Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 8, No.2, August : 2024

ENHANCING NETWORK TRAFFIC MANAGEMENT WITH DEEP LEARNING: A COMBINED LSTM AND CNN APPROACH

D. Yadaiah Research Scholar, Department of Computer Science and Engineering, University College of Engineering Osmania University, Hyderabad, Telangana-500007, INDIA. Email: yadaiah.dubbaka@gmail.com
Dr. K. Radhika Professor, Department of Artificial intelligence and Data Science, Chaitanya Bharathi Institute of Technology, Gandipet, Hyderabad, Telangana-500075, India. Email: kradhika_it@cbit.ac.in

Abstract
In the evolving sector of network traffic management, the capability to comprehend and predict traffic patterns is critical in facilitating efficient resource allocation and preventing traffic congestion. There is a need to bridge the existing gap in real-time data analysis in this scenario. This paper presents a deep learning approach that uses a combination of Recurrent Neural Networks (RNN) and Convolution Neural Networks (CNN) to predict network traffic flows and categorize network nodes depending on their distinctive features. As such, the RNN network known as Long Short-Term Memory (LSTM) can predict traffic flows trends based on the previous network traffic as CNNs differentiate various nodes based on meaningfully distinctive network features. Here, the LSTM network establishes a correlation between network time data on network flow patterns to predict future network traffic trends. CNNs, on the other hand, effectively classify nodes by analyzing spatial relationships within the network. The above results show that the proposed LSTM method realizes a prediction accuracy of 92%, surpassing the output of linear regression and neural network traditional methods. Similarly, the CNN node classification could realize 91.35% accuracy, making improvements on the present algorithms.

Keywords:
Network Traffic Prediction, LSTM Networks, CNNs, Node Classification, Network Management, Deep Learning

1. Introduction

Traffic flow in networks pertains to the transmission of data packets or information inside a network architecture. It involves the transmission of digital data between different nodes, such as computers, servers, routers, and switches [1]. The movement of traffic may fluctuate in terms of the amount of vehicles, their velocity, and the path they take, which is impacted by variables such as the structure of the network, the actions of users, the requirements of applications, and the level of congestion in the network. Traffic flow prediction involves using past data and present network conditions to forecast the future patterns of network traffic [2]. The objective of this predictive analysis is to anticipate and avert congestion, optimize resource allocation, enhance the Quality of Service (QoS), and enhance the overall performance of the network.

Accurately forecasting traffic patterns is essential for optimizing network administration and operation [3]. Through precise prediction of traffic patterns, network managers may strategically deploy resources, proactively mitigate bottlenecks, and guarantee seamless data transfer. Furthermore, precise forecasts provide improved capacity planning, assisting firms in efficiently expanding their networks to meet increasing needs [4]. In addition, traffic flow prediction helps improve network security by detecting aberrant or suspicious traffic patterns, allowing for prompt action to minimize possible risks. Various methodologies are used for the anticipation of traffic movement in networks [5]. These methods encompass statistical techniques such as time series analysis, autoregressive integrated moving average (ARIMA) models, and machine learning algorithms like artificial neural networks (ANNs), support vector machines (SVMs), and deep learning models such as recurrent neural networks

(RNNs) networks. These methods use previous traffic data, network parameters, and contextual information to provide precise forecasts of future traffic patterns [6].

Although there have been improvements in traffic flow prediction algorithms, there are still various areas of study that need to be addressed. An important deficiency exists in the incorporation of real-time data sources and dynamic environmental elements into prediction models [7]. Existing methodologies often depend on past data and may not promptly adjust to abrupt changes or irregularities in network circumstances. Furthermore, the capacity to scale and the computational intricacy of some prediction algorithms present difficulties, especially in extensive networks with a significant amount of traffic. Furthermore, there is a need for more extensive assessment frameworks to appraise the effectiveness and resilience of prediction models in various network circumstances and traffic patterns [8]. To address these shortcomings, it is necessary to conduct multidisciplinary research and develop creative methods to improve the accuracy, efficiency, and scalability of traffic flow forecast in networks.

2. Literature

Ming Li et al [9] presented LA-ResNet, a convolutional network that uses an attention strategy to handle spatio-temporal modeling and forecast wireless network traffic. The recurrent neural network, the attention mechanism, and the residual network make up the LA-ResNet. By simulating the temporal and spatial characteristics of wireless network traffic data, this method may improve the qualities that are related to it. As a result, it is possible to precisely capture the temporal and geographical correlation of data in wireless network traffic. The data's spatial information may be captured by the residual network. The temporal dependence of the data may be captured by the use of both the attention mechanism and the recurrent neural network.

Smita Mahajan et al [10] carried out a thorough analysis of a few of the methods used in traffic forecasting today. More specifically, analyzing a case study entails forecasting the High-Speed Diesel (HSD) pump's performance based on an analysis of its output. A mesh network is made up of a collection of sensors that act as nodes and gather information on temperature, vibration, three-phase current, voltage, and other factors. In this case study, the vibration parameter is used as the output parameter to anticipate the performance of high-speed diesel pumps. Other performance-related elements are among the identified variables influencing the variation in the High-Speed Diesel Pump's vibration value.

K. Tamil Selvi et al [11] employed fusion learning between the data plane and control plane of the SDN environment, a framework has been created with the goal of improving communication efficiency and intelligence in traffic prediction. Through fusion learning, the prediction model may share data distribution and parameters of the model of the SDN client frameworks in a single transaction. Network traffic forecasting is more accurate when the deep neural model for prediction is used in the SDN controller's global topology manager. The GRU model for time series analysis focuses only on capturing the temporal dependence. In order to effectively manage the fluctuations in network traffic and optimize the analysis of traffic patterns, it is necessary to accurately capture geographical dependencies. In the encoder-decoder design, the GRU includes a diffusion convolution operation that efficiently captures the temporal and spatial relationships of the features. By using the ideal decay rate, stochastic gradient-based planned sampling improves the prediction model's performance.

Zi Wang et al [12] presented a spatial-temporal analysis of a real-world dataset of cellular network traffic. It also reviews the latest research efforts in this subject. Consequently, we suggest a graph attention network called TSGAN, which utilizes time-series similarity to anticipate cellular traffic in a spatial-temporal context.

Amin Azari et al [13] conducted a comprehensive empirical assessment of the implemented solutions using an actual network traffic dataset. This investigation examines the effects of several factors, including temporal granularity, duration of future forecasts, and feature selection. One possible use for these technologies is an ML-powered Discontinuous Reception (DRX) method designed to save

energy. To achieve this goal, we use the ML models that have been generated to change the dynamic DRX parameters based on user traffic. The performance assessment findings clearly indicate that LSTM outperforms ARIMA in most cases, particularly when the training time series is sufficiently long and when it is enhanced by a well-chosen collection of features.

Atif Mahmood et al [14] proposed nhyttt The adaptive capacity and frequency optimization (ACFO) technique for adaptive optimization using a time series forecasting methodology. An analysis is conducted on the daily capacity use of microwave (MW) connections in order to utilize predicted demand. The capacity and frequency optimization will be carried out based on the predicted demand. We used two primary forecasting models, namely SARIMA and MLP. To assess their effectiveness, we employed the RMSE and MAPE criteria.

Hanqiu Wang et al [15] presented a new traffic flow prediction protocol that uses a spatial-temporal graph convolutional network (ST-GCN). This protocol takes into account both the geographical and temporal aspects of intersection traffic, resulting in a more precise prediction of traffic flow. In contrast to previous studies, our suggested protocol, using the Adjacent-Similar algorithm, is capable of accurately forecasting traffic flow at crossings even in the absence of historical data.

Su P. Sone et al [16] performed a comprehensive study of the temporal and geographical aspects of network traffic using actual data from a corporate network that consists of 470 access points (APs). We categorize and segregate access points (APs) based on their traffic use trends. We analyze numerous statistical characteristics of traffic data, including auto-correlations and cross-correlations, both inside and across distinct sets of access points (APs). Our investigation indicates that the group of access points (APs) with significant traffic usage exhibit pronounced seasonality trends. Nevertheless, there are additional access points (APs) that do not exhibit any seasonal variations. We also examine the correlation between the number of connected users and the amount of traffic created. Our findings demonstrate that an increase in connected users does not always result in a corresponding increase in traffic data, and conversely, a higher amount of traffic data does not always indicate a larger number of connected users.

3. Proposed Method

The proposed approach consists of two primary portions designed to improve network analysis and management. Firstly, it utilizes Long Short-Term Memory (LSTM) networks to forecast traffic flow inside networks. LSTM, a kind of RNN, is very skilled at representing sequential data, making it well-suited for forecasting traffic patterns over time inside a network architecture. This section aims to use the LSTM's capacity to capture temporal relationships in network data in order to properly predict traffic flow. Additionally, the approach utilizes Convolutional Neural Networks (CNNs) to classify incoming nodes. CNNs are very proficient in extracting hierarchical features from spatial data, which makes them exceptionally well-suited for applications like picture categorization. In this particular situation, CNNs are used to categorize incoming nodes in the network according to their distinct features or actions, hence assisting in the effective administration of the network and allocation of resources. The proposed process for traffic flow and classify the input nodes is shown in figure 1.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 8, No.2, August : 2024

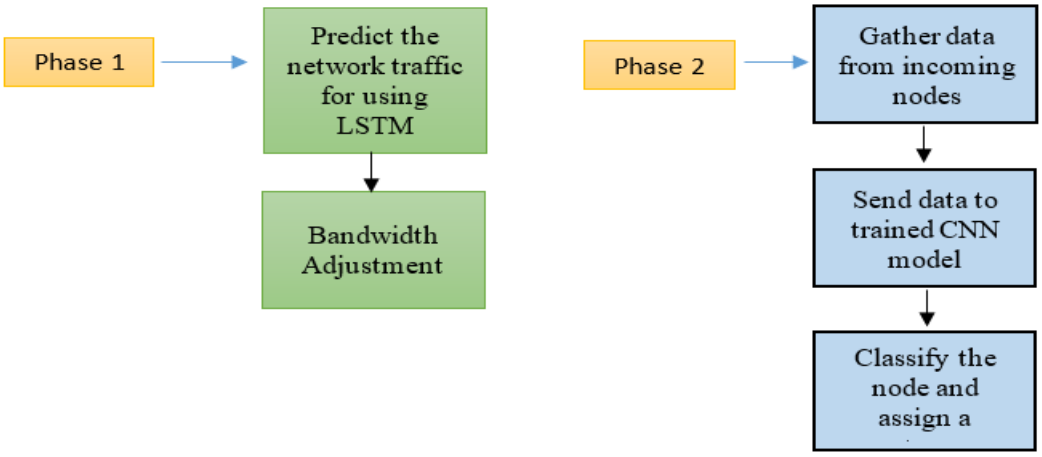

Figure 1: The traffic flow and assign the nodes using deep learning model

Phase 1: Predicting network traffic using Long Short Term Memory (LSTM)
Step 1: Gather past network traffic information.
Step 2: Use the input traffic data to train the LSTM network.
Step 3: Apply the trained model to forecast the traffic statistics in the future.
Phase 2: Use a deep learning model, Convolution Neural Networks (CNNs) to classify the input nodes.
Step 1: Use node assignment data to train the CNN model.
Step 2: Use the trained model to categorize newly incoming nodes.

3.1 Traffic Flow Prediction

3.1.1 Long Short-Term Memory (LSTM)

A particular kind of RNN structure called Long Short-Term Memory (LSTM) was created in response to the limitations of normal RNNs in precisely identifying long-term associations in sequential input. Deep learning has advanced significantly with the use of LSTM networks. Their exceptional ability to maintain and change information selectively over extended durations makes them particularly valuable in tasks involving the processing of sequential input.

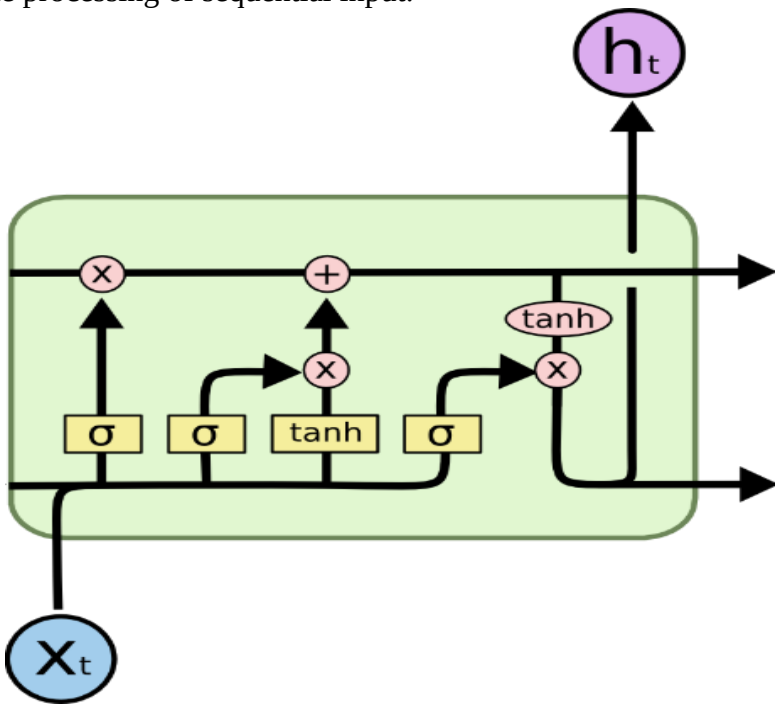

Figure 2: Architecture of LSTM

The cell state, shown by the horizontal line across the top of Figure 2, is the key component of LSTMs. The condition of the cell may be compared to that of a conveyor belt. It follows an unbroken path throughout the whole chain, with just a few brief sequential exchanges taking place in between. It allows information to go through it with ease and without change.

The LSTM has the capacity to alter the cell state by selectively removing or incorporating information, a process that is controlled by specialized structures known as gates. Gates provide the regulated transfer of information inside a system. The construction of these networks involves the use of a layer of sigmoid neural networks and a pointwise multiplication operation as the core components. The sigmoid layer generates an output that spans from 0 to 1, indicating the percentage of each component that should be allowed to pass through. A value of zero signifies a condition of "complete restriction," whereas a value of one denotes a state of "unrestricted access." An LSTM is equipped with three gates: the input gate, forget gate, and output gate. These gates have the function of safeguarding and regulating the cell state.

LSTM is a specific kind of RNN. In a RNN, the outcome from the previous time step is employed as the input for the future time step. The research addressed the issue of the RNN incapacity to forecast words stored in its long-term memory. However, it demonstrated that the RNN may generate more precise predictions by relying on more recent input. Increasing the length of the gap results in decreased efficiency of the RNN. LSTM has a default tendency to retain information for an extended duration. Time-series data is used for processing, forecasting, and categorization.

LSTM is a specialized kind of RNN that is particularly developed to handle sequential data, such as time series, audio, and text. The model's name, LSTMs, is an acronym formed from "Long Short-Term Memory." LSTM networks are very proficient in capturing and comprehending intricate patterns over long durations in sequential data. These capabilities make them well-suited for applications like as language translation, voice recognition, and time series forecasting.

A traditional RNN usually comprises a single hidden state that is transmitted across time, posing difficulties for the network to accurately capture long-term relationships. In order to tackle this issue, LSTM models include a memory cell that functions as a reservoir, capable of storing information for a prolonged period of time. The memory cell is controlled by three gates: the input gate, the forget gate, and the output gate. These gates control the process of including, excluding, and retrieving data in the memory cell.

The input gate regulates the transmission of data directed towards the memory cell. The forget gate is responsible for determining which segments of data are deleted from the memory cell. The output gate regulates the transfer of data from the memory cell. LSTM networks has the capacity to selectively keep or discard information as it traverses the network, allowing them to acquire and encapsulate long-term connections.

Deep LSTM networks may be constructed by stacking many LSTM layers, enabling them to acquire intricate patterns from sequential input by using their ability to learn multiple levels of the pattern. LSTMs may be used in conjunction with CNNs for the purpose of analyzing images and videos.

Structure of LSTM:

1. Forget Gate
2. Input gate
3. Output gate

1. Forget Gate: Any unnecessary information is removed from the cell state using the forget gate. Before the bias is applied, the two inputs to the gate—x_t, the input at this moment, and h_{t-1}, the preceding cell output—are multiplied by weight matrices. An activation function that produces a binary output is used to send the result. A cell state that produces a result of zero is said to have lost its information, while a result of one indicates that the information is kept for later use.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 8, No.2, August : 2024

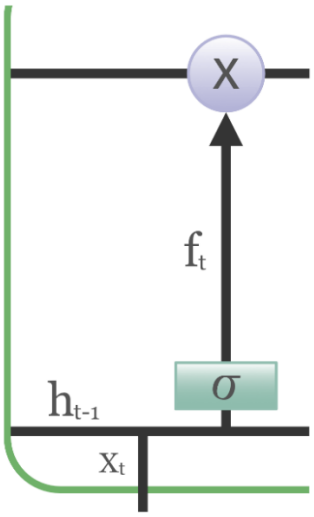

Figure 3: Forget Gate

2. Input gate: Relevant information is incorporated into the cell's current state via the input gate. The information is initially regulated by the sigmoid function. Then, in a manner similar to that of a forget gate, the inputs h_{t-1} and x_t are employed to keep certain values. In the end, the tanh function is used to create a vector, producing an output that ranges from -1 to +1. All of the potential values obtained from h_{t-1} and x_t are included in this vector. To offer pertinent information, the vector values and the controlled values are multiplied at the conclusion.

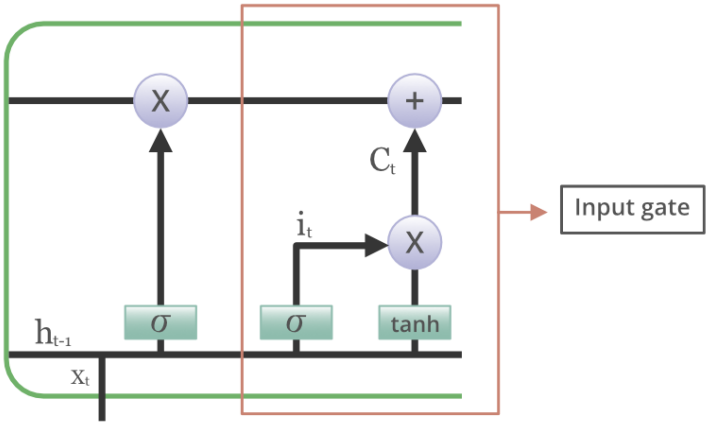

Figure 4: Input gate

3. Output gate: In order for the output to be shown, the output gate must gather pertinent data from the cell's present state. A vector is produced when the hyperbolic tangent function is applied to the cell. Employing the inputs h_{t-1} along with x_t, the data is then further filtered by determining which values need storage in memory. The data is regulated by the sigmoid function. In the end, the vector values and the controlled values are multiplied and transmitted to the next cell as an input and an output.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 8, No.2, August : 2024

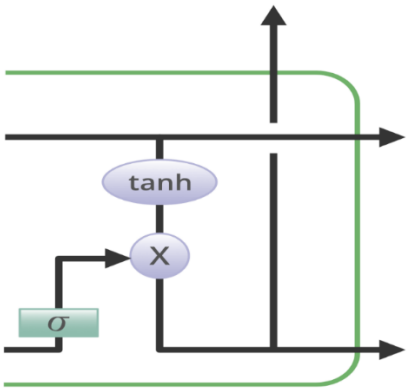

Figure 5: Output gate

Algorithm Steps for LSTM

Step 1: Set the initial values for the weights and biases of the LSTM units.

Step 2: Specify the input sequence and target sequence.

Perform forward propagation on the input sequence using the LSTM units.

Step 4: Calculate the discrepancy between the forecasted sequence and the desired sequence.

Step 5: Propagate the gradients backwards via the LSTM units.

Step 6: Revise the weights and biases by using an optimization method such as stochastic gradient descent.

Step 7: Iterate steps 3-6 for numerous epochs until convergence is achieved.

The first section of the proposed approach centers on forecasting traffic flow inside networks, using LSTM networks. LSTM, a kind of RNN, is very skilled at identifying temporal connections in sequential data, making it ideal for forecasting traffic patterns. The LSTM model utilizes previous traffic data to understand the fundamental patterns of network flow, allowing for precise predictions of future traffic circumstances.

3.2 Node Classification

3.2.1 Convolutional Neural Networks (CNNs)

CNN, a specific form of artificial neural network that is often used in tasks related to visual vision, such as picture identification and classification. CNNs excel in these tasks because to their ability to autonomously and flexibly acquire spatial hierarchies of characteristics from the input data.

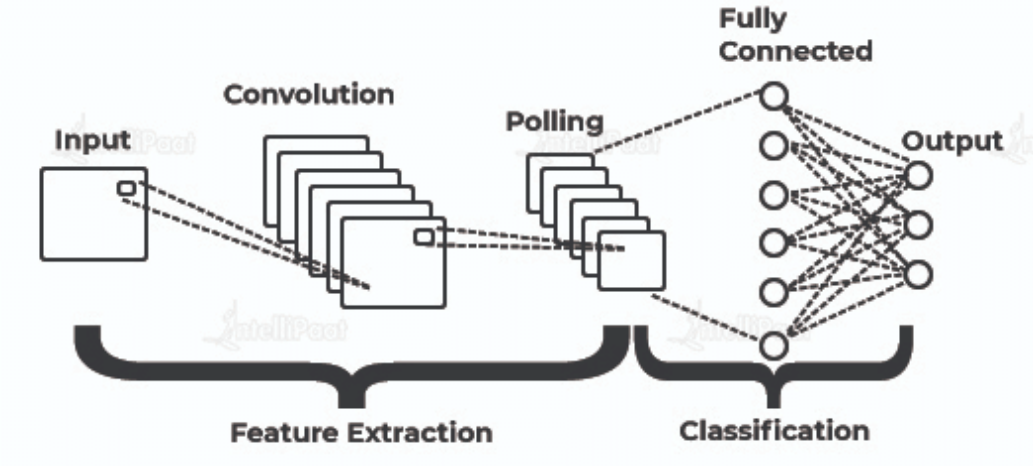

Figure 6: CNN Architecture

A CNN often has many layers, which include:

- **Input Layer**

The Input Layer of a CNN acts as the entry point for unprocessed data, such as a picture, word, or any other kind of input. This layer plays a vital role in image-related activities by processing pixel data and organizing them into a manner that the network can understand. For example, when dealing with

photographs, the input layer organizes the pixel values into a multi-dimensional array, usually with dimensions that indicate the height, width, and color channels. In addition to just receiving data, the input layer lays the groundwork for future layers to identify significant characteristics. The canvas serves as the medium via which the network expresses its comprehension of the incoming data. Every individual pixel value or fragment in the input layer may be compared to a brushstroke, containing information that the network progressively improves and understands as it passes through its levels.

In addition, the input layer establishes the initial boundaries that the network operates within. For instance, the size of the receptive fields across the network and the size of the filters used in convolutional layers are determined by the dimensions of the input layer. When evaluating the network's ability to understand intricate patterns and relationships in the data, these metrics are essential. As the first step in the network's development towards understanding and abstraction, the input layer essentially symbolizes the basic characteristics of the data. The meticulous organization and pre-processing of the input data provide a solid basis for the succeeding layers to uncover the intricacies hidden therein. This finally allows the network to make well-informed judgments or predictions depending on the data it is provided with. Although it may seem simple, the input layer has a crucial role in determining the capabilities and efficiency of the complete CNN design.

- **Convolutional layers**

Convolutional layers serve as the base of CNNs, enabling them to efficiently process visual data such as images. These layers operate by systematically applying learnable filters, also known as kernels, across the input data. Each filter is a small matrix of weights that is convolved with a corresponding region of the input data, resulting in a feature map that highlights certain patterns or features within the input. This process allows CNNs to automatically extract relevant features from the raw input, such as edges, textures, or shapes, without the need for manual feature engineering. One of the key advantages of convolutional layers is their ability to capture spatial hierarchies of features. As the input data passes through successive convolutional layers, higher-level features are constructed by combining lower-level features from previous layers. This hierarchical feature extraction enables CNNs to learn increasingly complex representations of the input data, ultimately leading to more discriminative features for tasks like object recognition or image classification.

Moreover, convolutional layers introduce parameter sharing and translational invariance, which are crucial for reducing the number of parameters in the network and making it robust to variations in the input. Parameter sharing refers to the fact that the same set of weights is used across different regions of the input, allowing the network to learn spatially invariant features. Translational invariance means that the learned features are insensitive to shifts or translations in the input, making the network more robust to changes in position or orientation of objects within an image. Convolutional layers often include additional elements, such as padding and stride, to regulate the spatial dimensions of the output feature maps, in addition to convolutional filters. Padding is used to maintain the spatial resolution of the input, while stride specifies the interval at which the filters are applied across the input. The settings have a significant impact on the operation of the convolutional layers and may affect the size and qualities of the learnt features.

- **Activation Function**

The Activation Function is a critical element of CNNs that introduces non-linearity into the network's calculations. Within the realm of CNNs, the activation function functions as a regulator, deciding whether and to what degree a neuron should be engaged based on the input it receives. Rectified Linear Unit (ReLU) is widely used as an activation function in CNNs because of its simplicity and efficacy.

The ReLU function functions by executing a basic thresholding operation: if the input to a neuron is positive, ReLU produces the same value, therefore activating the neuron; if the input is negative, ReLU outputs zero, so deactivating the neuron. ReLU's fundamental simplicity enables it to efficiently mitigate the vanishing gradient issue that often arises in deep neural networks. This problem occurs when the gradients become very tiny during backpropagation, impeding the network's learning capacity. ReLU addresses this problem by keeping a consistent slope for positive inputs, which helps

to improve the efficiency and stability of training. Moreover, the piecewise-linear characteristic of ReLU allows CNNs to represent rich and varied non-linear connections in the data. This is particularly important for applications like image recognition, where the underlying patterns and features may be exceedingly complex. ReLU aids in the extraction and enhancement of important characteristics from the input data in CNNs by using a series of convolutional and pooling layers. This enables the network to distinguish more complex and distinctive representations.

- **Pooling layer**

Pooling layers in CNNs are essential for decreasing the spatial dimensions of the feature maps generated by convolutional layers while preserving significant information. Reducing the dimensionality of the data helps to manage the amount of parameters and calculations in the network, resulting in improved computational efficiency. Furthermore, pooling layers contribute to the development of spatial invariance by detecting the existence of features irrespective of their precise positions in the input data. Max pooling is a frequently used kind of pooling in CNNs. Max pooling involves partitioning the input feature map into non-overlapping rectangular sections, known as pooling windows, and selecting the highest value inside each window while rejecting the other values. Max pooling preserves the most salient characteristics in the input data by retaining just the highest activation within each zone. This approach efficiently decreases the spatial dimensions of the feature maps, leading to a more condensed representation.

Another type of pooling is average pooling, where instead of retaining the maximum value, the average value within each pooling window is computed and retained. While average pooling is less commonly used than max pooling, it can still be effective in certain scenarios, particularly when the goal is to retain a more generalized representation of the input features. Pooling layers also contribute to the translational invariance of CNNs, meaning they can recognize the same features regardless of their precise location within the input data. This is achieved by aggregating information from neighboring regions and summarizing it into a single value, thereby capturing the presence of features irrespective of their spatial positions. This property is particularly beneficial in tasks such as image classification, where the location of objects within an image may vary.

- **Fully connected layers**

Fully connected layers, also known as dense layers, are essential components of CNNs, especially in tasks like picture classification. The aforementioned layers play a vital role in extracting high-level features from the low-level features obtained from the previous convolutional and pooling layers. The term "fully connected" is used to describe the fact that every neuron in a thick layer is connected to every neuron in the preceding layer. The connectivity design helps the network in obtaining knowledge of complex patterns and correlations among the features acquired from the input data. The outcome of each neuron in the dense layer is determined by calculating a weighted sum of the inputs from the previous layer, followed by the application of an activation function.

During the training phase, the dense layers' parameters (weights and biases) are acquired using backpropagation, in which the network modifies these parameters to minimize a predetermined loss function. The adjustment procedure enables the network to progressively enhance its capacity to appropriately identify incoming data. The quantity of neurons in the dense layers and the structure of the overall network, encompassing the number of dense layers, are generally established according to the intricacy of the job and the magnitude of the dataset. Networks with increased depth and denser layers have the ability to capture more complex correlations in the data. However, they also demand more computing resources and are susceptible to overfitting if not adequately regularized.

- **Output Layer**

The Output Layer of a Convolutional Neural Network (CNN) is the last level when the network generates predictions or classifications using the characteristics learnt from the input data. The architecture of the CNN is mostly determined by the specific requirements of the job it is meant to address. For classification tasks, which are often encountered in CNNs, the output layer usually

comprises neurons that correspond to the potential classes or categories that the model intends to categorize.

Every individual neuron in the output layer corresponds to a certain class, and the level of activity of these neurons is read as the likelihood that the input is associated with the relevant class. The softmax function is often used as the activation function in the output layer for classification tasks. Softmax function transforms the unprocessed output scores, also known as logits, from the previous layer into probabilities, guaranteeing that their total equals 1. This enables the network to generate a probability distribution over all the classes, facilitating the interpretation and comparison of the model's confidence in its predictions.

Algorithm Steps for CNN

Step 1: Input layer - Data input
Step 2: Convolutional layer - Utilize filters to extract distinctive characteristics.
Step 3: Activation layer - Apply activation functions such as ReLU to introduce non-linearity.
Step 4: Pooling layer - Spatial dimensions are reduced using procedures such as max pooling.
Step 5: Repeat steps 2-4 to perform more extensive feature extraction.
Step 6: Flatten layer - Transform 2D feature maps into a 1D vector.
Step 7: Fully connected layer - Establish connections between neurons from the preceding layers in order to carry out categorization tasks.
Step 8: Output layer - Generate ultimate categorization predictions.
Step 9: Loss computation - Determine the discrepancy between the expected and real labels.
Step 10: Backpropagation - Adjust model parameters to minimize loss.
Step 11: Iterate steps 1-10 for numerous epochs in order to enhance performance.

The second portion of the proposed technique is dedicated to the categorization of arriving nodes in the network. Using Convolutional Neural Networks (CNN), this phase seeks to precisely categorize nodes as they join the network. Within this particular framework, these methods are used to categorize arriving nodes, essentially considering the network topology as a picture. The CNN can accurately categorize incoming nodes by studying the spatial linkages and patterns inside the network, hence enhancing the efficiency and efficacy of the network management system.

4. Experimental Results

The simulations were conducted under the suggested methodology, and the results were analysed in this section.

4.1 Traffic Prediction

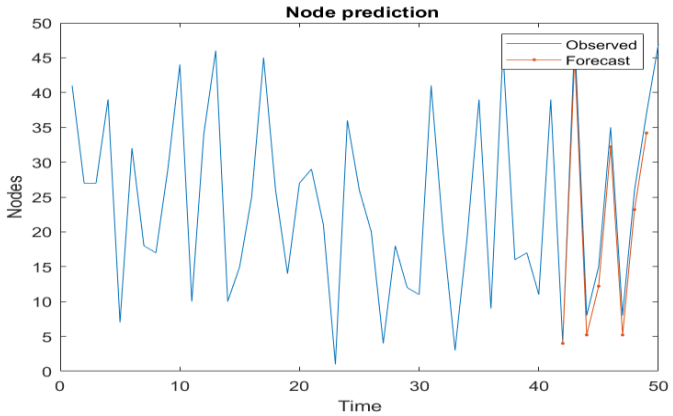

Figure 7: Node prediction

Figure 7 depicts the node forecast for network traffic. In the graph is a time series analysis of node predictions. The graph illustrates the recorded values and projected values over time. The x-axis represents time, while the y-axis represents the quantity of nodes. The measured values are shown by the blue line, exhibiting substantial fluctuations over time and displaying a broad range of node count.

The forecasted values are the red line that closely follows the blue line through time, although with distinctive deviations towards the end. The closely aligned forecasted and observed lines imply that the model applied has achievable forecasted the data trends.

Table 1: comparison with the current procedures

Method	Prediction accuracy
Linear Regression method	85%
Multi-Layer Perceptron	88%
Proposed LSTM method	92%

The table1 displays a comparative analysis of the accuracy of predictions across several methodologies. The Linear Regression approach attains an accuracy of 85%, whilst the Multi-Layer Perceptron obtains a little higher accuracy of 88%. Nevertheless, the Proposed LSTM approach surpasses both methods, demonstrating an amazing accuracy rate of 92%. These findings indicate that the LSTM technique is likely to be a superior strategy for the specified goal in comparison to standard linear regression or neural network approaches.

4.2 Node classification

The table 2 represents a sample dataset for node classification, with parameters including bandwidth for communication between gateways (Gateway1 to Gateway4), node energy, frequency of transmission, and the type of node. Each row corresponds to a different node, with values indicating the bandwidth between gateways, the energy level of the node, the frequency of transmission, and the type of node.

Table 2: Parameters considered for training

Bandwidth				Node Energy	Frequency of transmission	Type of node
Gateway1	Gateway2	Gateway3	Gateway4			
300 Kbps	400 Kbps	100 Kbps	50 Kbps	21	6	2
230 Kbps	150 Kbps	350 Kbps	80 Kbps	15	2	1
340 Kbps	200 Kbps	160 Kbps	400 Kbps	95	8	4
60 Kbps	370 Kbps	120 Kbps	280 Kbps	28	3	2
180 Kbps	320 Kbps	90 Kbps	220 Kbps	36	4	3
70 Kbps	170 Kbps	330 Kbps	290 Kbps	72	7	4
310 Kbps	120 Kbps	260 Kbps	60 Kbps	6	2	1
270 Kbps	40 Kbps	300 Kbps	110 Kbps	39	4	3

From table 2, the bandwidth availability for Gateway1 to Gateway4 in the first row is as follows: 300 Kbps, 400 Kbps, 100 Kbps, and 50 Kbps accordingly. Additionally, the node energy is 21, the frequency of transmission is 6, and the kind of node is 2. The values indicate the resources and features of nodes in a network, which are essential for training classification models to differentiate between various kinds of nodes based on variables such as bandwidth, energy, and transmission frequency.

Table 3: Comparative Analysis

Algorithm	Accuracy
ANN with Back propogation	70.91%
ANN with cascade-forward backpropagation	77.34%
SVM	83.88%
Proposed CNN model	91.35%

Table 3 illustrates the comparison of precision for different algorithms. The Artificial Neural Network with Backpropagation has obtained 70.91%; the ANN with cascade-forward backpropagation indicates 77.34%; the Support Vector Machine has been more efficient than both ANN models, providing 83.88%. Finally, the most reliable and promising results had been achieved by the Proposed Convolutional Neural Network model that has shown 91.35% as the most accurate readings. Thus, in

this Article it can be stated that the Proposed CNN model has the best precision between all algorithms mentioned above; thus, it can be highly efficient in the performance of the following task/dataset.

5. Conclusion

This paper presented the successful combination of Long Short Term Memory and Convolutional Neural Networks in managing network traffic. Key results from the study include the LSTM achieved a significant accuracy of 92% in the prediction of the volume of network traffic compared to other methods for prediction such as linear regression, as well as neural networks. Likewise, the CNN-based node classification had an accuracy of 91.35% which was considerably higher than the other classification algorithms. By capturing temporal dependencies in traffic data and effectively classifying nodes based on their spatial characteristics, the proposed model offers a robust solution for network management challenges.

References

[1] Zeng, Qingtian, Qiang Sun, Geng Chen, Hua Duan, Chao Li, and Ge Song. "Traffic prediction of wireless cellular networks based on deep transfer learning and cross-domain data." *IEEE access* 8 (2020): 172387-172397.

[2] He, Qing, Arash Moayyedi, György Dán, Georgios P. Koudouridis, and Per Tengkvist. "A meta-learning scheme for adaptive short-term network traffic prediction." *IEEE Journal on Selected Areas in Communications* 38, no. 10 (2020): 2271-2283.

[3] Kim, Meejoung. "Network traffic prediction based on INGARCH model." *Wireless Networks* 26, no. 8 (2020): 6189-6202.

[4] Jaffry, Shan, and Syed Faraz Hasan. "Cellular traffic prediction using recurrent neural networks." In *2020 IEEE 5th international symposium on telecommunication technologies (ISTT)*, pp. 94-98. IEEE, 2020.

[5] Abdellah, Ali R., and Andrey Koucheryavy. "Deep learning with long short-term memory for iot traffic prediction." In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems: 20th International Conference, NEW2AN 2020, and 13th Conference, ruSMART 2020, St. Petersburg, Russia, August 26–28, 2020, Proceedings, Part I 20*, pp. 267-280. Springer International Publishing, 2020.

[6] Chen, Aaron, Jeffrey Law, and Michal Aibin. "A survey on traffic prediction techniques using artificial intelligence for communication networks." In *Telecom*, vol. 2, no. 4, pp. 518-535. MDPI, 2021.

[7] Khedkar, Shilpa P., R. Aroul Canessane, and Moslem Lari Najafi. "Prediction of traffic generated by IoT devices using statistical learning time series algorithms." *Wireless Communications and Mobile Computing* 2021 (2021): 1-12.

[8] Li, Yingqi, Juan Wang, Xiaochuan Sun, Zhigang Li, Miao Liu, and Guan Gui. "Smoothing-aided support vector machine based nonstationary video traffic prediction towards B5G networks." *IEEE Transactions on Vehicular Technology* 69, no. 7 (2020): 7493-7502.

[9] Li, Ming, Yuewen Wang, Zhaowen Wang, and Huiying Zheng. "A deep learning method based on an attention mechanism for wireless network traffic prediction." *Ad Hoc Networks* 107 (2020): 102258.

[10] Mahajan, Smita, R. HariKrishnan, and Ketan Kotecha. "Prediction of network traffic in wireless mesh networks using hybrid deep learning model." *IEEE Access* 10 (2022): 7003-7015.

[11] Selvi, K. Tamil, and R. Thamilselvan. "An intelligent traffic prediction framework for 5G network using SDN and fusion learning." *Peer-to-Peer Networking and Applications* 15, no. 1 (2022): 751-767.

[12] Wang, Zi, Jia Hu, Geyong Min, Zhiwei Zhao, Zheng Chang, and Zhe Wang. "Spatial-temporal cellular traffic prediction for 5G and beyond: A graph neural networks-based approach." *IEEE Transactions on Industrial Informatics* 19, no. 4 (2022): 5722-5731.

[13] Azari, Amin, Fateme Salehi, Panagiotis Papapetrou, and Cicek Cavdar. "Energy and resource efficiency by user traffic prediction and classification in cellular networks." *IEEE Transactions on Green Communications and Networking* 6, no. 2 (2021): 1082-1095.
[14] Mahmood, Atif, Miss Laiha Mat Kiah, Muhammad Reza Z'aba, Adnan N. Qureshi, Muhammad Shahreeza Safiruz Kassim, Zati Hakim Azizul Hasan, Jagadeesh Kakarla, Iraj Sadegh Amiri, and Saaidal Razalli Azzuhri. "Capacity and frequency optimization of wireless backhaul network using traffic forecasting." *IEEE Access* 8 (2020): 23264-23276.
[15] Wang, Hanqiu, Rongqing Zhang, Xiang Cheng, and Liuqing Yang. "Hierarchical traffic flow prediction based on spatial-temporal graph convolutional network." *IEEE Transactions on Intelligent Transportation Systems* 23, no. 9 (2022): 16137-16147.
[16] Sone, Su P., Janne J. Lehtomäki, and Zaheer Khan. "Wireless traffic usage forecasting using real enterprise network data: Analysis and methods." *IEEE Open Journal of the Communications Society* 1 (2020): 777-797.