University of Baghdad
College of Engineering
JE
JOURNAL OF ENGINEERING

Journal of Engineering
journal homepage: www.jcoeng.edu.iq
Volume 31 Number 5 May 2025

Handling Heterogeneous Traffic for Software Defined Data-Center Network Using Spike Neural Network

Sanarya Jamal Al-Azawee *, **Nadia Adnan Shiltagh Al-Jamali**

Department of Computer Engineering, College of Engineering, University of Baghdad, Baghdad, Iraq

ABSTRACT

Software Defined Networking (SDN) allows for more flexible network administration than traditional architectures. Software-defined networks (SDNs) efficiently manage data flows and optimize network resources. However, heterogeneity influences the quality of the services. (QoS) needs and network resource demands. They behave differently when traveling to their last point. Currently, numerous data center networks (DCNs) struggle with the unfair use of several network resources by big packets (Elephant flowing) arriving during any instant affecting specific flows (mice flow). Elephant Flows (EF) account for just a small percentage of the entire traffic. Nevertheless, they are considered Long-lasting (LLF) and often burn network resources. Their actions cause congestion and delays in most Mice Flows (MF). Forecasting and categorizing flow traffic is essential for optimal resource usage, QoS provisioning, and reducing network congestion and delays. This paper suggested a third-generation Single Spike Neural Network (SSNN) supervised learning approach using temporal coding to identify heterogeneous traffic. The classifier approach uses three features: flow time, byte rate, and packet rate. The SSNN is then taught to categorize the traffic into two classes. This training classifies two types of traffic: elephant and mice flow. The effectiveness of the algorithm was examined when classifying traffic using many metrics and its efficiency was proven as it was able to reduce the average error and its accuracy reached 99%. The suggested model's usefulness is demonstrated by its efficient training procedure, which provides rapid and accurate results.

Keywords: Heterogeneous traffic, Software Defined Networking (SDN), Elephant flows, Mice flows.

1. INTRODUCTION

Most professional business organizations, regardless of size, believe that having their data centers (DCs) is necessary for effective competition. However, the fast expansion and the spread sites of the data centers raise the difficulty of control and management operations.

*Corresponding author

Peer review under the responsibility of University of Baghdad.
https://doi.org/10.31026/j.eng.2025.05.02

Article received: 24/10/2024
Article revised: 15/01/2025
Article accepted: 03/03/2025
Article published: 01/05/2025

Manual network configuration can take hours, days, or even weeks for operators to complete for individual devices. The Software Defined Network (SDN) addresses the limitations within traditional DC networks **(Darabseh et al., 2015)**. It streamlines the management of networks by segregating the control and infrastructure planes. This enhances network flexibility and efficiency **(Priyadarsini and Bera, 2021)**. In a DCN network, SDN separates the data plane (communication switches) and control plane (controller). The third layer is the application plane, which provides user-driven programs to utilize the DCN network through the northbound interface **(Isyaku et al., 2020**). However, the controller uses standard artificial intelligence and neural network methods for routing algorithms in current systems. Flow routing is crucial for optimizing network performance. The primary goal of flow management in the network is to obtain the necessary data in a short amount of time. Routing can improve network performance, providing an obvious benefit **(Wang et al., 2008; Mendiola et al., 2016)**. The traffic flow is heterogeneous, with varying arrival rates, durations, and sizes. The heterogeneous traffic impacts the quality of service (QoS) requirements of the network as well as resource demands. They act differently on the road to their final destination. **(Yusuf et al., 2023)**. Network flows are divided into elephant flows (big and long-lived) and mice flows (tiny, short-lived). Large flows, like Elephant Flows (EF), account for just 10% of the entire traffic. Nevertheless, they are considered Long-lasting (LLF) and use network buffers. As a result, their behavior causes congestion and delays in the majority of Mice Flows (MF). However, MFs demand high priority due to their delay sensitivity **(Yusuf et al., 2023)**. SDN architecture offers a global perspective and greater network programmability, enabling flexible capabilities and effective QoS provisioning techniques. Network performance deterioration is mostly due to congestion and imbalanced load, which can be caused by inefficient traffic routing. Many researchers have proposed solutions to reduce network congestion and balance network load, others have suggested improving traffic routing, and others have suggested traffic classification.

(Cui and Xu, 2016) provides a load-balancing solution, based on SDN's global network perspective. The suggested design Uses an additional load balancer, Which manages traffic across all the channels. After obtaining route information from the SDN controller, the load balancer analyzes the load status of various pathways and selects the least loaded path. The SDN architecture becomes more complex when an extra load balancer is implemented. This could entail extra expenses, setup work, and difficulties integrating with current network systems. **(Li et al., 2019)** offered a dynamic multi-controller distribution method based on load balancing. It converts flow inputs into a queuing mode and investigates traffic propagation latency and controller capability as two major elements influencing multi-controller deployment to prevent the issue of low performance in the network. Here we will enter into questions about the number of controllers that must be added, the number of switches under each controller's control, and whether it is under central control or separate control. The authors of **(Cheng and Jia, 2020)** introduced and implemented Network-Aware Multi-pathing which takes into account network heterogeneity bandwidth to minimize transmission time for flow groups. NAMP significantly lowers transmission time, according to findings from experiments. Although NAMP takes network heterogeneity bandwidth into consideration, it could have trouble adjusting to unexpected spikes in traffic or link failures. Temporary performance deterioration could result from traffic rerouting being delayed due to the overhead of dynamically recalculating pathways.

The trade-off between latency and bandwidth utilization, NAMP may not always give latency-sensitive applications priority because it concentrates on reducing transmission time based on available capacity. **(Zaher et al., 2021)** introduces Sieve, a novel distributed SDN-based platform for flow scheduling. Sieve first arranges a fraction of the flows according to available bandwidth, regardless of class. Create a heuristic, adaptable, and scalable method for scheduling elephant and mice flows. Although heuristic approaches are quick and scalable, they might not always identify the best solutions, especially in networks with wildly fluctuating and unexpected traffic loads. **(Shirali-Shahreza and Ganjali, 2018)** focuses on the restricted flow table issue. This technique predicts TCP flow termination from RST/FIN packets to speed up rule evictions while incubating non-TCP flows to postpone rule installation. It decreases the average flow table occupancy. This method consists of two strategies. Firstly, it deletes installed rules as soon as the relevant flow is complete or nearing completion. This simple adjustment can significantly reduce flow table occupancy. Also, employ a second mechanism to defer rule implementation for non-TCP traffic. For short flows, it will wait for numerous packets before advancing them. Flow rules for flows that might still contain active traffic may be removed too soon as a result of the expedited eviction policy. This might lead to repeated flow rule reinstallations, which would increase controller load and cause more delays. **(Liu et al., 2021)** presents A DRL-based routing system for SD-DCN. the agent on the controller for SDN is taught from the data of the system and makes adaptive routing decisions based on its state. First, it presents a mechanism for recombining several network resources (bandwidth and cache). Second, it presents a routing strategy with a resource-recombining state. They make use of new data-center network (DCN) characteristics (bandwidth, cache) since the cache should influence routing decisions because it can remove redundant traffic in DCN. Deep learning-based systems, such as DRL-R, frequently consume a large amount of energy, both during training and in real-time operation. This is a worry for data centers, which are already under pressure to improve energy efficiency and reduce operational expenses.

(Kumar et al., 2020) demonstrates How machine learning schemes may identify the lowest congested route for routing traffic via the network of SDN. A suggested route-finding technique creates a list of available paths by using network data from the SDN controller. Training and running machine learning models can be computationally expensive, resulting in significant energy usage. This is especially concerning for SDNs in large-scale data centres. **(Modi and Swain, 2023)** suggested routing technique for SD-DCN uses RNN deep learning algorithms that encompass LSTM and BiLSTM. It suggests an effective routing path combination by leveraging prior traffic statistics making the controller for SDN work better. When trained on sparse or skewed datasets, deep learning models such as LSTM and BiLSTM are susceptible to overfitting. When a model performs well on historical traffic but badly on unforeseen real-world scenarios, overfitting may result in poor generalization. **(Lin et al., 2014)** suggests an effective method for detecting elephant traffic in data centers. The authors suggest a Hierarchical Statistics-pulling technique that uses aggregate statistical information to isolate elephant flows. When there is significant network traffic, the hierarchical technique may cause delays in detecting elephant flows. **(Aymaz and Cavadar, 2023)** suggested technique identifies elephant flows to enhance routing efficiency. For elephant flow identification, this study used the Deep Learning technique. The presented classifier classifies flow using a Convolutional Neural Network (CNN) structure. Significant computational resources are needed for deep learning model training, particularly when complicated architectures are being used to obtain high accuracy. Longer training sessions and higher energy usage may result from this. Because deep learning models need time for processing and inference, they may cause delays in flow detection. **(Yusuf et al., 2023)**

integrates with the SDN controller to determine appropriate pathways for each traffic class. As a result, this work provides a strategy that combines composite measurements with flow categorization to identify congestion flows and redirect them via the optimal routes to prevent congestion and losses. The authors proposed an Algorithm that uses flow classification characteristics and composite path parameters such as traffic volume, latency, and bandwidth to solve these concerns.

Temporary congestion and less-than-ideal routing choices may result from the algorithm's inability to respond promptly enough to unexpected traffic spikes.**(Al-Saadi et al., 2023)** presents an SDN application that uses network performance data like flow duration, packet count, and average packet size to determine the optimal path for each flow type. These measurements are utilized to categorize flows as elephants and mice using K-means clustering. When applied to large-scale flow data, unsupervised machine learning models—particularly clustering methods like k-means or DBSCAN—can be computationally costly and to maintain optimal performance, the clustering model parameters (such as the number of clusters and distance metrics) may need to be adjusted frequently, which would increase operational complexity.

Handling traffic heterogeneity requires an accurate analysis of the traffic to categorize it for more efficient routing. This article uses the third generation of neural network, which is the spike neural network, which had not been employed in any earlier studies, to categorize network traffic as mice or elephant flows based on predetermined characteristics. Although it has been used in many applications and has given good results, for example, it has been used in terrain classification **(Ibrahim and Al-Jamali, 2024)**, image classification **(Shears and Yazdani, 2020)**, mobile robots **(Abubaker et al., 2023)** and many other applications.

2. THE PROPOSED MODEL OF TRAFFIC CLASSIFICATION

To address traffic classification issues with SDN and use the powerful performance of spiking neural networks, a strategy for applying them to SDN traffic classification has to be developed. This is critical for achieving high classification accuracy and efficiency. **Fig. 1** depicts the suggested model, which includes feature collection and classification. In reality, the traffic of software-defined data center networks is heterogeneous. Hence, the features employed to distinguish across training classes are (flow duration, byte rate, and packet rate); it was also used in prior research **(Al-Saadi et al., 2023)**. Those features are provided for the classifier part. In the current study, the SNN classifies the traffic into two classes: elephant and mice flow. The dynamic routing will then be done according to the categorization result in future work.

3. SINGLE SPIKE NEURAL NETWORK STRUCTURE

SSNN architecture, like classic ANN, is composed of three types of layers: input or encoding, hidden, and output layers. Part (A) of the **Fig. 2** depicts the feed-forward structure of SNN. SNN design differs from the second generation of ANN. The SNN has more connections between two neurons than ordinary ANNs **(Roy et al., 2017).** Each synaptic or connection between two spiking neurons has a delay value and weight as illustrated in part (B). Every neuron in SNN comprises three calculation steps: the first is the summing of all input spikes to create the membrane potential. The second step is to determine whether the membrane potential is above the threshold value. In the third phase, the spiking neuron only generates a spike when its membrane potential reaches the threshold value. The membrane potential of the spiking neuron is then reset to zero **(Shihab, 2016).** The SNN structure in this research has three input neurons for defining features, 20 hidden neuronal, and one output neuron to represent two classes either mice or elephant classes

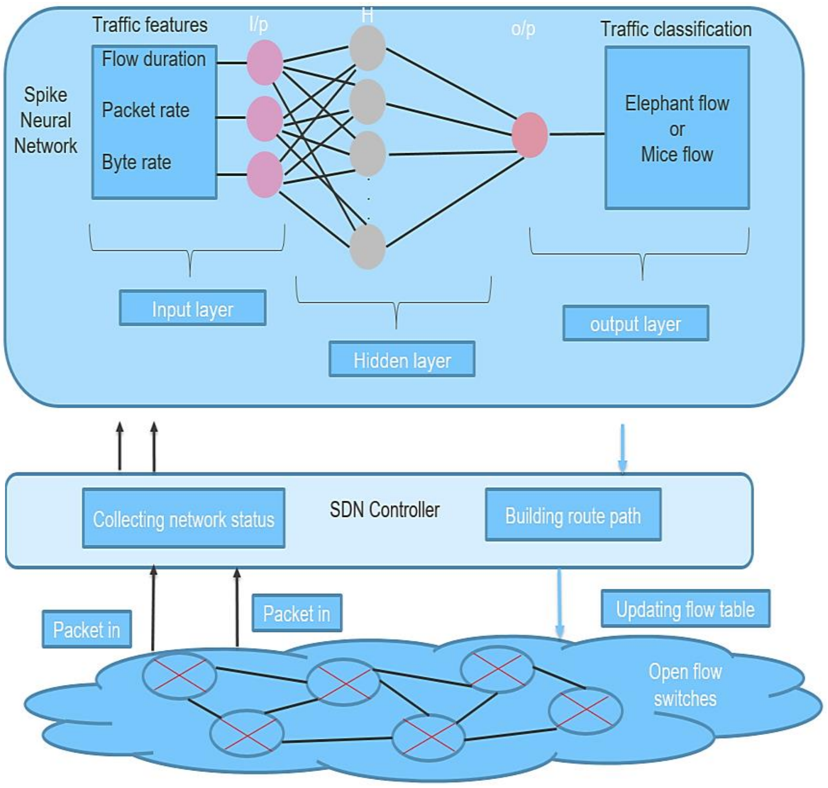

Figure 1. shows the suggested paradigm for spike terrain classification..

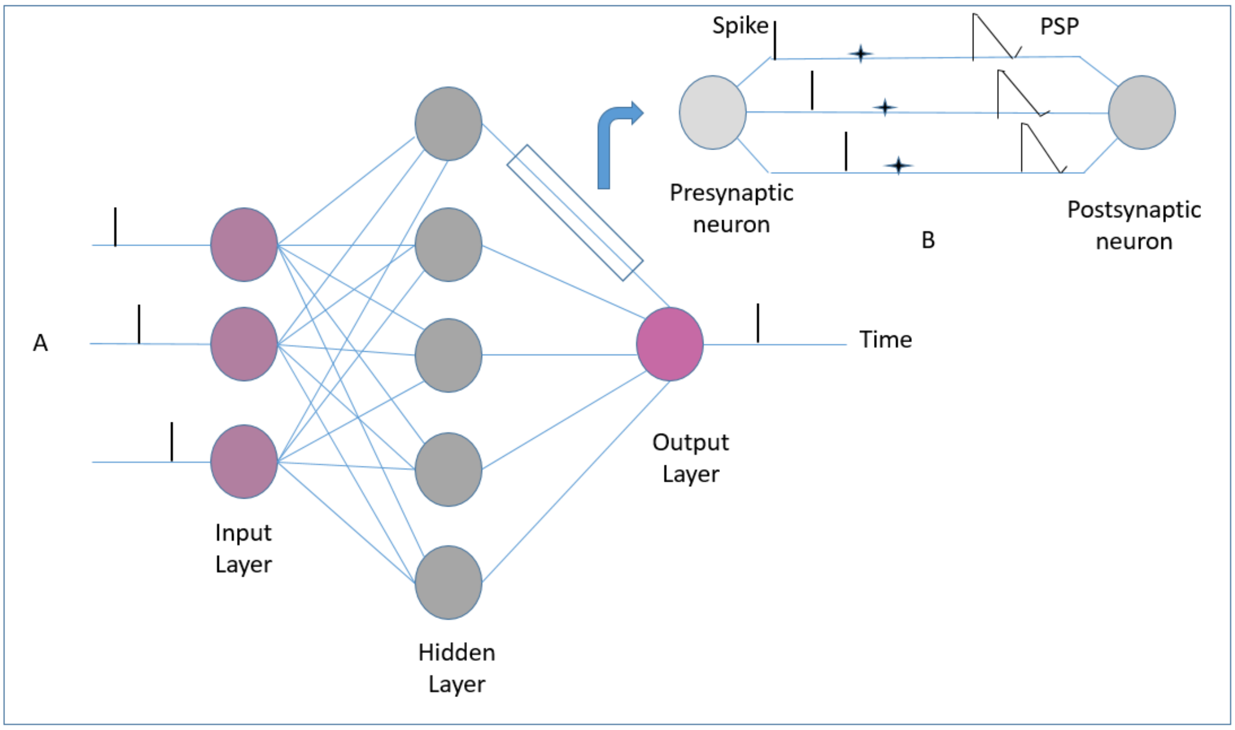

Figure 2. SSNN Architecture: (A) SSNN Feedforward. (B) The links between the presynaptic neuron and postsynaptic neuron.

4. TRAFFIC CLASSIFICATION DATASET

The training traffic dataset is created using VoIP, streaming video, and audio data transfer. The VoIP data originates from HTTP and GTalk apps from the Network Information Management and Security Group (NIMS) dataset, which has around 303,549 traffic flow statistics. The spike neural network processes and evaluates the dataset. The dataset is utilized to enhance the Quality of service (QoS) standards for traffic with elephant and mice flow classes. Data was classified based on packet size, flow duration, and byte count. Mice flow typically last at least ten seconds. Each brief flow often requires less than 15 packets, with each packet containing 500 bytes.

5. THE SPIKING TRAINING ALGORITHM

SNN deals with pulsed information rather than actual data. The initial stage in creating an SNN is to encode analogue input data into spike trains. Thus, Eq. (1) is employed for encoding the input data into spike timings **(Oniz et al., 2013; Al-Yassari and Al-Jamali, 2023)**.

$$t_h^f = T_{max} - round\left(T_{min} + \frac{(R_{in}-R_{min})(T_{max}-T_{min})}{R_{max}-R_{min}}\right) \tag{1}$$

Where T_{max} and T_{min} are the maximum and minimum interval, while R_{max} is a value greater than the highest value in the input and R_{min} is a value less than the smallest value in the input, Rin represents the present actual data. *round* is an operation that gives a rounded value. The next step is the feedforward stage, and it begins with the layer which is the hidden layer. The neurons of hidden are inspected to determine whether they have firing spikes. When a neuron's membrane potential exceeds the threshold value, it only fires spikes once during that time interval. the membrane potential $mp_h(t)$ of the postsynaptic neuron h is estimated using the following equation **(Cui and Xu, 2016)**.

$$mp_h(t) = \sum_{x=1}^{X}\sum_{k=1}^{K} w_{xh}^k(t)\varepsilon(t - t_x^f - d^k) \tag{2}$$

X represents the input neuron number, and K indicates the synapse number connecting two neurons. w_{xh}^k illustrates the synapse weight coefficient for presynapses neurons and postsynapses neurons. t_x^f *is* the time of presynaptic neural firing spike, while d^k represents the synapse delay. $\varepsilon\ (t)$ is the Spike Response Function That is more physiologically realistic. It takes a variety of mathematical representations. The hyperbolic tangential operation is applied using the following Eq. (3).

$$\varepsilon(t) = \begin{cases} 0 & , \quad t \le 0 \\ \tan h\left({}^{t}/_{\tau_s}\right) & , \quad t > 0 \end{cases} \tag{3}$$

The derivative of ε(t) is defined as follows:

$$\frac{\partial\varepsilon}{\partial t} = \frac{1}{\tau_s}\left(1 - tanh^2(t/\tau_s)\right) \tag{4}$$

where τ_s: time decay constant.

After checking the whole neurons of the hidden layer, A similar procedure is followed on the outcome layer. Ultimately, Eq. (5). converts the output spike to actual data as in **(Al-Jamali and Al-Raweshidy, 2020; Al-Jamali and Al-Raweshidy, 2021).**

$$RI\ (t_y^f\ ) = \left(\frac{(R_{in} - t_y^f - R_{min})(R_{max} - R_{min})}{T_{max} - T_{min}} \right) + R_{min} \tag{5}$$

$RI\ (t_y^f\ )$ represents the actual data of the output spike.
Total error is calculated using Mean Square Error (MSE), as illustrated in Eq. (6).

$$MSE = 1/2\ \sum_{y\epsilon Y}(t_y^f - t_y^d)^2 \tag{6}$$

Where y is the number of output neurons, t_y^d indicates the desired spike of output neuron(y) and t_y^f represents the actual spike time for the output neuron Y.
SSNN is trained by updating the weight of each synapse using the backpropagation method (SPIKEPROP)to decrease the MSE value **(Thiruvarudchelvan et al., 2013**). The weights of Synapses among the outcome and hidden neurons are modified using the Eqs. (7 to 9).

$$\delta_y = \frac{t_y^d - t_y^f}{\sum_{h=1}^{H} \sum_{k=1}^{K} w_{hy}^k \frac{\partial}{\partial t} y_h^k} \tag{7}$$

$$\Delta w_{hy}^k = \alpha . \delta_y y_h^k \tag{8}$$

$$w_{hy}^k(t+1) = w_{hy}^k(t) - \Delta w_{hy}^k \tag{9}$$

Where δ_y delta function applies to the outcome layer.

Synapses weights among hidden neurons and the input neuron is adjusted using the Eqs. (10 to 12).

$$\delta_h = \frac{\sum_{h=1}^{H} \delta_y \sum_{k=1}^{K} w_{hy}^k(t) \frac{\partial}{\partial t} y_h^k}{\sum_{x=1}^{X} \sum_{k=1}^{K} w_{xh}^k(t) \frac{\partial}{\partial t} y_x^k} \tag{10}$$

$$\Delta w_{xh}^k = \alpha . \delta_h y_x^k \tag{11}$$

$$w_{xh}^k(t+1) = w_{xh}^k(t) - \Delta w_{xh}^k \tag{12}$$

Where δ_h is the delta function of hidden neurons.

6. SIMULATION RESULTS

Measuring and analyzing the traffic can help reduce congestion. However, before transmitting data, the classifier's effectiveness must be checked using different metrics. SSNN is implemented by using pycharm 2023.3.2 and Tensor Flow and Keras installed on an Intel (R) Core (TM) i7-8550U CPU @ 1.80 GHz 2.40 GHz laptop computer. Jumpy, Pandas, and Matplotlib are also used to process data and graph results. The dataset contains 3,000 traffic flows for two classes, with 70% used for training and 30% for testing. The trained model's performance was evaluated using several matrices **(Nasser and Behadili, 2022; Abdulrezzak and Sabir, 2023; Soud and Al-Jamali, 2023)** indicators for performance include accuracy, precision, recall, and F1 scores.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FN+FP} * 100\% \quad (13)$$

$$\text{Precision} = \frac{TP}{TP+FP} * 100\% \quad (14)$$

$$\text{Recall} = \frac{TP}{TP+FN} * 100\% \quad (15)$$

$$\text{F1-score} = \frac{2*(\text{Recall} * \text{Precision})}{\text{Recall} + \text{Precision}} \quad (16)$$

A single-spike neural network is distinguished by its ability to encode information through the use of time. **Fig. 3** depicts the spike of output neurons. The neuron produces a spike whenever the membrane voltage surpasses the threshold value. Membrane potential then restores to zero.

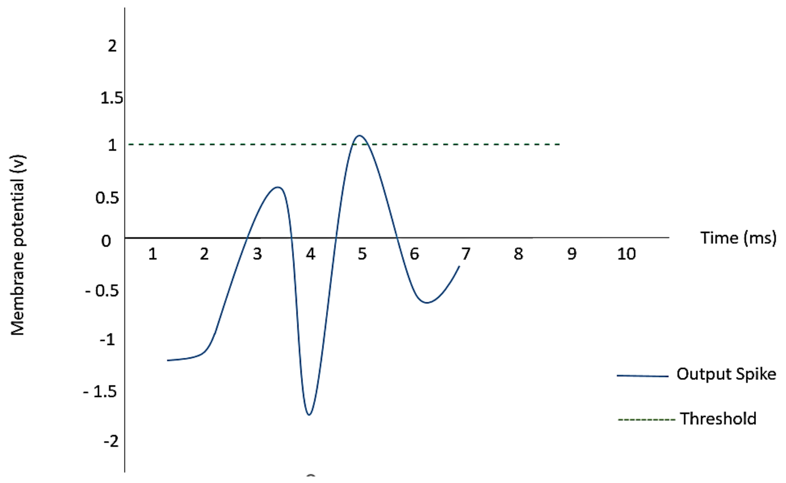

Figure 3. The output neuron spike.

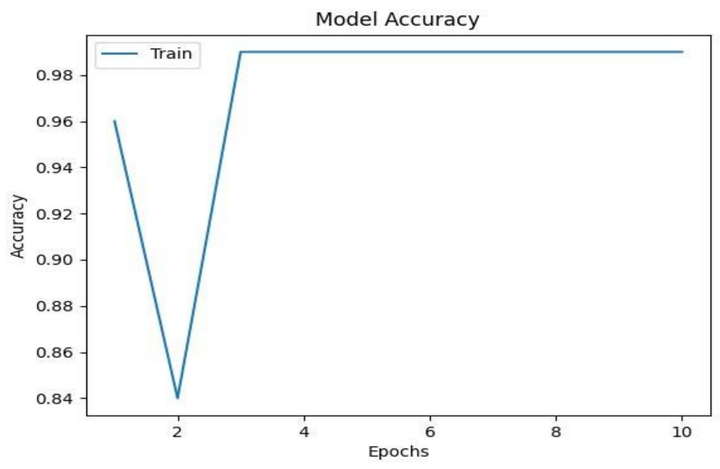

Figure 4. The SSNN Accuracy

Training accuracy grows, reaching 99% on the dataset by epoch 3 as shown in **Fig. 4.** We develop a spike learning model capable of detecting elephant and mice flows. The model achieves 99% training accuracy this might be interpreted as having great and powerful accuracy. However, depending just on the accuracy measure leads to uncertainty and lacks information concerning FP/FN. The recall measure is necessary for understanding FNs, Whereas the precision meter is for comprehension of FP. Consequently, the accuracy and recall ratios are utilized.

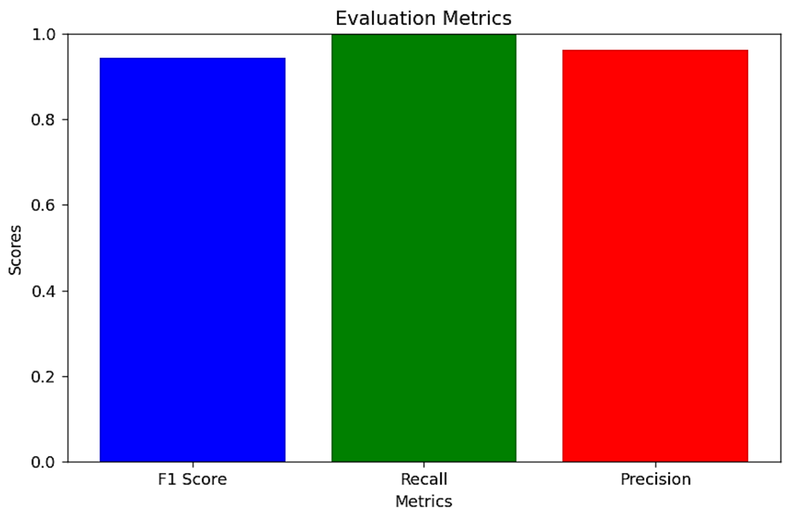

Figure 5. The Evaluation Metrics

Fig. 5 Illustrates the Evaluation Matrices. The high precision score for spike learning of 99.85% implies that the model has a low false positive rate, which means it is effective at not categorizing negative data as positive. The training model's high recall value of 100% implies that it has a low false negative rate, implying that it is effective at catching the majority of positive cases. To balance recall and precision, the F1 Score is frequently utilized. The harmonic mean of precision and recall reveals 99.76%.

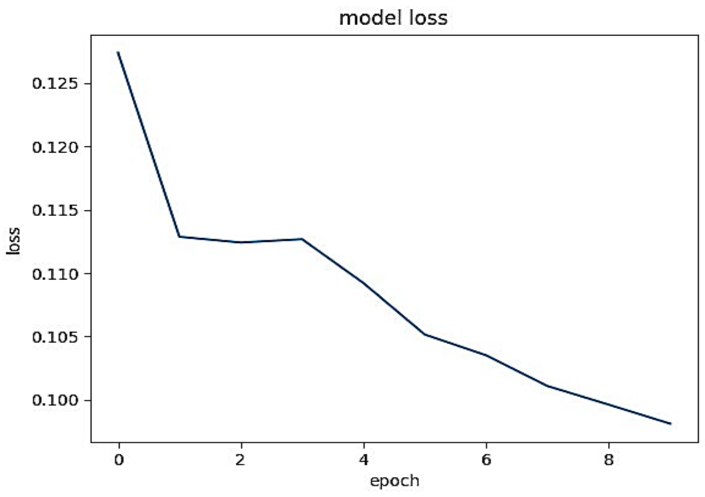

Figure 6. The model loss

The model loss decreased from 0.125 to zero under epoch 9, as shown in **Fig. 6,** which is ideal since it indicates faultless predictions. We lead the model towards improved performance by monitoring and decreasing loss, ensuring it makes accurate predictions while still being able to generalize to new, unknown data.

7. CONCLUSIONS

This article introduces an SDN-based classification system based on flow characterization and identification, known as SSNN. This structure demonstrates the effectiveness of the SNN-supervised learning algorithm, which can classify traffic into two classes. We discovered that network traffic, as characterized by its performance indicators, can be categorized into elephant and mice traffic using these matrices. This improves the management system for networks and adds to the solution of the situation of improper use of network buffers, which causes network congestion and delay due to elephant flows. The suggested algorithm's usefulness was carefully tested in traffic classification tasks utilizing a number of performance indicators. The results confirmed its resilience and effectiveness, with an astounding 99% accuracy rate and a considerable reduction in average error. This high level of precision demonstrates the model's capacity to handle complex classification issues. Furthermore, the algorithm's utility is enhanced by its efficient training approach, which not only provides fast convergence but also consistently produces correct and dependable results. These characteristics demonstrate the model's suitability for real-world applications in traffic analysis and related fields. It is also worth noting that the difficulties we faced were creating an environment closer to a data center. This strategy benefits the SDN system, especially for traffic routing. In future work, the current study will be used by SDN to route heterogeneous traffic based on its type.

Nomenclature

Symbol	Definition	Symbol	Definition
α	learning rate	T_{max}	The maximum interval time
t_y^d	The desired spike time for output neuron	T_{min}	The minimum interval time
t_y^f	Actual spike time for the output neuron	$mp_h(t)$	Membrane potential of postsynaptic neuron
K	Synapses number between two neurons	X	Number of input neurons
h	Number of hidden neurons	$\varepsilon(t)$	Spike Response Function(SRF)
w_{xh}^k	The synapse weight coefficient among presynaptic neurons and postsynaptic neurons	d^k	Represents synapse's delay
$round$	The function that returns a rounded number	δ_y	Delta function of the outcome layer
δ_h	Delta function of hidden layer	$w_{hy}^k(t)$	The synapse weight among the output neuron and hidden neuron
t_x^f	The spike time	Y	The number of output neurons
FP	False Positive	FN	False Negative
TP	True Positive	TN	True Negative

Acknowledgements

The authors of this paper are grateful to the University of Baghdad, College of Engineering, Computer Engineering Department for their assistance.

Credit Authorship Contribution Statement

Sanarya Jamal AL_ Azawee: Writing - originally draft. Nadia Adnan Shiltagh Al-Jamali: Writing - review and modifying, validation, conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

Abdulrezzak, S. and Sabir, F. 2023. An empirical investigation on Snort NIDS versus supervised machine learning classifiers. *Journal of Engineering,* 29(02), pp. 164-178. https://doi.org/10.31026/j.eng.2023.02.11

Abubaker, B.A., Ahmed, S.R., Guron, A.T., Fadhil, M., Algburi, S. and Abdulrahman, B.F., 2023, November. Spiking neural network for enhanced mobile robots' navigation control. In *2023 7th International Symposium on Innovative Approaches in Smart Technologies (ISAS)* (pp. 1-8). IEEE. https://doi.org/10.1109/ISAS60782.2023.10391395

Ali, T.E., Morad, A.H. and Abdala, M.A. 2020. Traffic management inside software-defined data centre networking. *Bulletin of Electrical Engineering and Informatics,* 9(5), pp. 2045-2054. https://doi.org/10.11591/eei.v9i5.1928

Al-Jamali, N.A.S. and Al-Raweshidy, H.S. 2021. Intelligent traffic management and load balance based on spike ISDN-IoT. *IEEE Systems Journal,* 15(2), pp. 1640-1651. https://doi.org/10.1109/JSYST.2020.2996185

Al-Jamali, N.A.S. and Al-Raweshidy, H.S., 2020. Modified Elman spike neural network for identification and control of dynamic system. *IEEE Access*, *8*, pp.61246-61254. https://doi.org/10.1109/ACCESS.2020.2984311

Al-Saadi M, Khan A, Kelefouras V, Walker DJ, Al-Saadi B., 2023. SDN-based routing framework for elephant and mice flows using unsupervised machine learning. *Network* 3(1), pp. 218-238. https://doi.org/10.3390/network3010011

Al-Yassari, M.M.R. and Al-Jamali, N.A.S. 2023. Automatic spike neural technique for slicing bandwidth estimated virtual buffer-size in network environment. *Journal of Engineering,* 29(06), pp. 87-97. https://doi.org/10.31026/j.eng.2023.06.07

Averkin, A. and Yarushev, S. 2019. Averkin, A. and Yarushev, S., 2019. Deep neural networks in digital economy. In *CEUR Workshop Proceedings* (Vol. 2413, pp. 2-8). https://doi.org/17-07-01558

Aymaz, Ş. and Cavdar, T. 2023. Efficient routing by detecting elephant flows with deep learning method in SDN. *Advances in Electrical & Computer Engineering,* 23(3). https://doi.org/10.4316/AECE.2023.03007

Cheng, Y. and Jia, X. 2020. NAMP: Network-aware multipathing in software-defined data center networks. *IEEE/ACM Transactions on Networking,* 28(2), pp. 846-859. https://doi.org/10.1109/TNET.2020.2971587

Cui, C. and Xu, Y.B. 2016. Research on load balance method in SDN. *International Journal of Grid and Distributed Computing,* 9(1), pp. 25-36. https://doi.org/10.14257/ijgdc.2016.9.1.03

Darabseh, A., Al-Ayyoub, M., Jararweh, Y., Benkhelifa, E., Vouk, M. and Rindos, A., 2015, August. SDDC: A software defined datacenter experimental framework. In *2015 3rd International Conference on future internet of Things and Cloud* (pp. 189-194). IEEE.

Ibrahim, L.T., Hassan, R., Ahmad, K., Asat, A.N. and Omar, H., 2016. Online traffic measurement and analysis in big data: Comparative research review. *American Journal of Applied Sciences*, *13*(4), pp.420-431. https://doi.org/10.3844/ajassp.2016.420.431

Ibrahim, Z.I. and Al-Jamali, N.A.S. 2024. Single spike neural network model for superficial environment classification for mobile robot navigation. *Journal of Engineering*, 30(4), pp. 118-133. https://doi.org/ 10.31026/j.eng.2024.04.08

Isyaku, B., Mohd Zahid, M.S., Bte Kamat, M., Abu Bakar, K. and Ghaleb, F.A., 2020. Software defined networking flow table management of openflow switches performance and security challenges: A survey. *Future Internet*, *12*(9), P.147. https://doi.org/10.3390/fi12090147

Kumar, S., Bansal, G. and Shekhawat, V.S., 2020, January. A machine learning approach for traffic flow provisioning in software defined networks. In *2020 International Conference on Information Networking (ICOIN)* (pp. 602-607). IEEE. https://doi.org/10.1109/ICOIN48656.2020.9016529

Li, G., Wang, X. and Zhang, Z., 2019. SDN-based load balancing scheme for multi-controller deployment. *IEEE Access*, *7*, pp.39612-39622. https://doi.org/10.1109/ACCESS.2019.2906683

Lin, C.Y., Chen, C., Chang, J.W. and Chu, Y.H., 2014, December. Elephant flow detection in datacenters using openflow-based hierarchical statistics pulling. In *2014 IEEE Global Communications Conference* (pp. 2264-2269). IEEE. https://doi.org/10.1109/GLOCOM.2014.7037145

Liu, W.X., Cai, J., Chen, Q.C. and Wang, Y. 2021. DRL-R: Deep reinforcement learning approach for intelligent routing in software-defined data-center networks*. Journal of Network and Computer Applications*, 177, P. 102865. https://doi.org/10.1016/j.jnca.2020.102865

Liu, Y., Cao, K., Wang, R., Tian, M. and Xie, Y., 2022. Hyperspectral image classification of brain-inspired spiking neural network based on attention mechanism. *IEEE Geoscience and Remote Sensing Letters*, *19*, pp.1-5. https://doi.org/10.1109/GLOCOM.2014.7037145

Mendiola, A., Astorga, J., Jacob, E. and Higuero, M. 2016. A survey on the contributions of software-defined networking to traffic engineering. *IEEE Communications Surveys & Tutorials*, 19(2), pp. 918-953. https://doi.org/10.1109/COMST.2016.2633579

Modi, T.M. and Swain, P. 2023. Enhanced routing using recurrent neural networks in software defined-data center network. *Concurrency and Computation: Practice and Experience*, 35(5). P. e7557. https://doi.org/10.1002/cpe.7557

Nasser, F.K. and Behadili, S.F. 2022. Breast cancer detection using decision tree and k-nearest neighbour classifiers. *Iraqi Journal of Science, pp.* 4987-5003. https://doi.org/10.24996/ijs.2022.63.11.34

Oniz, Y., Kaynak, O. and Abiyev, R., 2013, February. Spiking neural networks for the control of a servo system. In *2013 IEEE International Conference on Mechatronics (ICM)* (pp. 94-98). IEEE. https://doi.org/10.1109/ICMECH.2013.6518517

Priyadarsini, M. and Bera, P. 2021. Software defined networking architecture, traffic management, security, and placement: A survey. *Computer Networks*, 192, P. 108047. https://doi.org/10.1016/j.comnet.2021.108047

Roy, A., Venkataramani, S., Gala, N., Sen, S., Veezhinathan, K. and Raghunathan, A., 2017, July. A programmable event-driven architecture for evaluating spiking neural networks. In *2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)* (pp. 1-6). IEEE. https://doi.org/10.1109/ISLPED.2017.8009176

Shears, O. and Yazdani, A. 2020. Spiking neural networks for image classification. *Advanced Machine Learning,* 11, pp. 1-17. https://doi.org/10.13140/RG.2.2.27001.80486

Shihab, Z. I. 2016. *Practical Adoption of Modified Spike Neural Network for Indoor Mobile Robot Navigation. M.Sc. Thesis, Computer Engineering, University of Baghdad.*

Shirali-Shahreza, S. and Ganjali, Y. 2018. Delayed installation and expedited eviction: An alternative approach to reduce flow table occupancy in SDN switches. *IEEE/ACM Transactions on Networking*, 26(4), pp. 1547-1561. https://doi.org/10.1109/TNET.2018.2841397

Soud, N.S. and Al-Jamali, N.A.S. 2023. Intelligent congestion control of 5G traffic in SDN using dual-spike neural network. *Journal of Engineering,* 29(1), pp. 110-127. https://doi.org/10.31026/j.eng.2023.01.07

Thiruvarudchelvan, V., Crane, J.W. and Bossomaier, T., 2013, April. Analysis of SpikeProp convergence with alternative spike response functions. In *2013 IEEE Symposium on Foundations of Computational Intelligence (FOCI)*, pp. 98-105. https://doi.org/10.1109/FOCI.2013.6602461

Xu, Y., Yang, J. and Zhong, S. 2013. An online supervised learning method based on gradient descent for spiking neurons. *Neural Networks* 93 7-20. https://doi.org/10.1016/j.neunet.2013.04.010

Wang, N., Ho, K.H., Pavlou, G. and Howarth, M. 2008. An overview of routing optimization for internet traffic engineering. *IEEE Communications Surveys & Tutorials* 10(1), pp. 36-56. https://doi.org/10.1109/COMST.2008.4483669

Yusuf, M.N., Bakar, K.B.A., Isyaku, B., Osman, A.H., Nasser, M. and Elhaj, F.A., 2023. Adaptive path selection algorithm with flow classification for software-defined networks. *Mathematics*, *11*(6), p.1404. https://doi.org/10.3390/math11061404

Yusuf, M.N., bin Abu Bakar, K., Isyaku, B. and Saheed, A.L., 2023. Review of path selection algorithms with link quality and critical switch aware for heterogeneous traffic in SDN. *International journal of Electrical and Computer Engineering Systems*, *14*(3), pp. 345-370. https://doi.org/10.32985/ijeces.14.3.12

Zaher, M., Alawadi, A.H. and Molnár, S. 2021. Sieve: A flow scheduling framework in SDN based data center networks. *Computer Communications,* 171, pp. 99-111. https://doi.org/10.1016/j.comcom.2021.02.013

التعامل مع حركة المرور غير المتجانسة لشبكة مركز البيانات المحددة بالبرمجيات باستخدام شبكة السبايك العصبية

سناريا جمال ابراهيم*، نادية عدنان شلتاغ الجمالي

قسم هندسة الحاسبات، كلية الهندسة، جامعة بغداد.بغداد.العراق

الخلاصة

تسمح الشبكات المحددة بالبرمجيات بإدارة شبكة أكثر مرونة من الشبكات التقليدية. تدير الشبكات المحددة بالبرمجيات تدفقات البيانات بكفاءة وتحسن موارد الشبكة. ومع ذلك، فإن التباين في الترافك يؤثر على احتياجات جودة الخدمة ومتطلبات موارد الشبكة. حيث انه يتصرف بشكل مختلف عند الذهاب إلى الوجهة الخاصة به. حاليًا، تكافح العديد من شبكات مراكز البيانات من الاستخدام الغير العادل لموارد الشبكة المختلفة من خلال الحزم الكبيرة (تدفقات الأفيال) التي تصل أثناء أي لحظة و تؤثر على تدفقات محددة (تدفق الفئران). تمثل تدفقات الأفيال نسبة صغيرة فقط من إجمالي حركة المرور ومع ذلك، فهي تعتبر طويلة الأمد وغالبًا ما تستهلك موارد الشبكة. يتسبب وجودها في الازدحام والتأخير في معظم تدفقات الفئران . يعد التنبؤ بحركة المرور وتصنيفها أمرًا ضروريًا للاستخدام الأمثل للموارد وتوفير جودة الخدمة والحد من ازدحام الشبكة والتأخير.

تقترح هذه الورقة نهج التعلم الخاضع للإشراف لشبكة الجيل الثالث (شبكه السبايك العصبية) باستخدام الترميز الزمني لتحديد حركة المرور غير المتجانسة. يستخدم نهج التصنيف ثلاث ميزات: وقت التدفق ومعدل البايت ومعدل الحزمة. ثم يتم تعليم تصنيف الترافك إلى فئتين. يصنف هذا التدريب نوعين من الترافك: تدفق الفيلة والفئران. تم فحص جودة الخوارزمية عند تصنيف حركة المرور باستخدام العديد من المقاييس وتم إثبات كفاءتها حيث كانت قادرة على تقليل الخطأ المتوسط وبلغت دقتها 99٪. يتم إثبات فائدة النموذج المقترح من خلال إجراء التدريب الفعال الخاص به، والذي يوفر نتائج سريعة ودقيقة.

الكلمات المفتاحية: حركة مرور غير متجانسة، شبكات محددة بالبرمجيات، تدفقات الفيلة، تدفقات الفئران