International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Optimized Traffic Classification System for Software-Defined Networking using a Deep Learning-based Approach

[1]Dr. Trapty Agarwal, [2]Prof. Soumya k, [3]Manish Nagpal, [4]Dr. Karishma Desai, [5]Krishna Nandan

Submitted: 07/02/2024 **Revised**: 15/03/2024 **Accepted**: 21/03/2024

Abstract: Software-Defined Networking (SDN) increases scalability and flexibility of network administration by eliminating the control as well as data planes. SDN improves network management by doing away with the control and data planes, consequential in increased scalability and flexibility. A traffic classification system for SDN improves network efficiency by classifying the data flows. Quality of Services (QoS) enhances and optimizes use of resources with flexible adaptation to changing network requirements. To create an optimal traffic classification system for SDN, we proposed a novel Deep Learning (DL) approach called Lightning Search fine-tuned Generative Adversarial Networks (LS-GAN). We collected a dataset comprising several kinds of network traffic logs to train the suggested methodology. The obtained raw data is pre-processed using the Unit Vector Transformation (UVT) technique. Kernel Principal Component Analysis (K-PCA) is used with the processed data to determine the key features. The LS-GAN approach combines the potent capabilities of Generative Adversarial Networks (GANs) with blazingly quick search algorithms. The system can effectively and precisely detect different kinds of network traffic inside SDN designs by combining these methods. The proposed LS-GAN obtained a Precision (96.2%), Accuracy (98.3%), Recall (97.3%) and F1-score (98.6%). The experimental outcome show that the suggested LS-GAN approach performed better than existing approaches in SDN infrastructure for increased traffic classification.

Keywords: *Deep Learning (DL), Lightning Search fine-tuned Generative Adversarial Networks (LS-GAN), Software-Defined Networking (SDN).*

1. Introduction

Numerous applications have experienced tremendous development as a result of the massive rise in data and the correspondingly high network requirements. Conventional network devices need closed interfaces or proprietary protocols to function [1]. The network administrator can utilize the information from the network traffic analysis to make well-informed decisions and allocate resources as efficiently as feasible. The convergence of networking technologies such as cloud computing has led to the rapid growth of Internet of Things (IoT) data [2].

Globally, there are exponentially more connections and devices than people and Internet users, which are driving up network traffic. Furthermore, traffic flow patterns and network performance have been dramatically altered by the adoption of new devices with notable intelligence and capabilities, along with the consequent development of new applications as well as services [3]. The demand for bandwidth usage has increased dramatically because of self-driving cars, cloud computing, etc., giving network operators more opportunities to explore novel ideas in network management [4]. The underlying network technology helps the operator to manage the entire network comprehensively and consistently [5]. Modern network infrastructures might be managed and controlled with the help of software-defined networks, or SDNs. SDNs are being used in many different areas, including smart cities, transportation, healthcare and agriculture. The emergence of SDN has brought significant modifications to modern communication networks [6]. Research on wireless networks has been the main emphasis of the networking technology field. Although there are certain security and data rate issues with its acceptance as a mainstream technology, numerous standards have been created throughout time to address these issues [7]. The rapid growth of technology and the advantages that the IoT provides to improve our lives, the number of people using the Internet and connected devices has expanded dramatically. Qos, distribution of resources, scheduling of networks and security-related concerns are currently

[1]Associate Professor, Maharishi School of Engineering and Technolgy, Maharishi University of Information Technology, Lucknow, India - 226036, Email Id- trapty@muit.in, Orcid Id- 0009-0007-4081-4999
[2]Assistant Professor, Department of Computer Science and Information Technology, Jain (Deemed to be University), Bangalore, Karnataka, India, Email Id- soumya.k@jainuniversity.ac.in, Orcid Id- 0000-0002-6657-386X
[3]Centre of Research Impact and Outcome, Chitkara University, Rajpura-140417, Punjab, India, Email ID : manish.nagpal.orp@chitkara.edu.in, Orcid Id:https://orcid.org/0009-0000-9823-5251
[4]Associate Professor, Department of ISME, ATLAS SkillTech University, Mumbai, Maharashtra, India, Email Id-karishma.desai@atlasuniversity.edu.in, Orcid Id- 0009-0009-5171-9588
[5]Assistant professor, Department of Mechanical Engineering, Vivekananda Global University, Jaipur krishna_nandan@vgu.ac.in, Orcid Id- 0009-0004-7228-3772

addressed using SDN [8]. SDN has three distinct planes: Control Plane (CP), the Application Plane (AP) and Data Plane (DP). As the network operating system (NOS), the centrally located CP uses the network policies [9]. This study's goal is to develop a traffic classification system for SDN using a Lightning Search fine-tuned Generative Adversarial Networks (LS-GAN).

This part of the paper follows a similar format as the rest: part 2 discusses about literature review. In part 3, the proposed LS-GAN is covered in great detail. In part 4, the suggested strategy's experimental design, results and performance evaluation are discussed. The conclusions are summarized in part 5 along with the suggestions for more study.

2. Literature Survey

The study [10] examined the traffic flows on the SDN for many protocols. Data was gathered based on the features generated by the SDN controller in the physical network to create a real-time dataset. The study [11] sought to optimal routing in SDN through the use of Gated Recurrent Unit (GRU)'s capacity to predict traffic using data for network performance that were collected at various times. The study [12] looked at the SDN Quality of Service (QoS) assurance system to classify traffic. The architecture comprised a 2 stage classification system: the classifier was trained and evaluated offline, as well as its speed was tested online using spark streaming to simulate flows. The study [13] created a Spike Elman Neural Network based on spider monkeys to detect intrusion risks in SDN. The research [14] looked at the DRL-Idle Deep Reinforcement Learning (DRL) system, this might maximize the SDN network's flow service time without considering long-term status assumptions. The research [15] offered a SDN Intelligent Intrusion Detection System to detect DDoS attacks. The study [16] demonstrated the application of temporal deep Q learning (TDQN) in the Down Syndrome Diagnosis Network (DSDN) controller. An autonomous reinforcement-learning model called TDQN was used in the article. The article [17] was to broaden its application to the new network architecture known as SDN, which was an emerging topic. A modified Golden Jackal Optimization (mGJO) was suggested to improve the GJO's performance. The study [18] looked at a Deep Q Residue technique for load-balancing phase analysis of both conflicting and normal flow circumstances. Tensor Flow was used in the simulation to create an open SDN network.

3. Methodology

This section covers the use of a LS-GAN approach to create an optimized traffic classification system for SDN. Fig.1 depicts the methodological framework.

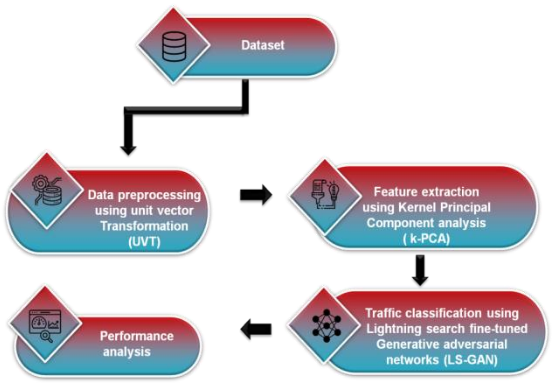
Dataset
Data preprocessing using unit vector Transformation (UVT)
Feature extraction using Kernel Principal Component analysis (k-PCA)
Traffic classification using Lightning search fine-tuned Generative adversarial networks (LS-GAN)
Performance analysis

Fig 1. Methodological Framework

3.1 Dataset

The Distributed Internet Traffic Generator (D-ITG) program was used to generate traffic flow data. The following traffic categories were employed in the traffic classification training dataset: Telnet, Ping, DNS, Voice and Video. Three classes were employed for the classification: audio and video traffic went into the Voice and Video classes [19]. Table 1 displays a sample dataset.

Table 1. Sample Dataset

Packets Forward	Average Forward Packets/second	Inverse Packets	Type of Traffic
856	0.10483871	508	Data
1622	0.320512821	507	Video
454	1.166666667	507	Voice
856	0.105691057	508	Data
790	0.923076923	1018	Voice
856	0.09352518	507	Video

3.2 Data preprocessing by using Unit Vector Transformation (UVT)

The traffic data is preprocessed using a technique called Unit Vector Transformation. When a vector is scaled to a unit length, it is either stretched or shrunk to a unit sphere. When the changed data is applied to the entire network traffic dataset is as indicated by equation (1).

$$norm(w) = \sqrt{w_1^2 + w_2^2 + \cdots + w_m^2} \qquad (1)$$

Determine the length, or norm, for every sample vector. The Euclidean norm formula, which is the square root of the sum of the vector's squared elements, can be used to determine the norm of a vector as shown in equation (2).

$$unit_{vector(w)} = \frac{w}{norm(w)} \qquad (2)$$

Divide the vector's elements by their respective norms. This gives the vector an overall unit norm by scaling each element proportionately. Finally the data will be preprocessed using UVT.

3.3 Feature Extraction by using Kernel Principle Component Analysis (K-PCA)

The preprocessed data will be extracted by using Kernel Principle Component Analysis (K-PCA). The classic Principal Component Analysis (PCA) technique, which is used in data analysis for feature extraction and dimensionality decrease, is extended by K-PCA. Assume the map serves as the focus of the information, $\sum_{r=1}^{N} \emptyset(y_r) = 0$in the feature space L, then the covariance matrix in L is$V^L = \frac{1}{N}\sum_{j=1}^{N} \emptyset(y_j)\emptyset(y_j)^D$. The next stage is to find the Eigen values $\lambda^L \geq 0$ and eigenvectors $U^L \in L/\{0\}$ satisfying$V^L\lambda^L = \lambda^L U^L$. The range of $\emptyset(y_r), \ldots \emptyset(y_n)$ contains every solution U^L with$\lambda^L \neq 0$. There are coefficients such that $U^L = \sum_{j=1}^{N} \alpha_j \emptyset(y_r)$. exist for$\alpha_j (j = 1, \ldots., N)$. Thus, can discover in equation (3):

$$\lambda^L = \sum_{j=1}^{N} \alpha_j \left(\emptyset(y_r).\emptyset(y_j)\right) = \frac{1}{N}\sum_{j=1}^{N} \alpha_j \left(\emptyset(y_r).\sum_{i=1}^{N} \emptyset(y_i))\emptyset(y_i) . \emptyset(y_j)\right) r = 1, \ldots, N \qquad (3)$$

Defining an $N \times N$ matrix R by using equation (4),

$$R_{ji} = \emptyset(y_i).\emptyset(y_i)). \qquad (4)$$

Equation (5) can be written as $N\lambda^L R\alpha = R^2\alpha$. This is equivalent to:

$$N\lambda^L\alpha = R\alpha \qquad (5)$$

Let $\lambda_1^L. \geq \lambda_2^L \geq \cdots \geq \lambda_N^L$ signify the Eigen values, the complete set of eigenvectors corresponding to equation (6) is$\alpha_1, \alpha_2, \ldots, \alpha_N$.
$(U_r^L.U_r^L) = 1$ is equivalent to

$$(U_r^L.U_r^L) = \sum_{j,i=1}^{N} \alpha_j^r \alpha_i^r \left(\emptyset(y_i).\emptyset(y_i)\right) = \sum_{j,i=1}^{N} \alpha_j^r \alpha_i^r R(y_j, y_i) = \lambda_1^L(\alpha^r.\alpha^r) = 1; r = 1,.., f \qquad (6)$$

Calculating a data point's projections onto the eigenvectors U_r^Lin L is necessary to identify its primary component. Assume that y is a test point in L with a projection of$\emptyset(y)$, therefore the eigenvectors projecting$\emptyset(y)$, U_r^L is the nonlinear principal components corresponding to equation (7):

$$\left(U_r^L.\emptyset(y)\right) = \sum_{j=1}^{N} \alpha_j^r (\emptyset(y_i).\emptyset(y_i)) \qquad (7)$$

3.4 Traffic classification for SDN using Lightning Search fine-tuned Generative Adversarial Networks (LS-GAN)

Lightning search is included into the GAN architecture in the context of LS-GAN to improve training process efficacy and efficiency. Because of this integration, the GAN model performs better overall and can converge more quickly during training. This approach can entail optimizing or expediting the process of fine-tuning the GANs' parameters.

3.4.1 Generative Adversarial Network (GAN)

The GAN adversarial training is to produce a phony sample that seems authentic. Two parts make up the antagonistic training: a discriminator and a generator. The discriminator assesses whether the fake images generated by the generator are authentic or fraudulent. For each cycle of training a mini-batch of real images (W) and sounds (Y) is selected at random. The generator network (H) then generates counterfeit images $(H\ (Y))$. The discriminator network C outputs a probability, $P\ (D\ = 1|X)$ proving the authenticity of X. The discriminator takes in either W or $H\ (Y)$ as input and outputs the probability. These two neural networks are trained at the same time. The objective function listed below in equation (8-9) can be used to optimize the GAN:

$$\hat{H} = \underset{H}{\operatorname{argmin}} \left\{K_{adv}\left(q, C\left(H(Y)\right)\right)\right\} \qquad (8)$$

$$C = arg\min_{C}\{K_{adv}\left(q, C(W)\right) + K_{adv}\left(e\left(, C\left(H(Y)\right)\right)\right\} \qquad (9)$$

Where e and q have corresponding values of zero and one. It shows GAN is an effective method for classifying traffic in SDN.

3.4.2 Lightning Search Optimization (LSO)

A meta-heuristic algorithm with natural phenomena is the LSO. The step leader propagation technique serves as the foundation for the LSO. According to this technique, the step ladder's binary tree structure is formed by the projectile motion. These projectiles serve as the algorithm's representation of the population size. Equation (10) controls the projectile's velocity.

$$U_o = \left[1 - \left(\frac{1}{\sqrt{1-\left(\frac{v_0}{D}\right)^2 - \frac{TE_j}{(nd^2)}}}\right)^{-1}\right]^{\frac{-1}{2}} \qquad (10)$$

Where n is the mass of the projectile, In a vacuum, light moves at the speed of D, υ_0 is the projectile's beginning velocity, and s is the length of the path traveled. E_j is also the constant ionization rate.

A random direction is chosen for the transition projectile's ejection. Therefore, the transition projectile is modeled using the standard uniform probability distribution, as shown in Equation (11):

$$e(W^S) = \begin{cases} \frac{1}{a} - b \; for \; a \le W^S \le a \\ 0 \; for \; W^S < b \; or \; W^S > a \end{cases} \tag{11}$$

Where a and b represent the lower and upper bounds, respectively, and W^S is the first tip energy of the leader of steps I or a random value that produces the solution. To get the solution, N random projectiles must be fired at a population of N step leaders. The LSO algorithm 1 is as follows:

Algorithm 1: LSO

```
Step 1: Initialize the population
Step 2: Generate population randomly
Step 3: Initialize the direction for each dimension
While (Termination condition satisfies) do
Step 4: Determine the fitness of every answer
Step 5: Sort the population
Step 6: Describe the best and worst demographics
Step 7: Revise the energy
Step 8: Each dimension's direction should be updated
Step 9: For each I=1: N where N ∈ Pop
Step 10: if (Pop (i) == Leader Resolution) then
Step 11: Adjust the pop's location.
Step 12: else if (Pop(i)==Space Solution)then
Step 13: Revise the Pop's location
Step 14: end if
Step 15: if (Pop(i) is not improvised) then
Step 16: Initialize new Pop
Step 17: end if
Step 18: end for
Step 19: end while
```

The energetic projectile ionizes the surrounding elder leader after collecting suggestions from N step leaders. Equation (12) provides the probability density function's exponential distribution:

$$e(W^t) = \begin{cases} \frac{1}{\mu} f^{-W^S/\mu} \; for \; W^S \ge 0 \\ 0 \quad for \; W^S \ge 0 \end{cases} \tag{12}$$

In the subsequent phase, the space projectile's position or trajectory is determined by the shaping parameter μ. Equation (13) comes after the projectile's position:

$$O_{inew}^T = O_j^T \pm \exp rand \left(\mu_j\right) \tag{13}$$

Equation (14) displays the normal probability distribution function for the lead bullet with a scalar parameter σ:

$$e(W^K) = \frac{1}{\sigma\sqrt{2\pi}} r^{-(W^K - \mu)^2/2\sigma^2} \tag{14}$$

The lead projectile's position can be represented by the following equation (15):

$$O_{new}^K = O^K + normrand(\mu_K, \sigma_K) \tag{15}$$

An extension is granted to the step leader and the lead projectile's position is adjusted if the new location offers a better solution.

4. Results and Discussion

In this study, the Python 3.9.16 was used together with the UNIX OS and 13.62 GB of RAM for evaluating the proposed LS-GAN. Table 2 displays the parameters for evaluation.

Table 2. Performance metrics

Parameter	**Formulae**
Accuracy	$\frac{TP + TN}{TP + FP + FN + TN}$
Precision	$\frac{TP}{TP + FP}$
Recall	$\frac{TP}{TP + FN}$
F1-score	$2 \times \frac{(Precision \times Recall)}{(Precision + Recall)}$

The dependability and effectiveness of the proposed LS-GAN are compared to those of more established techniques like Convolutional Neural Network (CNN) [20], Multi-layer perceptron (MLP) [20], and Stacked Auto-Encoder (SAE)

[20]. Table 3 shows the numerical outcomes of classification methods.

Table 3. Numerical Outcomes of classification methods

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
CNN [20]	87.2	85.1	84.4	84.8
MLP [20]	87.1	84.4	87.7	84.7
SAE [20]	87.2	85.1	84.2	84.7
LS-GAN [Proposed]	**98.3**	**96.2**	**97.3**	**98.6**

Accuracy is the application of properly classifying the total number of occurrences. The accuracy of the recommended and current systems is shown in Fig. 2.The LS-GAN has a rate of accuracy of 98.3%, which is more than the accuracy rates of 87.2% for CNN, 87.1% for MLP, and 87.2% for SAE. This demonstrates that the LS-GAN is more accurate than the existing one.

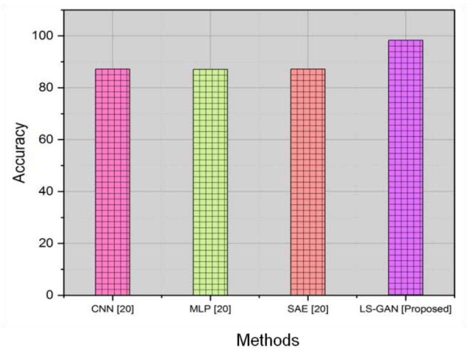

Fig 2. Comparison of Accuracy

Precision describes the accuracy of a prediction or classification model. Fig. 3 displays the precision for the suggested and existing methods. The LS-GAN has a rate of precision of 96.2%, which is more than the precision rates of 85.1% for CNN, 84.4% for MLP, and 85.1% for SAE. This demonstrates that the LS-GAN is more accurate than the existing one.

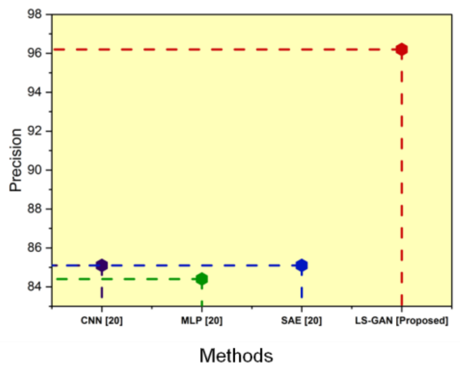

Fig 3. Comparison of Precision

The total number of True Positives (TP) minus the total number of false negatives (FN) is the mathematical formula for recall. Fig. 4 depicts the suggested and existing system recall. The LS-GAN has a rate of precision of 97.3%, which is more than the recall rates of 84.4% for CNN, 87.7% for MLP, and 84.2% for SAE. This demonstrates that the LS-GAN recall is higher than the existing technique.

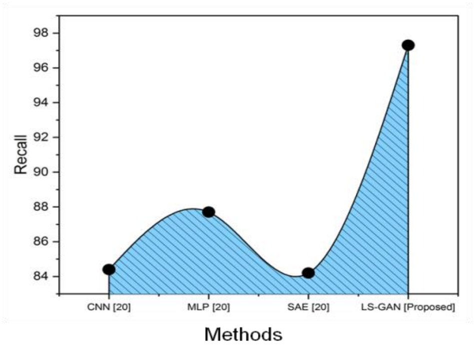

Fig 4. Comparison of Recall

The F1 score is a statistic that combines accuracy and recall to assess a classification model or system's overall efficacy. F1-score of the LS-GAN and existing approach is displayed in Fig. 5. The LS-GAN achieved 98.6% of the F1-score, comparing to CNN 84.8%, MLP 84.7%, and SAE 84.7%. This shows that the LS-GAN F1-score outperforms the existing approach.

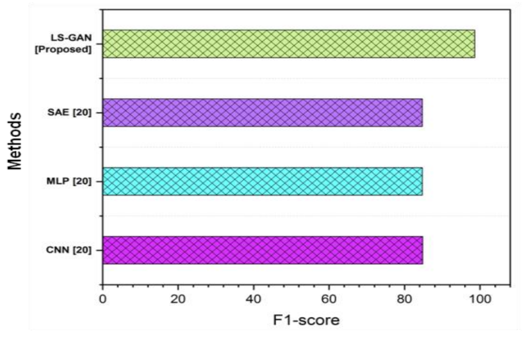

Fig. 5 Comparison of F1-score

5. Conclusion

Computer vision, voice recognition, and other fields are among the numerous applications that DL techniques are finding to be one of the most interesting and practical areas. In this study, we employed the supervised learning approach to investigate traffic classification in an SDN environment. Through the utilization of LS-GAN, the suggested traffic categorization system exhibits its efficacy in managing a wide range of intricate traffic patterns, rendering it a viable instrument for SDN implementations. The approach's optimization results in better decision-making processes in SDN controllers, which in turn improves network security, lowers latency, and boosts overall efficiency. The aforementioned statistics indicate that the LS-GAN technique's Accuracy

(98.3%), Precision (96.2%), Recall (97.3%), and F1-score (98.6%) respectively. This shows that LS-GAN outperforms the existing techniques. We demonstrated the effectiveness of these algorithms in accurately categorizing network traffic in such a setting and the influence of feature selection on the outcomes. Network slicing in SDN networks with QoS mechanisms and effective traffic classification may be the subject of future research.

References

[1] Raikar, M. M., Meena, S. M., Mulla, M. M., Shetti, N. S., & Karanandi, M. (2020). Data traffic classification in software defined networks (SDN) using supervised-learning. *Procedia Computer Science*, *171*, 2750-2759

[2] Pradhan, B., Hussain, M. W., Srivastava, G., Debbarma, M. K., Barik, R. K., & Lin, J. C. W. (2022). A neuro-evolutionary approach for software defined wireless network traffic classification. *IET Communications*

[3] Nunez-Agurto, D., Fuertes, W., Marrone, L., & Macas, M. (2022). Machine Learning-Based Traffic Classification in Software-Defined Networking: A Systematic Literature Review, Challenges, and Future Research Directions. *IAENG International Journal of Computer Science*, *49*(4)

[4] Perera Jayasuriya Kuranage, M., Piamrat, K., & Hamma, S. (2020). Network traffic classification using machine learning for software defined networks. In *Machine Learning for Networking: Second IFIP TC 6 International Conference, MLN 2019, Paris, France, December 3–5, 2019, Revised Selected Papers 2* (pp. 28-39). Springer International Publishing.

[5] Shukla, P. K., Maheshwary, P., Subramanian, E. K., Shilpa, V. J., & Varma, P. R. K. (2023). Traffic flow monitoring in software-defined network using modified recursive learning. *Physical Communication*, *57*, 101997

[6] Masood, F., Khan, W. U., Jan, S. U., & Ahmad, J. (2023). AI-enabled traffic control prioritization in software-defined IoT networks for smart agriculture. *Sensors*, *23*(19), 8218

[7] Kumar, R., Venkanna, U., & Tiwari, V. (2023). Optimized traffic engineering in Software Defined Wireless Network based IoT (SDWN-IoT): State-of-the-art, research opportunities and challenges. *Computer Science Review*, *49*, 100572

[8] Eissa, M. E., Abdel Azim, M., & Ata, M. M. (2023). Design of an optimized traffic-aware routing algorithm using integer linear programming for software-defined networking. *International Journal of Communication Systems*, e5517

[9] Yusuf, M. N., Bakar, K. B. A., Isyaku, B., Osman, A. H., Nasser, M., & Elhaj, F. A. (2023). Adaptive Path Selection Algorithm with Flow Classification for Software-Defined Networks. *Mathematics*, *11*(6), 1404

[10] Ashour, M. M., Yakout, M. A., & AbdElhalim, E. (2024). Traffic Classification in Software Defined Networks based on Machine Learning Algorithms. *International Journal of Telecommunications*, *4*(01), 1-19

[11] Gunavathie, M. A., & Umamaheswari, S. (2024). Traffic-aware optimal routing in software defined networks by predicting traffic using neural network. *Expert Systems with Applications*, *239*, 122415

[12] Eissa, M. E., Mohamed, M. A., & Ata, M. M. (2024). A robust supervised machine learning based approach for offline-online traffic classification of software-defined networking. *Peer-to-Peer Networking and Applications*, *17*(1), 479-506.

[13] Charanarur, P., Thanh Hung, B., Chakrabarti, P., & Siva Shankar, S. (2024). Design optimization-based software-defined networking scheme for detecting and preventing attacks. *Multimedia Tools and Applications*, 1-19.

[14] Jiménez-Lázaro, M., Berrocal, J., & Galán-Jiménez, J. (2024). Flow-based Service Time optimization in software-defined networks using Deep Reinforcement Learning. *Computer Communications*, *216*, 54-67

[15] Shaji, N. S., Muthalagu, R., & Pawar, P. M. (2024). SD-IIDS: intelligent intrusion detection system for software-defined networks. *Multimedia Tools and Applications*, *83*(4), 11077-11109.

[16] Sharma, A., Balasubramanian, V., & Kamruzzaman, J. (2024). A Temporal Deep Q Learning for Optimal Load Balancing in Software-Defined Networks. *Sensors*, *24*(4), 1216

[17] Qiu, F., Xu, H., & Li, F. (2024). Applying modified golden jackal optimization to intrusion detection for Software-Defined Networking. *Electronic Research Archive*, *32*(1), 418-444

[18] Ananth, B. (2024). Hybrid Support Vector Machine for Predicting Accuracy of Conflict Flows in Software Defined Networks. *Salud, Ciencia y Tecnología*, *4*, 797-797

[19] Karn, G., Sapkota, B., & Dawadi, B. R. (2023). Traffic Classification and Load Balancing in SDN Environment

[20] Chang, L. H., Lee, T. H., Chu, H. C., & Su, C. W. (2020). Application-based online traffic classification with deep learning models on SDN networks. *Adv. Technol. Innov*, *5*(4), 216-229.