I.J. Intelligent Systems and Applications, **2024, 6, 73-83**
Published Online on December 8, 2024 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2024.06.04

Deep Learning Based Traffic Management in Knowledge Defined Network

Tejas M. Modi*
Department of Computer Science and Engineering, Adani University, Ahmedabad, Gujarat, India
E-mail: dr.tejasmmodi@gmail.com
ORCID iD: https://orcid.org/0000-0003-3326-4734
*Corresponding author

Kuna Venkateswararao
Department of Computer Science and Engineering, GITAM (Deemed to be University), Visakhapatnam, Andhra Pradesh, India
E-mail: kvrvenkateshkvr@gmail.com
ORCID iD: https://orcid.org/0000-0001-6336-7297

Pravati Swain
Department of Computer Science and Engineering, National Institute of Technology Goa, South Goa, Goa, India
E-mail: pravati@nitgoa.ac.in
ORCID iD: https://orcid.org/0000-0001-6794-0919

Received: 23 April 2024; Revised: 10 August 2024; Accepted: 23 September 2024; Published: 08 December 2024

Abstract: In recent Artificial Intelligence developments, large datasets as knowledge are a prime requirement for analysis and prediction. To manage the knowledge of the network, the Data Center Network (DCN) has been considered a global data storage facility on edge servers and cloud servers. In recent research trends, knowledge-defined networking (KDN) architecture is considered, where the management plane works as the knowledge plane. The major network management task in the DCN is to control traffic congestion. To improve network management, i.e., optimized resource management, enhanced Quality of Service (QoS), we propose a path prediction technique by combining the convolution layer with the RNN deep learning model, i.e., Convolution-Long short-term memory network as Convolution-LSTM and the bi-directional long short-term memory (BiLSTM) network as Convolution-BiLSTM. The experimental results demonstrate that, in terms of many metrics, i.e., network latency, packet loss ratio, network throughput, and overhead, our proposed methodologies perform better than the existing works, i.e., OSPF, FlowDCN, modified discrete PSO, ANN, CNN, and LSTM-based routing approaches. The proposed approach improves the network throughput by approximately 30% and 12% as compared to existing CNN and LSTM-based routing approaches, respectively.

Index Terms: Knowledge Defined Network (KDN), Data Center Network (DCN), Convolutional Layer, Recurrent Neural Network (RNN), LSTM, BiLSTM.

1. Introduction

Nowadays, scalable and highly flexible networking mechanisms are in demand to improve the performance of different network applications. Currently, knowledge-defined networking (KDN) [1] is an emerging networking paradigm that considers deep learning-based solutions to enhance network architecture and performance. The KDN provides a flexible and agile network architecture by splitting the control plane from the data plane. The detached control plane enhances network management by using a central controller in the network topology. The controller collects details of networking devices from the data plane through the OpenFlow [2] southbound interface. The knowledge plane enhances the intelligence in the KDN network by using different analytical and deep learning approaches. The knowledge plane establishes the connection to the control plane through the northbound interface [3].

In recent research trends, the Fat-Tree Data Center Network (DCN) topology has been widely used to enhance network performance. The DCN topology provides multiple routing paths to enhance communication between source-destination pairs. KDN-based DCN (KD-DCN) provides a central controller to manage the load balancing, flow management, and routing in the DCN network environment. To manage dynamic traffic changes in the KD-DCN

network, an agile and intelligent mechanism is required to reduce the congestion in the network.

Currently, conventional deep learning (DL) approaches, i.e., deep neural networks and deep reinforcement learning, are proposed to manage the KDN network [4, 5]. However, these conventional approaches provide limited intelligence and a complex training process to handle dynamic traffic changes. As a result, in the KDN network, conventional approaches are inadequate to improve average network throughput in high network traffic. Moreover, the KDN provides optimal solutions in different application domains, i.e., energy distribution [6], traffic management [7], and routing management [8].

Motivation and Contribution

In this paper, we present several convolution-RNN deep learning-based intelligent routing path selection mechanisms that are deployed at the central controller to improve the network performance of the KD-DCN architecture. The convolution-RNN models uncover the long-term relationships between congestion features and determine the key congestion features using the convolution layer to extract features. Here, Convolution-LSTM and Convolution-BiLSTM are two deep learning techniques demonstrated in the proposed mechanism. Additionally, the system works in two modules: dataset pre-processing and the routing stage. During the dataset pre-processing stage, traditional routing methods are used to collect the initial dataset from networking devices. The collected dataset is pre-processed for the training dataset in the subsequent routing stage. The controller first runs the convolution layer during the routing phase, and the output it produces serves as the input for both RNN variant deep learning models. The future routing decision is based on the output of deep learning results which uses the training weights of the routing stage as input and offers the best routing paths in the KD-DCN network.

Additionally, the proposed system considers an average network delay of *D* to execute the online learning process. Here, the continuous network statistics measurement technique is considered to collect *D*. Next, the *D* is available as the input for the online learning mechanism. Compared to periodic training, online learning continuously assesses *D*, which improves the network's performance. Moreover, the experimental setup of the presented deep learning models and performance is contrasted with those of conventional Open Shortest Path First (OSPF), FlowDCN [9], Modified discrete PSO [10], and existing deep learning techniques like ANN [11], CNN [12], and LSTM [13] in KD-DCN architecture. The main contributions of the proposed work are mentioned in the following manner:

- A convolution-RNN model-based approach is presented to overcome the limitations of existing approaches, i.e., OSPF, FlowDCN, modified discrete PSO, ANN, CNN, and LSTM deep learning-based routing approaches, in KD- DCN. Integrating an effective routing path with the historical traffic dataset improves the efficiency of the KDN controller.
- Online learning of models incorporates the *D* values to discover the interconnections of real-time traffic data. Compared to the standard model learning approach, this methodology boosts the learning process.
- The models considered in the proposed system utilize real-time D for the model learning phase instead of binary link weights.
- Several network performance matrices, such as network latency, packet loss ratio, throughput, and execution time, are examined in the performance evaluations. As compared to existing state-of-the-art routing approaches, it has been discovered that the evaluation results of the proposed approach are improved.
- The proposed system can be integrated into Edge Computing and 5G cellular architecture where Knowledge Defined Network-based architecture can improve routing and resource management.

The Manuscript is divided into five major sections, i.e., Introduction, Related Work, System Model, Evaluation Results, and Conclusion. The Section 1 Introduction briefly overviews the defined network and problem statement. The Section 2 related work provides limitations of existing mechanisms in the domain. Next, based on constraints, the Section 3system model presents the working mechanism of the proposed system. The Section 4 evaluation results present the result analysis of the proposed system with existing systems. Finally, the Section 5 Conclusion concludes the manuscript.

2. Related Works

To boost the accuracy and efficiency of any system performance, machine, and deep learning architectures are available that provide a multi-tier training technique that is applied to describe the complicated connection between input and output [14]. The literature survey is conducted for different existing applications and systems. Here, based on the existing work, the literature survey section is divided into two subsections: 1) Machine and Deep Learning-Based Systems; and 2) Research Gaps.

2.1. Machine and Deep Learning Based Systems

In the paper [15], different machine-learning techniques are applied in the domain of SDN with issues and challenges. The improvised procedure of machine learning is known as deep learning, which delivers more intelligent and effective outcomes for tasks such as network management, network control, and security [16].

The Artificial Neural Network (ANN)-based KDN is proposed in the paper [11] for star and ring network topologies. The results show that this approach improves the network's learning rate compared to traditional approaches. In paper [17], ANN-based load balancing is proposed for KD-DCN topology. The results indicate that this approach outperformed ECMP load balancing.

The machine learning (ML)-based approach in KDN is proposed in paper [18]. The results show that this approach shrinks the average network delay in the KDN network. The Deep Neural Network (DNN)-based approaches are proposed in papers [4, 5]. In paper [4], a deep neural network (DNN)-based approach is proposed for KD-DCN topology. The results show that this approach obtains high-level knowledge in terms of system architecture and operation procedures. Similarly, in paper [5], a self-organized DNN-based approach is proposed to reduce KDN network overhead.

The Deep Reinforcement Learning (DRL)-based approaches in KDN are proposed in papers [19, 20]. In papers [19, 20], DRL-based routing is proposed for random network topology. The results show that these approaches outperform the traditional routing mechanisms. However, DRL approaches proposed in papers [19, 20] are more time-consuming mechanisms to discover actions and rewards during the learning process.

2.2. Research Gaps

The literature presented is based on different deep learning approaches, i.e., ANN, DNN, and DRL, based on routing strategies introduced in KDN. The previous studies were unable to extract traffic feature dependencies from the periodic traffic dataset [21]. Moreover, the existing systems consider the output of the model learning process as input into the OSPF routing for path selection. Also, in the literature, small network topologies and communication for a single pair of source- destination is considered. Thus, in these scenarios, the previous studies were unable to prove their performance in high-traffic congestion situations.

Hence, to overcome the above limitations, the Convolution-RNN (Convolution-LSTM and Convolution-BiLSTM) models are considered in the proposed mechanism to discover multiple characteristics, i.e., congestion features and feature dependencies, from the collected dataset. Additionally, the online learning of Convolution-RNN models is considered by considering the continuous average network delay D in the network. Moreover, the FlowDCN is integrated to utilize the output of the Convolution-RNN models, which improves the path prediction process in the network. Also, the proposed system is implemented for the KD-DCN topology, which has 48 communication links and multiple source-destination pairs to handle high-traffic congestion scenarios. Furthermore, the proposed mechanism is analyzed for multiple performance metrics.

3. System Model

In this section, three sub-sections, i.e., system architecture, convolution-RNN model learning, and proposed system, are presented to explain the proposed work.

3.1. System Architecture

The proposed system considers Convolution-RNN models to improve the network management task in the Knowledge Defined-Data Center Network (KD-DCN) architecture. In KD-DCN, the data center networks can be programmed using a centralized controller.

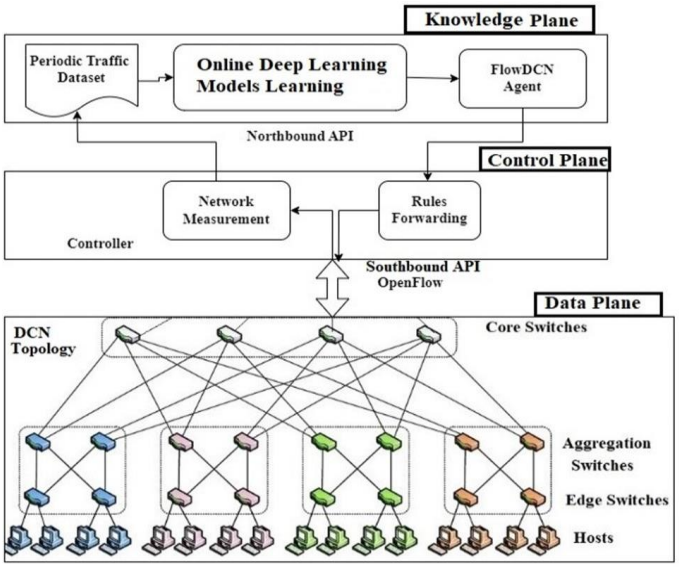

Fig.1. KD-DCN architecture

Fig. 1 represents the KD-DCN system architecture, which is a combination of multiple planes, i.e., data, control, and knowledge. Additionally, two interfaces are available in the architecture: southbound and northbound. The data plane contains pod 4 DCN topology, which consists of 16 hosts and 20 switches such as core, aggregation, and edge switches. Moreover, in this architecture, four switches are combined into one cluster, which is known as a pod. The northbound interface is managed by the RYU controller. Moreover, the controller manages the logical connections with DCN topology in the data plane through the OpenFlow interface [22].

In the knowledge plane, the controller integrates Convolution-LSTM or Convolution-BiLSTM deep learning agents to generate training datasets through the model learning process. Periodically, the controller provides traffic data to the Convolution-LSTM or Convolution-BiLSTM model, which extracts the traffic patterns and generates new link weights. Next, the FlowDCN algorithm is used to route traffic in the KD-DCN topology. In addition to online training, the deep learning models continuously train the traffic dataset based on the current value of D, which is the average network delay.

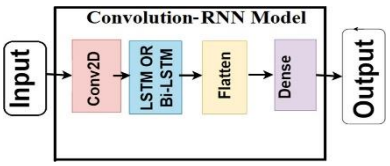

Fig.2. Convolution-RNN model learning process

3.2. Convolution-RNN Model Learning

This section gives details on convolution-RNN models. Fig. 2 presents the architecture and detailed layers of Convolution- RNN models. Initially, the convolution layer considers the traffic dataset *TM* and different link weights *LW* to analyze the traffic patterns in the network. The convolution layer extracts the congestion-related features. The extracted features reduce the complexity of RNN models and improve feature dependency extraction through RNN models. Finally, RNN models generate the prediction through the learning process.

3.3. Proposed System

This section presents the working mechanism of the proposed work. Fig. 3 represents the proposed system flowchart, which presents a combination of two phases, i.e., dataset pre-processing and routing stage. To analyze the communication between different source-destination pairs, the network monitor first gathers network data using the OSPF routing protocol. The *TM* and *LW* are generated as the output of the data pre-processing stage, which will be employed in later deep learning model training. The proposed deep learning models are trained to recognize changes in the network environment in real-time during the routing stage. The newly modified link weights LW_u are gathered as an output of the training process, which is utilized to generate new communications paths for upcoming routing in the KD-DCN topology by considering the FlowDCN. This process is considered for initial training purposes. Next, in the following Algorithm 1, the proposed system is explained through pseudo-code.

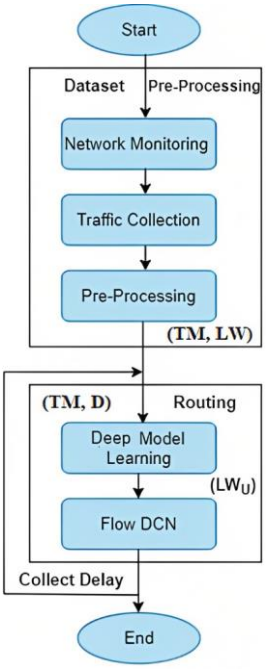

Fig.3. Proposed system flowchart

Initially, the controller starts the network monitoring process for individual monitoring intervals t_u and logs the switch and interface status. In addition to its core function, the controller also develops packet forwarding rules in real-time network topology. By following the packet forwarding rules of the data plane, the controller constructs routing pathways for certain source-destination pairs.

The initial step in the proposed mechanism is to extract crucial traffic details from the dataset by using the controller. The result is the creation of the periodic traffic dataset TM, which has three main components, i.e., datapath, available link bandwidth, and received packets. Next, the link weights LW are computed using the Equation (1). Here, for each unique switch, TP represents the number of packets sent, and RP represents the number of packets received in each datapath in the above equation. The total number of packets TRP are received over the network during the t_u monitoring interval. Furthermore, the proposed deep learning models use a set of link weights LW as a one-dimensional input.

$$LW = \frac{1+(RP-TP)}{TRP} \tag{1}$$

Algorithm 1 is presented to explain the proposed routing strategy. The TM and LW are generated from the pre-processing of the dataset. The TM is a combination of multiple datapaths, the number of packets received in each datapath, and the availability of link bandwidth. The convolution-RNN models consider TM and LW as input met- rics. For each routing stage, unique TM and LW are considered. The convolution layer gains the correlation between congestion-related features of the KD-DCN network.

Next, the RNN deep learning models learn the fundamentals of topology, routing routes, and packet forwarding during training. The end outcome of the learning procedure is the newly modified set of link weights, i.e., the LW_u value. The KD-DCN architecture also makes use of FlowDCN and considers LW_u as input. This process is considered for the initial 10 recording intervals t_u10. The initial process collects the average network delay D from the network and forwards it to the next online model learning process.

The online model learning process uses knowledge of the current network environment for a better training process, i.e., average network delay D. The whole scenario offers an online model learning process that considers current network statistics instead of prior link weights. The online model learning stage generates a newly updated set of link weights. Due to the constant consideration of the average network delay D for the deep learning models, the online model learning process requires more processing time. Here, an updated set of link weights LW_u takes into account both deep learning models in the following routing stage when the online training is finished.

Algorithm 1 Proposed Routing Algorithm

Input: (TM, LW)
Output: LW_u
0: **Initial Model Learning Process.**
0: **for** t_u1 ,t_u2 ,. ,t_u10 in t_u **do**
0:Execute Convolution-RNN model learning by considering (TM,LW).
0:Each t_u collect LW_u.
0:**for** Every Datapath take LW_u **do**
0: Implement FlowDCN.
0:**end for**
0:Collect D.
0: **end for**
0: **Online Model Learning Process.**
0: **for** Every t_u11 ,t_u12 ,..,t_ui ,..,t_un in t_u **do**
0:Models Learning of Convolution-RNN using (TM, D).
0:Collect LW_u.
0:**for** Every Datapath take LW_u **do**
0: Implement FlowDCN.
0:**end for**
0:Collect (TM, D) for next online model learning process.
0: **end for**=0

4. Evaluation Results

This section is divided into four subsections, i.e., simulation setup, performance parameters, network performance analysis, execution time analysis, and error analysis.

4.1. Simulation Setup

The simulation-based approach is used to implement the proposed work. The Ubuntu 16.04 LTS is installed on an Intel Core i9 processor-based workstation with 16 GB of RAM and a 4 GB GPU. Moreover, the RYU controller is taken into consideration for simulation. The KD-DCN topology is implemented using the Mininet and iPerf traffic analyzers. Each host has iPerf incorporated within it, and each host can individually activate iPerf. The different Python programming libraries, such as Tensorflow and Keras, are considered to execute the Convolution-RNN models. Here, a link bandwidth of 50 Mbps is considered optimal. The experimental analysis is conducted for multiple source-destination pairs in the topology. Moreover, the RYU controller is connected to the KD-DCN topology through the OpenFlow southbound API.

4.2. Performance Parameters

In the Evaluation Results, five different performance parameters are considered, i.e., network latency, packet loss ratio, network throughput, execution time analysis, and error analysis. All the performance parameters are compared with a total of six existing systems. Moreover, the QoS parameters (network latency, packet loss ratio, network throughput) are analyzed for different traffic intensities and different simulation times. Additionally, the overhead analysis is discussed as compared to existing systems. Furthermore, the execution time analysis and error analysis are presented for four different datasets.

4.3. Network Performance Analysis

In this section, the system performance is evaluated for the different metrics, such as network latency, packet loss ratio, throughput, and overhead. Moreover, these performance parameters are measured for different simulation times and network traffic loads.

For up to 120 seconds, the deep learning model-based proposed solution is simulated. Simulation time is a crucial factor to consider when analyzing the impact of proposed deep learning models on network performance. Data communication between several source-destination pairings is examined in each period, i.e., 20 seconds. The acquired network performance results are compared with modified discrete PSO, FlowDCN, OSPF, and deep learning models, i.e., ANN, CNN, and LSTM. It has been noted that the proposed system improves upon the fixed routing path allocation's drawbacks. The two key factors in a network environment that affect performance are network latency and packet loss ratio. The average network latency and packet loss ratio are depicted in Fig. 4 and Fig. 5, respectively. At simulation time t = 20s, it is seen that the average latency and loss ratio are almost identical as compared to existing approaches.

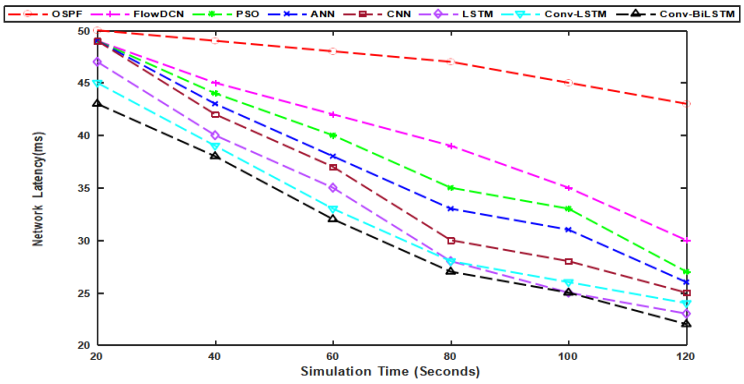

Fig.4. Network latency vs time

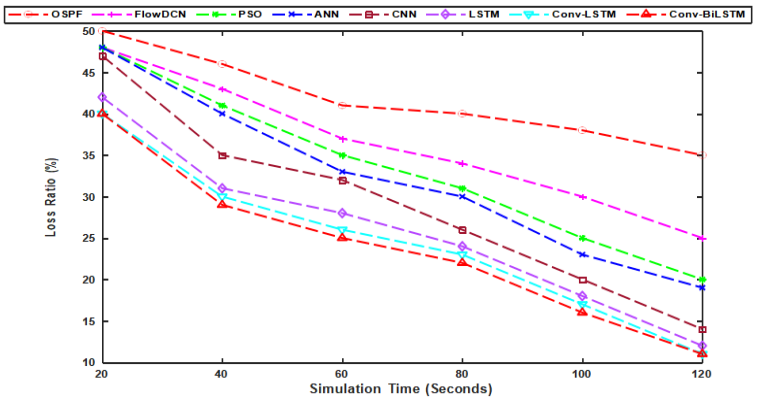

Fig.5. Packet loss ratio vs time

The network latency and packet loss ratio drop after 40 seconds of simulation time. Regarding the network latency and packet loss ratio, the proposed models outperform the traditional routing protocols and existing deep learning models. It has been observed that ongoing training has caused the accuracy of proposed deep learning models to steadily grow.

Similarly, the proposed work boosts the network throughput as compared to OSPF and FlowDCN by around 80% as illustrated in Fig. 6. Similarly, as compared to modified discrete PSO and ANN, the proposed work enhances the network throughput by 35%, approximately. Moreover, the proposed system boosts the network throughput by about 30% in comparison to existing CNN deep learning models. Additionally, the proposed system boosts the network throughput by about 12% in comparison to the LSTM-based routing strategy. Thus, the convolution-RNN-based proposed system boosts network throughput with incremental training.

The primary benefits of the proposed work are the knowledge gained about the network environment and the regular model training utilizing various types of network traffic. Additionally, in the proposed work, RNN deep learning models use forget gates to examine the forward and backward traffic histories, while the convolution layer extracts co-relationships between congestion features. Therefore, in contrast to OSPF, FlowDCN, modified discrete PSO, ANN, CNN, and LSTM deep learning models, the proposed work provides effective and better results in high traffic congestion with increased network performance.

The network performance mainly depends on the current network statistics, i.e., traffic load and available bandwidth. To measure network performance, i.e., network latency, packet loss ratio, and network throughput, different traffic loads are analyzed in the proposed work. In the evaluation process, the network load is considered in different values, i.e., 20%, 30%, 40%, 50%, and 60%.

The network latency varies with different network traffic loads. As shown in Fig. 7, in the very low network traffic load compared to OSPF, FlowDCN, modified discrete PSO, and ANN, the proposed work reduces the network latency by approximately 25%. Similarly, as compared to CNN and LSTM-based approaches, the proposed work reduces the network latency by approximately 15% and 5%, respectively. Similarly, as shown in Fig. 8, the proposed work reduces the packet loss rate by around 30% as compared to OSPF, FlowDCN, Modified discrete PSO, and ANN. Moreover, as compared to CNN and LSTM-based approaches, the proposed work reduces the loss ratio by around 10% and 5%, respectively. Additionally, in high network traffic load scenarios, the proposed work outperforms all the existing work and reduces the packet loss ratio in the KD-DCN topology.

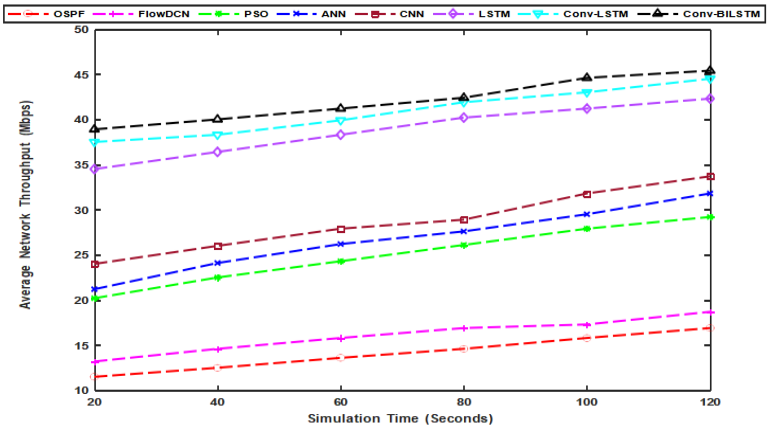

Fig.6. Network throughput vs time

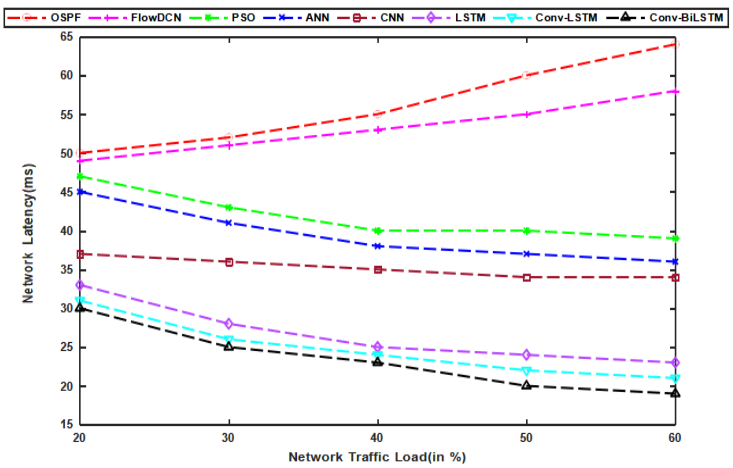

Fig.7. Network latency vs traffic

The network throughput is a major performance parameter to analyze the proposed work. The network throughput mainly depends on the current network load in the topology. If the network traffic load and packet loss ratio are high then the network throughput of systems gets lower. However, as shown in Fig. 9, in the proposed work, the system is efficient in handling the large network traffic and increases the network throughput by 20% as compared to OSPF, FlowDCN, Modified discrete PSO, and ANN. The proposed work increases the network throughput by 20% and 8% as compared to CNN and LSTM-based approaches, in the high network traffic load.

Thus, Fig. 4 to Fig. 9 is presented for the result analysis of the proposed work with six different existing systems, i.e., OSPF, FlowDCN, PSO, ANN, CNN, and LSTM. Here, Fig. 4 to Fig. 6 is presented for analysis of network latency, packet loss ratio, and network throughput during different simulation periods. Similarly, Fig. 7 to Fig. 9 is presented for analysis of network latency, packet loss ratio, and network throughput for different network traffic intensities. The proposed work efficiently handles the high network traffic load and improves the KD-DCN network performance. Next, the execution time analysis is presented for the proposed work and compared with the existing work.

Next, the routing overhead is presented in Fig. 10. The routing overhead provides information about the additional bandwidth required to manage the communication between the controller and switches in the data plane. As compared to existing mechanisms, the proposed work requires a low amount of network bandwidth to manage network communication. The major advantage of the proposed work is the less amount of communication interaction between the central controller and data plane devices. The prime reason for this situation is the high level of network intelligence provided by the model learning process of convolution-RNN models.

4.4. Execution Time Analysis

The performance evaluation and operational constraints of our proposed work are described in this section. Deep learning is an innovative paradigm that makes use of a lot of processing time to enhance network performance. The training of a deep learning model with multiple layers and large input data requires a significant amount of time. Deep learning models initially only need a small amount of computation time, but as the input data volume grows, they require more computation time for training the models.

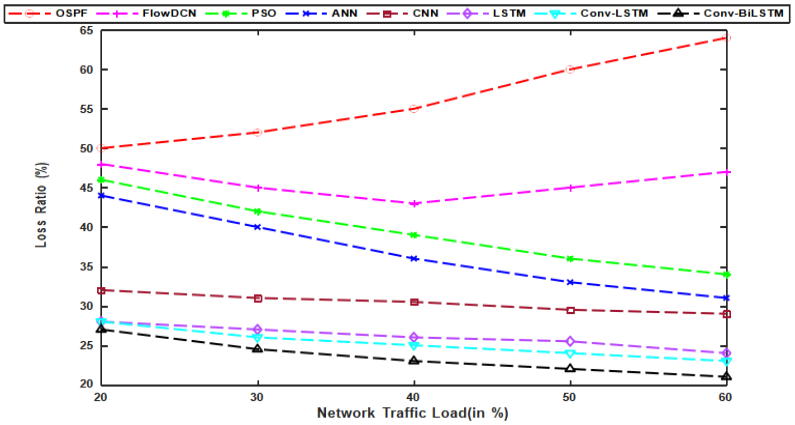

Fig.8. Packet loss ratio vs traffic

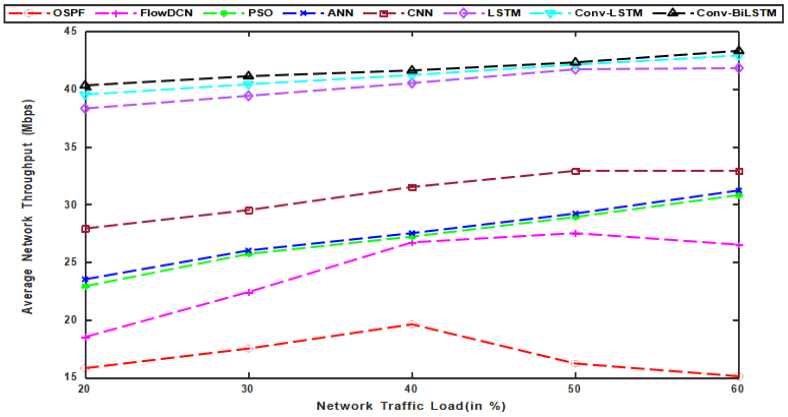

Fig.9. Network throughput vs traffic table 1. execution time analysis

Table 1. Execution time analysis

Traffic Load **(in MB)**	ANN **Routing**	CNN **Routing**	LSTM **Routing**	Convolution- LSTM **Routing**	Convolution-BiLSTM **Routing**
20	20.1 Seconds	20.3 Seconds	20.6 Seconds	20.7 Seconds	21 Seconds
40	40.4 Seconds	40.6 Seconds	40.8 Seconds	41.1 Seconds	41.3 Seconds
60	60.5 Seconds	60.8 Seconds	61.2 Seconds	61.6 Seconds	61.7 Seconds
90	80.6 Seconds	80.9 Seconds	81.5 Seconds	81.7 Seconds	82 Seconds
110	100.3 Seconds	100.9 Seconds	101.7 Seconds	102 Seconds	102.2 Seconds
135	120.7 Seconds	121.1 Seconds	121.9 Seconds	122.1 Seconds	122.3 Seconds

According to Table 1, the Convolution-RNN models require additional execution time to complete the pre-processing of the dataset, the model learning (training and testing), and the routing strategy. The Convolution-LSTM deep learning model needs a few more milliseconds to execute the routing strategy as compared to ANN, CNN, and LSTM model-based strategies. Convolution-RNN deep learning models, therefore, require more processing time for execution when the size of the input dataset grows as compared to the initial traffic dataset. Thus, the Convolution-RNN-based proposed solution offers better real-time routing with the aid of ongoing network monitoring for a minimal amount of additional time. Additionally, the major motivation is to analyze the model learning (training and testing) time concerning different traffic loads.

4.5. Error Analysis

Convolution-RNN models are employed in the proposed work, and they take periodic datasets into account as input for the ongoing learning process. Initially, the primary dataset, or Dataset t1, is the periodic traffic dataset that the models collect in the recording interval t_u. In the presented work, several recording intervals, $n \times t_u$, where n offers recording intervals to collect various periodic datasets, are studied. Here, every $n \times t_u$ generates n different datasets. Four different datasets are utilized to examine the model's learning process. Hyperparameters tuning, such as epochs number of convolution layers, number of LSTM/BiLSTM Layers, and learning rate, are taken into consideration while presenting the error analysis.

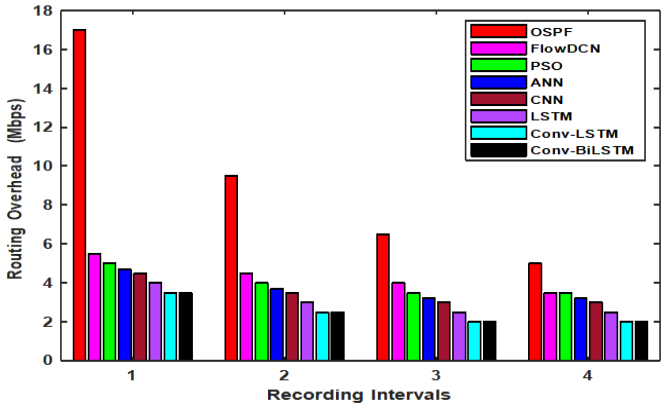

Fig.10. Overhead vs recording interval

Table 2. Error analysis

Features	Dataset t1	Dataset t2	Dataset t3	Dataset t4
Epochs	20	30	40	50
Batch Size	32	32	32	32
Learning Optimizer	Adam	Adam	ReLU	ReLU
Convolution Layer	1	1	2	2
LSTM/BiLSTM Layers	2	2	2	2
Convolution-LSTM Train MAPE	6.02	5.72	5.32	5.05
Convolution-LSTM Test MAPE	5.92	5.53	5.15	4.98
Convolution-BiLSTM Train MAPE	5.63	5.28	5.01	4.91
Convolution-BiLSTM Test MAPE	5.39	5.19	4.87	4.81
Average Training Time (Seconds)	0.4	0.75	1.07	1.22
Learning Rate	0.01	0.01	0.02	0.02

The convolution-RNN model's training and testing error analysis for various periodic datasets is shown in Table 2. Here, the MAPE is assessed for both the training dataset and the testing dataset, and all datasets are trained using the same training hyperparameters. The study showed that as dataset sizes expand (datasets t2, t3, and t4), convolution-RNN models decrease the MAPE.

5. Conclusions

The knowledge plane in KDN improvises the SDN network by providing a detailed analysis of the SDN network topology. To enhance the network performance of the KD-DCN topology, in this paper, we propose a convolution-RNN deep learning model-based online routing strategy. To improve network performance, the Convolution-RNN deep learning model-based proposed system examines historical patterns of network traffic and generates the optimal routing routes. The proposed work performs more effectively in massive traffic datasets, boosting average network throughput by approximately 20% while dropping network latency and loss ratios by approximately 15%, according to experimental evaluation. Hence, it can be stated that the proposed work outperforms deep learning model-based routing systems (ANN, CNN, and LSTM-based approaches), as well as OSPF, FlowDCN, and modified discrete PSO.

Future developments can be considered as various novel and hybrid deep learning models to enhance the KD-DCN topology's network performance. Furthermore, in the next generation of networking, resource management, and energy optimization are major challenges to overcome in the cellular network. The future direction is to enhance cellular network performance by using the KD-DCN topology.

References

[1] Albert Mestres, Alberto Rodriguez-Natal, Josep Carner, Pere Barlet-Ros, Eduard Alarco´n, Marc Sole´, Victor Munte´s-Mulero, David Meyer, Sharon Barkai, Mike J Hibbett, et al. Knowledge-defined networking. *ACM SIG- COMM Computer Communication Review*, 47(3):2–10, 2017.

[2] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communi- cation Review*, 38(2):69–74, 2008.

[3] Michael Jarschel, Thomas Zinner, Tobias Hoßfeld, Phuoc Tran-Gia, and Wolfgang Kellerer. Interfaces, attributes, and use cases: A compass for sdn. *IEEE Communications Magazine*, 52(6):210–217, 2014.

[4] Wei Lu, Lipei Liang, Bingxin Kong, Baojia Li, and Zuqing Zhu. Ai-assisted knowledge-defined network orchestra- tion for energy-efficient data center networks. *IEEE Communications Magazine*, 58(1):86–92, 2020.

[5] Saptarshi Gosh, Brahim El Boudani, Tasos Dagiuklas, and Muddesar Iqbal. So-kdn: A self-organised knowledge defined networks architecture for reliable routing. In *Proceedings of the 4th International Conference on Information Science and Systems*, pages 160–166, 2021.

[6] Weiye Zheng, Jizhong Zhu, and Qingju Luo. Distributed dispatch of integrated electricity-heat systems with variable mass flow. *IEEE Transactions on Smart Grid*, 14(3):1907–1919, 2022.

[7] Bomin Mao, Fengxiao Tang, Zubair Md Fadlullah, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani. A novel non-supervised deep-learning-based network traffic control method for software defined wireless networks. *IEEE Wireless Communications*, 25(4):74–81, 2018.

[8] Bomin Mao, Zubair Md Fadlullah, Fengxiao Tang, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani. Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning. *IEEE Transactions on Computers*, 66(11):1946–1960, 2017.

[9] Tejas Modi and Pravati Swain. Flowdcn: Flow scheduling in software defined data center networks. In *proceedings of the 3rd IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pages 1–5, 2019.

[10] Tug˘rul Ç avdar and S¸ eyma Aymaz. New approach to dynamic load balancing in software-defined network-based data centers. *ETRI Journal*, 45(3):433–447, 2023.

[11] Albert Mestres, Eduard Alarco´n, Yusheng Ji, and Albert Cabellos-Aparicio. Understanding the modeling of com- puter network delays using neural networks. In *proceedings of the Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, pages 46–52, 2018.

[12] Bomin Mao, Fengxiao Tang, Zubair Md Fadlullah, and Nei Kato. An intelligent route computation approach based on real-time deep learning strategy for software defined communication systems. *IEEE Transactions on Emerging Topics in Computing*, 9(3):1554–1565, 2019.

[13] Abdelhadi Azzouni and Guy Pujolle. Neutm: A neural network-based framework for traffic matrix prediction in sdn. In *proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS)*, pages 1–5, 2018.

[14] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[15] Junfeng Xie, F Richard Yu, Tao Huang, Renchao Xie, Jiang Liu, Chenmeng Wang, and Yunjie Liu. A survey of machine learning techniques applied to software defined networking (sdn): Research issues and challenges. *IEEE Communications Surveys & Tutorials*, 21(1):393–430, 2018.

[16] Priyanka Lokhande and Bhavana S Tiple. A step towards advanced machine learning approach: Deep learning. In *proceeding of the IEEE International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pages 1112–1116, 2017.

[17] Alex MR Ruelas and Christian Esteve Rothenberg. A load balancing method based on artificial neural networks for knowledge-defined data center networking. In *proceedings of the 10th Latin America Networking Conference*, pages 106–109, 2018.

[18] Jonghwan Hyun, Nguyen Van Tu, and James Won-Ki Hong. Towards knowledge-defined networking using in-band network telemetry. In *proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS)*, pages 1–7, 2018.
[19] Daniela Maria Casas Velasco, Nelson Luis Saldanha da Fonseca, and Oscar Maur´ıcio Caicedo Rendo´n. Routing based on reinforcement learning for software-defined networking. In *Extended Annals of the XXXIX Brazilian Symposium on Computer Networks and Distributed Systems*, pages 185–192, 2021.
[20] Yuxiang Hu, Ziyong Li, Julong Lan, Jiangxing Wu, and Lan Yao. Ears: Intelligence-driven experiential network architecture for automatic routing in software-defined networking. *China Communications*, 17(2):149–162, 2020.
[21] Yu Li, Hu Wang, and Juanjuan Liu. Can cnn construct highly accurate models efficiently for high-dimensional problems in complex product designs? *arXiv preprint arXiv:1712.01639*, 2017.
[22] Wolfgang Braun and Michael Menth. Software-defined networking using openflow: Protocols, applications and architectural design choices. *Future Internet*, 6(2):302–336, 2014.

Authors' Profiles

Tejas M. Modi is an Assistant Professor in the Department of Computer Science and Engineering at Adani University, Ahmedabad, Gujarat, India. He has a Ph.D. from the Department of Computer Science and Engineering, National Institute of Technology Goa. Dr. Modi has a B.E. degree in Computer Engineering and an M.E. degree in Information Technology from Gujarat Technological University, Ahmedabad, Gujarat, India in 2013 and 2015, respectively. His research interests include Software Defined Networks, UAV Communications, Edge Computing, and Blockchain Technology.

Kuna Venkateswararao is an Assistant Professor in the Department of Computer Science and Engineering at GITAM (Deemed to be) University, Visakhapatnam, India. He received his B.E. degree and M.Tech degree in Computer Science and Engineering from Jawaharlal Nehru Technological University, Kakinada, India, in 2014 and 2017, respectively. He completed PhD in the Department of Computer Science and Engineering at the National Institute of Technology Goa. His research interests include Wireless Software Defined Networks, 5G Mobile Networks, blockchain, and UAV communications.

Pravati Swain received her Master's degree in Computer Science and Master's in Mathematics from Utkal University, Orissa, and PhD degree from the Department of Computer Science and Engineering, Indian Institutes of Technology Guwahati, India. She is an Assistant Professor in the Computer Science and Engineering Department at the National Institute of Technology Goa, India. Prior to NIT Goa, she was a research associate in the Department of Computer Science and Electrical Engineering at the University of Missouri Kansas City, USA. Her present research focus is on Software Defined Networks, 5G Cellular Networks, UAV communications, Performance Modeling of Computer Networks using the Markov model, and Game theory.

How to cite this paper: Tejas M. Modi, Kuna Venkateswararao, Pravati Swain, "Deep Learning Based Traffic Management in Knowledge Defined Network", International Journal of Intelligent Systems and Applications(IJISA), Vol.16, No.6, pp.73-83, 2024. DOI:10.5815/ijisa.2024.06.04