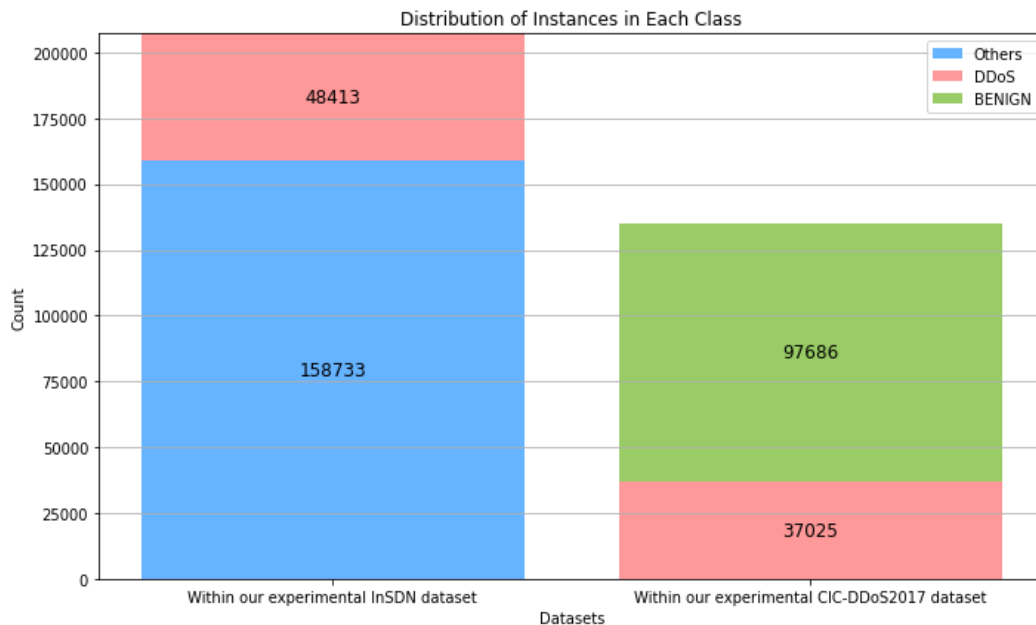


Figure 1. Class instance distribution

2.2. Features employed in our experiments

This section presents the features utilized in our experiments and details the feature selection process. We employed the SelectKBest method from the scikit-learn library to identify the most relevant features from our dataset. This method iterates over a range of ‘*k*’ values, where ‘*k*’ represents the number of features to be selected. For each ‘*k*’ value, the method computes the chi-squared (chi2) statistic between each feature and the target variable, selecting the top ‘*k*’ features with the highest chi2 scores. To determine the optimal ‘*k*’ value, we used a cross-validation strategy with a RF classifier. The feature subset that achieved the highest mean cross-validation score across different ‘*k*’ values was selected as the final set of features. This systematic approach ensures that the chosen features significantly enhance the classifier’s predictive performance, thereby improving overall model accuracy.

Table 1 shows the selected subset of features from our experimental InSDN and CIC-DDoS2017 datasets. The features are listed according to their importance, using the SelectKBest method, which identifies the top features based on a specified scoring function. This allows us to focus on the most relevant features. Notably, in our analysis of the CIC-DDoS2017 dataset, we excluded the ‘destination port’ feature to avoid overfitting and to prioritize features directly relevant to identifying DDoS attacks.

Table 1. Extracted subset features from our experimental datasets

No	Feature	No.	Feature
InSDN dataset			
1	Protocol	7	SYN flag Cnt
2	Flow duration	8	PSH flag Cnt
3	Flow Pkts/s	9	ACK flag Cnt
4	Bwd PSH flags	10	Down/up tatio
5	Bwd Pkts/s	11	Init Bwd win byts
6	FIN flag Cnt		
CIC-DDoS2017 dataset			
1	Fwd packet length max	15	Max packet length
2	Fwd packet length min	16	Packet length mean
3	Fwd packet length mean	17	Packet length std
4	Fwd packet length Std	18	Packet length variance
5	Bwd packet length max	19	SYN flag count
6	Bwd packet length min	20	PSH flag count
7	Bwd packet length mean	21	URG flag count
8	Bwd packet length std	22	Down/up ratio
9	Bwd IAT total	23	Average packet size
10	Bwd IAT mean	24	Avg Fwd segment size
11	Bwd IAT Std	25	Avg Bwd segment size
12	Bwd IAT max	26	Subflow Fwd bytes
13	Fwd PSH flags	27	Init_Win_bytes_backward
14	Min packet length		

2.3. Proposed approach

As shown in Figure 2, this paper introduces a DDoS detection technique that combines multiple k-means models with a naïve Bayes model. The proposed methodology involves the following general phases:

- a. Training multiple k-means models and a naïve Bayes model: We iteratively train k-means models using the training dataset, varying the desired number of clusters (k) within a predefined range. Each iteration corresponds to a different k value, allowing exploration of various clustering configurations. For each k value, a separate k-means model is trained for each feature in the dataset, enabling feature-specific clustering. Unlike clustering the entire dataset collectively, this approach operates independently on each column. For each feature, the k-means algorithm organizes the data into a maximum of k cohesive clusters, discerning patterns and groups within each column separately. After training the k-means models, each model is employed to assign a cluster to every data point within its respective column. The assigned clusters for the training set are utilized as input to train a Gaussian naïve Bayes model for binary classification.

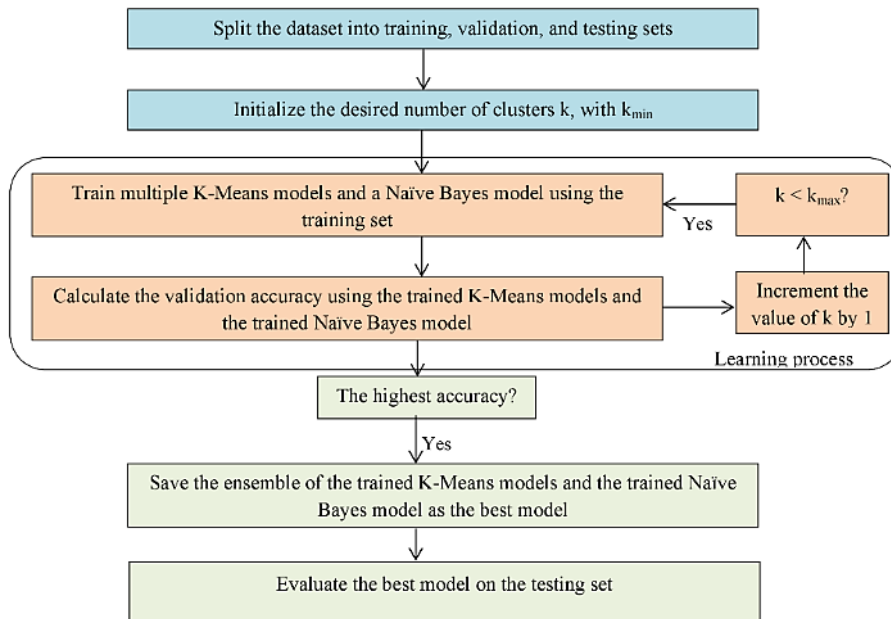


Figure 2. Flow diagram of the proposed method

- b. Select the best model: During the iteration process, the validation set is utilized to evaluate the performance of each model. Each model consists of multiple trained k-means models and a trained naïve Bayes model. This evaluation involves computing the validation accuracy based on the clustering results obtained from the trained k-means models and the subsequent classification using the trained naïve Bayes model. The model with the highest validation accuracy is selected as the best-performing model.
- c. Model evaluation: The performance of the best selected model is assessed using the testing set. Initially, the trained k-means models cluster the testing set. Subsequently, the assigned clusters are used as input for the trained naïve Bayes model to perform binary classification.

Algorithm 1 outlines the key steps of our proposed approach, focusing on how k-means clustering and the naïve Bayes model are integrated into the methodology. By demonstrating the collaboration between k-means and the naïve Bayes classifier, we emphasize their crucial roles in improving the accuracy and effectiveness of our method. This clear representation enables readers to understand the significant contributions of both k-means and naïve Bayes to the overall success of our approach.

Algorithm 1. Proposed DDoS detection approach

```

Input: Training dataset, validation dataset and testing dataset
Output: Best-performing DDoS detection model
Initialize best_KMeans_models;
Initialize best_NB_model;
Initialize best_accuracy=0;
Define X as the matrix of feature values and y as the vector of target values;
Step 1: Train k-means models KMeans_modelsk and naïve Bayes model NB_modelk
for each k in a predefined range do
    for each column in Xtrain do
        Train k-means model KMeans_modelk,column on Xtrain[column];
        Append KMeans_modelk,column to KMeans_modelsk;
    end
    Assign clusters to Xtrain using KMeans_modelsk;
    Train naïve Bayes model NB_modelk using cluster assignments;
Step 2: Evaluate on validation set;
Evaluate NB_modelk on Xval;
Compute validation accuracy accuracyk;
if accuracyk>best_accuracy then
    Update best accuracy to accuracyk;
    Update best_KMeans_models to KMeans_modelsk;
    Update best_NB model to NB_modelk;
end
end
Step 3: Evaluate the best model on testing set;
Apply best_KMeans_models to Xtest for clustering;
Apply best_NB model for classification on clustered data;

```

2.4. Evaluation process and metrics

The evaluation of our approach involves fine-tuning the number of clusters to identify the optimal model. We start by comparing our model's performance to that of a naïve Bayes model using the InSDN dataset, and subsequently, we assess its performance on the CIC-DDoS2017 dataset. We employed recognized performance metrics such as accuracy, precision, recall, and F1-score to assess the effectiveness of our model. These metrics are computed as (1)-(4):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4)$$

True positives (TP) and true negatives (TN) represent instances that the model has correctly classified. In contrast, false positives (FP) and false negatives (FN) refer to instances that the model has incorrectly classified. These metrics are crucial for evaluating the effectiveness of classification models, providing insights into their accuracy and reliability.

3. RESULTS AND DISCUSSION

In this section, we present the results obtained from the experiments conducted using our proposed approach. First, we compare the accuracy achieved by the NB model and our model with different k values. Next, we assess the performance of our model that attained the highest accuracy by examining the classification report in detail. The results of our experiments are illustrated in Figures 3 to 6.

3.1. Result on InSDN dataset

As depicted in Figure 3, both models demonstrate excellent accuracy. Our model aligns closely with naïve Bayes, achieving an impressive 99.9742% accuracy across various values of k . Notably, with 32 clusters, our model reaches a slightly higher accuracy of 99.9839%, representing a modest improvement of 0.0097% over naïve Bayes. This improvement, though small, highlights that our model performs marginally better while maintaining high accuracy. Additionally, Figure 4 shows that with $k = 32$, our model achieves robust precision, recall, and F1 score values of approximately 99.9455%, 99.9863%, and 99.9659%, respectively, underscoring its effectiveness in accurately identifying positive instances and minimizing misclassifications.

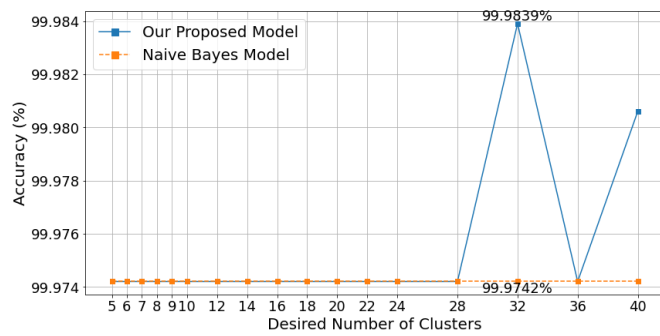


Figure 3. Accuracy across desired number of clusters within our experimental InSDN dataset

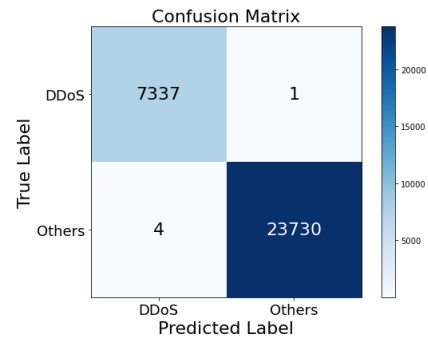


Figure 4. Confusion matrix within our experimental InSDN dataset

3.2. Result on CIC-DDoS2017 dataset

As shown in Figure 5, the proposed model initially has lower accuracy compared to the naïve Bayes model. However, as the number of clusters increases, the accuracy of our model gradually improves. By around $k = 24$, our model surpasses the naïve Bayes model and continues to demonstrate superior performance. At $k = 86$, our model achieves an accuracy of 99.7030%, which significantly exceeds the naïve Bayes model’s accuracy of 96.3774%, representing an increase of 3.3256%. This highlights the enhanced predictive capability of our model. Additionally, Figure 6 shows that with $k = 86$, our model achieves a precision of 99.2487%, indicating that 99.2487% of predicted positives are correctly classified, and a recall of 99.5693%, reflecting its ability to identify 99.5693% of actual positives. Consequently, the F1-score, which balances precision and recall, reaches 99.4088%.

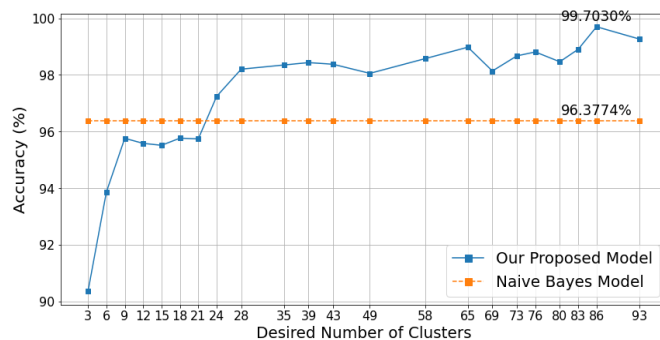


Figure 5. Accuracy across desired number of clusters within our experimental CIC-DDoS2017 dataset

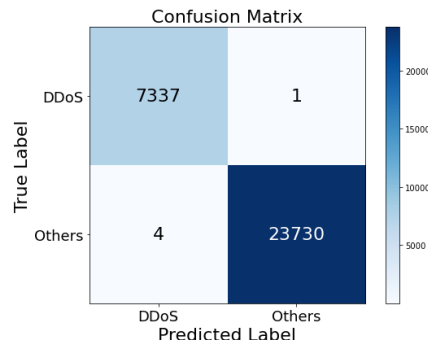


Figure 6. Confusion matrix within our experimental CIC-DDoS2017 dataset

3.3. Computational overhead

For each k value, we construct a model that combines multiple k -means models and a naïve Bayes classifier to identify the one with the highest accuracy. This approach significantly impacts CPU usage and memory requirements, leading to longer training time, which includes the combined durations required to train the multiple k -means models and the naïve Bayes classifier. Training multiple k -means models involves iterative clustering operations for each feature in the dataset, which demands substantial CPU power and memory. Each model updates cluster centroids and assigns data points to clusters, leading to increased CPU utilization with more k values and features. Additionally, memory demands rise as each k -means model needs to store data points, cluster centroids, and intermediate results. As shown in Figures 7 and 8, training time increases with the number of desired clusters for both datasets. For the InSDN dataset, this duration ranges from 3.28 seconds for 5 clusters to 29.59 seconds for 40 clusters. The CIC-DDoS2017 dataset shows an increase in training time from 5.20 seconds for 3 clusters to 124.06 seconds for 93 clusters. To mitigate this computational overhead, using mini-batch k -means can help reduce training time, especially with large datasets. Additionally, parallel processing and utilizing graphics processing units (GPUs) can significantly enhance performance and efficiency.

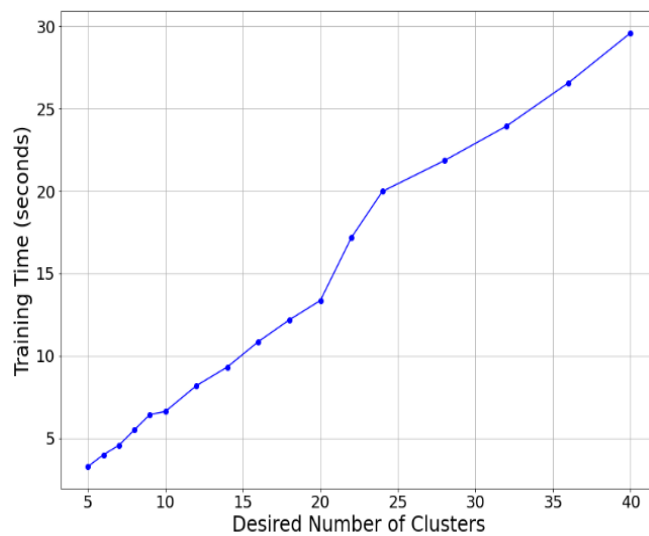


Figure 7. Training time across desired number of clusters within our experimental InSDN dataset

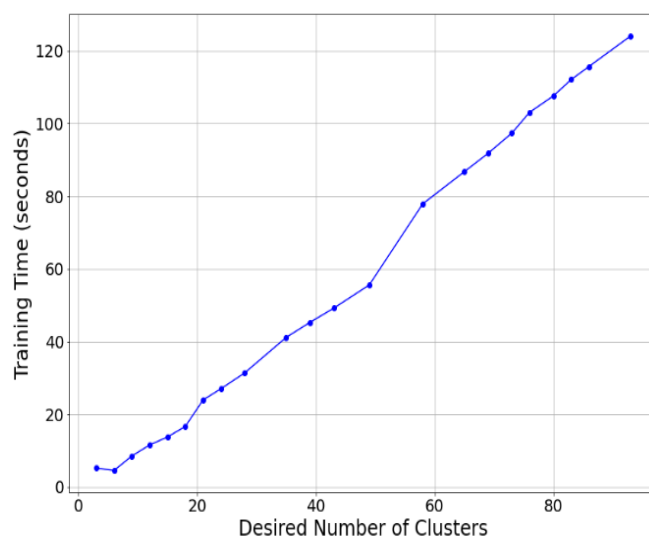


Figure 8. Training time across desired number of clusters within our experimental CIC-DDoS2017 dataset

3.4. Comparison

Table 2 compares our proposed approach with existing research, highlighting the superior performance of our hybrid model. For instance, Wang *et al.* [20] achieves 99.64% accuracy with bagging tree (BT), while our model reaches 99.98% on the InSDN dataset. Similarly, Wang and Liu [21] reports 98.98% accuracy with CNN, whereas our method attains 99.70% on the CIC-DDoS2017 dataset. Studies such as study [22] and [23] report maximum accuracies of 98.7% and 98.3% with CART and K-NN, respectively, which our model exceeds. Additionally, Khamaiseh *et al.* [24] reports 96% precision and 98% recall for known attacks with K-NN, while our method achieves higher precision and recall of 99.95% and 99.99% on the InSDN dataset.

Furthermore, Sahoo *et al.* [25] achieves an accuracy of 98.907% using a combination of kernel principal component analysis (KPCA), SVM, and genetic algorithm (GA) on the NSL-KDD and self-generated datasets. In comparison, our model significantly surpasses this with an accuracy of 99.98% on the InSDN dataset and 99.70% on the CIC-DDoS2017 dataset. Additionally, Meti *et al.* [26] reports 80% accuracy, precision, and recall using SVM on real-time TCP traffic data. Our model demonstrates superior performance with 99.98% accuracy, 99.95% precision, and 99.99% recall on the InSDN dataset. Finally, Tuan *et al.* [27] shows that K-NN achieves an accuracy rate exceeding 99%, while our model achieves a higher accuracy. Despite the promising results of our approach, addressing the computational overhead introduced by multiple k-means models is essential, as this can pose challenges in large-scale scenarios. However, implementing efficient parallelization strategies and leveraging distributed computing frameworks can mitigate these challenges, making our model suitable for large-scale deployment.

Table 2. Comparison of various DDoS detection models

Ref.	Dataset	ML techniques	Performance
[20]	DARPA, InSDN, and self-generated dataset using simulation	Support vector machines (SVM), generalized linear model (GLM), naïve Bayes (NB), discriminant analysis (DA), feedforward neural network (FNN), decision tree (DT), k-nearest neighbors (K-NN), and bagging tree (BT)	BT achieves the highest accuracy of 99.64%
[21]	CICIDS2017 and self-generated dataset	CNN, SVM, deep neural network (DNN), and DT	CNN: 98.98% (4.25% to 8.20% better)
[22]	Self-generated dataset	quadratic discriminant analysis (QDA), gaussian naïve bayes (GNB), K-NN, and classification and regression trees (CART)	CART achieved the highest accuracy of 98.7%, with all methods exceeding 95%
[23]	Self-generated dataset using hping3	SVM, K-NN, neural network, and NB	K-NN achieved the highest accuracy rate of 98.3%
[24]	Self-generated dataset by simulating different types of DDoS attacks in an SDN testbed	K-NN and ANN	K-NN: 96% precision, 98% recall; ANN: 90% precision, 86% recall
[25]	NSL-KDD and self-generated dataset using network simulator NS2	Combination of KPCA, SVM and GA	The accuracy of the KPCA+SVM+GA model is 98.907%
[26]	Dataset collected in real-time from TCP traffic	NB, SVM, and NN	SVM shows 80% accuracy, precision, and recall. SVM is considered the better choice
[27]	CAIDA 2007 and self-generated dataset	K-NN, DT and NN	The accuracy is above 98% for all 3 ML techniques. The K-NN achieved an accuracy rate of over 99% with K set to 9
The proposed model	InSDN and CIC-DDoS2017	Hybrid k-means and naïve Bayes model	In the InSDN dataset: 99.98% accuracy, 99.95% precision, 99.99% recall, and 99.97% F1 score. In the CIC-DDoS2017 dataset: 99.70% accuracy, 99.25% precision, 99.57% recall, and 99.41% F1 score

4. CONCLUSION AND FUTURE WORK





This paper explores the landscape of flooding DDoS attacks in SDN environments and the role of machine learning in detecting these threats. Flooding DDoS attacks, characterized by high-volume traffic floods, pose significant risks and require robust security measures for effective detection and mitigation. Machine learning techniques, including both supervised and unsupervised methods and ensemble approaches, have proven promising in addressing these challenges by offering adaptive and proactive defenses. Our proposed model integrates k-means clustering with gaussian naïve Bayes classification to enhance the accuracy of flooding DDoS attack detection. Through experimentation with real-world datasets such as InSDN and

CIC-DDoS2017, we have shown that our model achieves exceptional accuracy by optimizing the desired number of clusters. However, this approach introduces substantial computational overhead due to the need to run multiple k-means models. To address this, future research will focus on optimizing the computational efficiency of our approach by exploring advanced parallelization techniques, distributed computing frameworks, and hardware acceleration. This will aim to mitigate the computational challenges and enhance scalability, making our model more practical for deployment in large-scale network environments.





REFERENCES

- [1] I. Sumantra and S. Indira Gandhi, "DDoS attack detection and mitigation in software defined networks," in *2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*, Jul. 2020, pp. 1–5, doi: 10.1109/ICSCAN49426.2020.9262408.
- [2] B. Pfaff *et al.*, "Openflow switch specification, version 1.3. 1 (wire protocol 0x04)," *Open Networking Foundation*, vol. 3, 2012.
- [3] J. Cui, J. Zhang, J. He, H. Zhong, and Y. Lu, "DDoS detection and defense mechanism for SDN controllers with k-means," in *2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC)*, Dec. 2020, pp. 394–401, doi: 10.1109/UCC48980.2020.00062.
- [4] V. Deepa, K. M. Sudar, and P. Deepalakshmi, "Design of ensemble learning methods for DDoS detection in SDN environment," in *2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)*, Mar. 2019, pp. 1–6, doi: 10.1109/ViTECoN.2019.8899682.
- [5] M. S. Elsayed, N.-A. Le-Khac, and A. D. Jurcut, "InSDN: a novel SDN intrusion dataset," *IEEE Access*, vol. 8, pp. 165263–165284, 2020, doi: 10.1109/ACCESS.2020.3022633.
- [6] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Intrusion detection evaluation dataset (CIC-IDS2017)," in *Proceedings of the of Canadian Institute for Cybersecurity*, 2018.
- [7] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, 2018, pp. 108–116, doi: 10.5220/0006639801080116.
- [8] R. Abdelhadi, M. H. Alsafasfeh, and B. I. Alqudah, "Encountering distributed denial of service attack utilizing federated software defined network," *International Journal of Electrical and Computer Engineering*, vol. 14, no. 1, Feb. 2024, doi: 10.11591/ijece.v14i1.pp574-588.
- [9] D. Arivudainambi, V. K. K.A., and S. Sibi Chakkaravarthy, "Lion IDS: a meta-heuristics approach to detect DDoS attacks against software-defined networks," *Neural Computing and Applications*, vol. 31, no. 5, pp. 1491–1501, May 2019, doi: 10.1007/s00521-018-3383-7.
- [10] M. H. Zaib, "NSL-KDD dataset," *Kaggle*, 2018. <https://www.kaggle.com/datasets/hassan06/nslkdd> (accessed Nov. 10, 2023).
- [11] K. Muthamil Sudar and P. Deepalakshmi, "A two level security mechanism to detect a DDoS flooding attack in software-defined networks using entropy-based and C4.5 technique," *Journal of High Speed Networks*, vol. 26, no. 1, pp. 55–76, Mar. 2020, doi: 10.3233/JHS-200630.
- [12] J. Ye, X. Cheng, J. Zhu, L. Feng, and L. Song, "A DDoS attack detection method based on SVM in software defined network," *Security and Communication Networks*, vol. 2018, pp. 1–8, 2018, doi: 10.1155/2018/9804061.
- [13] F. Musumeci, A. C. Fidanci, F. Paolucci, F. Cugini, and M. Tornatore, "Machine-learning-enabled DDoS attacks detection in P4 programmable networks," *Journal of Network and Systems Management*, vol. 30, no. 1, Jan. 2022, doi: 10.1007/s10922-021-09633-5.
- [14] R. Swami, M. Dave, and V. Ranga, "Detection and analysis of TCP-SYN DDoS attack in software-defined networking," *Wireless Personal Communications*, vol. 118, no. 4, pp. 2295–2317, Jun. 2021, doi: 10.1007/s11277-021-08127-6.
- [15] O. Rahman, M. A. G. Quraishi, and C.-H. Lung, "DDoS attacks detection and mitigation in SDN using machine learning," in *2019 IEEE World Congress on Services (SERVICES)*, Jul. 2019, pp. 184–189, doi: 10.1109/SERVICES.2019.00051.
- [16] Ö. Kasim, "A robust DNS flood attack detection with a hybrid deeper learning model," *Computers and Electrical Engineering*, vol. 100, May 2022, doi: 10.1016/j.compeleceng.2022.107883.
- [17] N. M. and Y. B. N., "Preemptive modelling towards classifying vulnerability of DDoS attack in SDN environment," *International Journal of Electrical and Computer Engineering*, vol. 10, no. 2, Apr. 2020, doi: 10.11591/ijece.v10i2.pp1599-1611.
- [18] H. Yzzogh, "InSDN_DDoS_Exp.rar," *GitHub*, 2024. <https://github.com/Yzzogh/DDoS/tree/main> (accessed Nov. 10, 2023).
- [19] H. Yzzogh, "CIC_DDoS2017_Exp.rar," *GitHub*, 2024. <https://github.com/Yzzogh/DDoS/tree/main> (accessed Nov. 10, 2023).
- [20] S. Wang *et al.*, "Detecting flooding DDoS attacks in software defined networks using supervised learning techniques," *Engineering Science and Technology, an International Journal*, vol. 35, Nov. 2022, doi: 10.1016/j.jestch.2022.101176.
- [21] L. Wang and Y. Liu, "A DDoS attack detection method based on information entropy and deep learning in SDN," in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Jun. 2020, pp. 1084–1088, doi: 10.1109/ITNEC48623.2020.9085007.
- [22] A. O. Sangodoyin, M. O. Akinsolu, P. Pillai, and V. Grout, "Detection and classification of DDoS flooding attacks on software-defined networks: a case study for the application of machine learning," *IEEE Access*, vol. 9, pp. 122495–122508, 2021, doi: 10.1109/ACCESS.2021.3109490.
- [23] H. Polat, O. Polat, and A. Cetin, "Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models," *Sustainability*, vol. 12, no. 3, Feb. 2020, doi: 10.3390/su12031035.
- [24] S. Y. Khamaiseh, A. Al-Alaj, and A. Warner, "Flooddetector: detecting unknown DoS flooding attacks in SDN," in *2020 International Conference on Internet of Things and Intelligent Applications (ITIA)*, Nov. 2020, pp. 1–5, doi: 10.1109/ITIA50152.2020.9312310.
- [25] K. S. Sahoo *et al.*, "An evolutionary SVM model for DDOS attack detection in software defined networks," *IEEE Access*, vol. 8, pp. 132502–132513, 2020, doi: 10.1109/ACCESS.2020.3009733.
- [26] N. Meti, D. G. Narayan, and V. P. Baligar, "Detection of distributed denial of service attacks using machine learning algorithms in software defined networks," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Sep. 2017, pp. 1366–1371, doi: 10.1109/ICACCI.2017.8126031.
- [27] N. N. Tuan, P. H. Hung, N. D. Nghia, N. Van Tho, T. Van Phan, and N. H. Thanh, "A DDoS attack mitigation scheme in ISP networks using machine learning based on SDN," *Electronics*, vol. 9, no. 3, Feb. 2020, doi: 10.3390/electronics9030413.

BIOGRAPHIES OF AUTHORS

Hicham Yzzogh     received the engineering degree in telecommunications from INPT, Morocco, in 2005. With over 18 years of experience at Nokia, he has worked across diverse core network environments and possesses strong skills in coding and machine learning algorithms. He is certified as an azure solutions architect expert and is currently pursuing a Ph.D. at Mohammed V University in Rabat, Morocco. His research interests include network security, SDN, natural language processing, and image processing. He can be contacted at hicham.yzzogh@um5.ac.ma.



Hafssa Benaboud     received her Ph.D. degree in computer sciences from Burgundy University Dijon-France in 2004. In 2005, she joined as an assistant professor at Applied Sciences National School (ENSA) of Tangier, Morocco, and has been working as a full professor since 2011 in the department of computer sciences at Mohammed V University in Rabat, Morocco. She has authored more than 20 articles published in international journals and international conference proceedings. Her research interests include network protocols, network security, internet of things, traffic analysis and quality of services. She can be contacted at hafssa.benaboud@fsr.um5.ac.ma.