International Journal of Scientific Research in Computer Science, Engineering and Information Technology

ISSN : 2456-3307

Available Online at : www.ijsrcseit.com

doi : https://doi.org/10.32628/IJSRCSEIT

Traffic Prediction Using Graph Convolution Networks

Bathina Bhanukowshik

Department of Computer Science and Engineering, National Institute of Technology, Andhra Pradesh, India

ARTICLEINFO

Article History:

Accepted: 20 June 2023

Published: 09 July 2023

Publication Issue

Volume 9, Issue 4

July-August-2023

Page Number

57-65

ABSTRACT

Traffic Prediction has become very essential now a days. It is helpful to manage the traffic in some major cities. In reality, the traffic data generated contains some missing values, due to communication errors, sensor faults, etc. Since the traffic data is Spatial-Temporal, which is very complex, it becomes difficult to handle these situations. So, the authors propose a Graph Convolution Network-based model to predict future traffic with missing values. The missing value processing and traffic prediction are processed in one step. The model considers both Spatial-Temporal data and the history of the particular junction traffic. The computational time complexity of the network is optimized for predicting the traffic.

Keywords : Traffic prediction, Graph convolutional network (GCN), Deep learning, Spatial-temporal data, Time series forecasting, Hyperparameter optimization, Model evaluation, Accuracy, Precision, Recall, F1 score, Root mean squared error (RMSE), Mean absolute error (MAE), Mean absolute percentage error (MAPE).

I. INTRODUCTION

Traffic prediction plays a crucial role in transportation planning, traffic management, and urban development. With the increasing availability of transportation data and the advancement of machine learning techniques, Graph Convolutional Networks (GCNs) have emerged as a powerful tool for traffic prediction tasks. GCNs are a type of neural network that can effectively model data with graph structures, making them well-suited for traffic prediction tasks that involve complex spatial relationships and interactions. Several deep learning models commonly used in traffic prediction projects incorporate Graph Convolutional Networks (GCNs). Here are a few notable examples: Graph Convolutional Recurrent Neural Network (GCRNN): GCRNN combines the power of GCNs and recurrent neural networks (RNNs) to capture both spatial and temporal dependencies in traffic data. It incorporates GCNs to model spatial relationships and RNNs to capture temporal patterns, enabling accurate traffic prediction.

Graph WaveNet: Graph WaveNet is inspired by the WaveNet architecture and extends it to graph-structured data. It utilizes dilated convolutions to capture both local and global dependencies within the

traffic network. Graph WaveNet effectively captures long-range dependencies and produces accurate predictions by stacking multiple dilated convolutions. Graph WaveNet effectively captures long-range dependencies and produces accurate predictions.Graph Attention Networks (GAT): GAT is a graph neural network that utilizes attention mechanisms to assign weights to different traffic graph nodes. This allows the model to focus on relevant nodes and capture important spatial relationships. GAT has been applied to traffic prediction tasks and achieved competitive results. Spatial-Temporal Graph Convolutional Networks (STGCN): STGCN is designed specifically for traffic prediction. It incorporates GCNs to model spatial dependencies and temporal convolutional layers to capture temporal patterns. By combining both spatial and temporal information, STGCN effectively predicts traffic flow at different locations and time intervals.

Graph LSTM: Graph LSTM is a variant of the traditional LSTM architecture that is modified to work with graph-structured data. It employs message passing between nodes in the graph to capture spatial dependencies and uses LSTM cells to model temporal dynamics. Graph LSTM has been successfully applied to traffic prediction tasks.These models are just a few examples of the deep learning architectures used in traffic prediction projects that leverage GCNs. Researchers and practitioners continue to explore and develop new models that can better capture the complexities of traffic data and improve prediction accuracy. GCNs are used in traffic prediction projects for several reasons: Capturing Spatial Dependencies: Traffic networks exhibit complex spatial relationships, where the flow of traffic at one location is influenced by neighboring locations. GCNs are well-suited for modeling such spatial dependencies by leveraging the graph structure of the traffic network. They can propagate information between connected nodes and capture the influence of neighboring locations on the traffic conditions at a given location.

Handling Irregular and Non-Euclidean Data: Traffic networks are typically irregular and non-Euclidean, as they consist of nodes (e.g., intersections) connected by edges (e.g., roads). Traditional neural networks assume grid-like or regular structures, which may not capture the inherent characteristics of traffic networks. GCNs, on the other hand, can handle graph-structured data and effectively model the irregular topology of traffic networks.Learning Node Representations: GCNs learn meaningful representations for each node in the traffic network. By propagating information through the graph, GCNs can capture each node's local and global characteristics, such as traffic flow patterns, road characteristics, or nearby amenities. These learned representations can then be used to predict traffic conditions at different locations in the network.Integration with Temporal Modeling: Traffic prediction requires capturing temporal dynamics and trends. GCNs can be combined with recurrent or temporal modeling techniques, such as LSTM or convolutional layers, to model the temporal aspects of traffic data. This integration allows the model to capture both spatial and temporal dependencies, leading to accurate traffic predictions.

Scalability and Efficiency: GCNs can handle large-scale traffic networks efficiently. They leverage localized information through message passing and shared weights, which reduces the computational complexity compared to fully connected neural networks. This makes GCNs suitable for processing large-scale traffic data, enabling real-time or near-real-time traffic prediction.By utilizing GCNs in traffic prediction projects, researchers and practitioners can effectively model spatial dependencies, handle irregular data structures, capture temporal dynamics, and achieve accurate predictions in transportation systems. GCNs offer a powerful framework for analyzing and predicting traffic patterns, which can inform traffic management strategies, optimize transportation infrastructure, and improve overall urban mobility.

II. LITERATURE REVIEW

In recent years, the use of graph convolutional networks (GCNs) for traffic prediction has gained significant attention and has shown great potential. Several novel models have been proposed, leveraging the strengths of GCNs to capture spatial dependencies between traffic nodes in a graph. These models have demonstrated their superiority over traditional methods such as SVR, and LSTM in terms of prediction accuracy and performance.The advantage of GCNs lies in their ability to effectively model the spatial relationships between traffic nodes. Unlike traditional methods that treat traffic nodes as independent entities, GCNs take into account the connectivity and interactions between nodes. By considering the graph structure of the traffic network, GCNs can capture the influence and dependencies among nodes, which are crucial for accurate traffic prediction.

One approach that has been widely explored is the combination of GCNs with recurrent neural networks (RNNs), such as LSTM. This integration allows the models to capture both spatial and temporal dependencies in traffic data. The GCN component learns the spatial relationships between nodes, while the LSTM component captures the temporal patterns and dynamics. This combination proves to be effective in capturing the complex nature of traffic data, resulting in improved prediction accuracy.

Additionally, the introduction of attention mechanisms in GCNs, such as graph attention networks (GATs), has further enhanced the performance of traffic prediction models. Attention mechanisms allow the models to dynamically assign importance to different nodes in the traffic graph, enabling them to focus on relevant and influential nodes for prediction. This attention-based approach has shown promising results in capturing the varying degrees of impact that different nodes have on the overall traffic patterns.

Furthermore, recent models have explored the use of advanced architectures like Graph WaveNet, which combines the strengths of GCNs and WaveNet architecture. WaveNet, originally proposed for speech generation, is a deep generative model capable of capturing long-term dependencies. By incorporating a GCN within the WaveNet framework, the Graph WaveNet model achieves powerful representation learning of both spatial and temporal dependencies, leading to improved traffic prediction accuracy.

It is important to note that while GCNs have shown promising results for traffic prediction, there are still challenges and areas for improvement. The scalability of GCNs to large-scale traffic networks, the incorporation of external factors (such as weather conditions or events) into the models, and the robustness of noisy or incomplete data are among the ongoing research directions.In conclusion, the utilization of GCNs for traffic prediction has demonstrated significant advancements in recent years. The ability of GCNs to model spatial dependencies in traffic networks, combined with recurrent or generative architectures, has led to improved prediction accuracy compared to traditional methods. Continued research in this area holds great potential for further enhancing our understanding and capabilities in predicting and managing traffic congestion effectively.

III. PROBLEM FORMULATION

Traffic data can be represented as multivariate time series on a traffic network, like a set of roads and junctions. Let the traffic network G = {V, E}, where V = {V_1, V_2, V_3, … V_n} is a set of N traffic nodes, which can be known as the junctions, and E = {E_1, E_2, E_3, … E_n} denotes set of E edges connecting the nodes, which denotes the roads connecting the junctions. Each node or a value of an observation contains some F features of the traffic like vehicle flow, the average speed of the

vehicle, occupancy of the roads, etc. A mask sequence M is defined to denote the missing value using a Boolean value 0 or 1. $M_t = (m_t^1, m_t^2, \ldots m_t^n)$. Value of $m_t^i = 0$ if a value is missing else, it will be 1. The model takes the set of traffic features as input and predicts the traffic for the future.

IV. METHODOLOGY

4.1. Model Architecture

This model contains a Multi-scale Memory module, an Output Forecasting module, and l – Spatial-Temporal (ST) blocks. Each Spatial-Temporal block contains Temporal Convolution, Dynamic Graph Construction, and Dynamic Graph Convolution. The output forecasting module uses the skip connections on the output of the final Spatial-Temporal block and the hidden states after each temporal convolution layer and the final output is predicted.

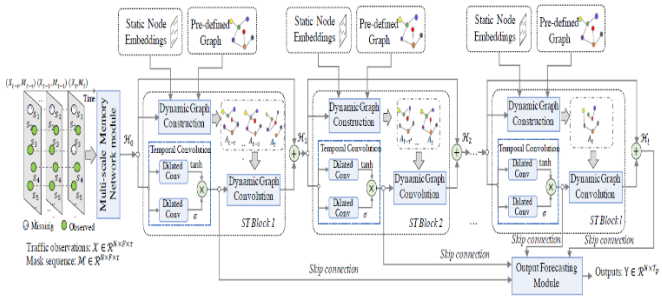

Figure 1: Model Architecture [1]

4.2. Multi-Scale Memory Network

The multi-scale memory network module uses historical data, like hourly, daily, and weekly data, and combines them to form an input and uses this input and local features like Empirical temporal mean, empirical spatial mean, nearest spatial observation, nearest temporal observations to form Enriched traffic embeddings

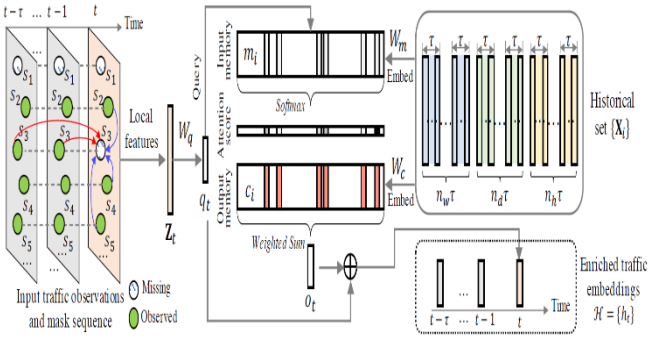

Figure 2: Multiscale Memory Module [1]

4.2.1 Local Features

4.2.1.1 Empirical Temporal Mean

The mean of previous L samples can be useful for predicting the current value since the mean denotes the average of all the samples, hence if it is missing, a better idea would be to take a mean of the last L samples. Hence, for every missing sample, we take the empirical temporal mean using the last L Samples.

$$\bar{x}_t^i = \sum_{l=t-L}^{t-1} m_l^i x_l^i \; / \; \sum_{l=t-1}^{t-1} m_l^i$$

4.2.1.2 Last Temporal Observation

Since the value is missing, it will be somewhat dependent on the last occurring value, hence we consider the last temporal observation.

4.2.1.3 Empirical Spatial Mean

The nearby nodes can affect the traffic in a great way. For example, if 2 junctions are adjacent to each other, the traffic at both junctions could be similar (even after some time). Hence we take the empirical spatial mean. It is calculated using S nearby samples.

$$\bar{\bar{x}}i_t = \sum_{s=1}^{S} m_t^s x_t^s \; / \; \sum_{s=1}^{S} m_i^s$$

4.2.1.4 Nearest Spatial Observation

Particularly in a traffic graph where the surrounding nodes share comparable traffic conditions, a graph node's state generally stays similar to that of its neighbors. Using all 4 features, we calculate local features as

$$z_t^i = m_t^i x_t^i + (1 - m_t^i)(\gamma t \dot{x}_t^i + \gamma s \ddot{x}_t^i + (1 - \gamma t)\bar{x}_t^i + (1 - \gamma s)\bar{\bar{x}} i_t)$$

Where z_t^i denotes the local features, and m^i denotes the mask sequence, and since the missing values have mask value 0, it would be dependent on the local features, else it would be dependent on the value itself.

Now, the query will be constructed, using the Local features, as

$$qt = Z_t W_q + b_q \in R^{N \times d}$$

Input Xi, is generated by taking hourly, daily, and monthly data, and concatenating them to form Xi.

$$\{X_i\} = [X_h \| X_d \| X_w]$$

These inputs are embedded into input memory vectors, and output memory vectors.

$$m_i = X_i W_m + b_m \in R^{N \times d}$$
$$c_i = X_i W_c + b_c \in R^{N \times d}$$

Building an enriched traffic embedding requires careful consideration of global historical patterns. Hourly, daily, and weekly recurring portions of input. The probsparse attention method, which uses the top u queries instead of all the queries for efficient calculation of the attention score [2], is used to determine the attention score between the query (q) and memory (m_i). The resemblance of each historical observation to the inquiry is indicated by the attention score. The response vector from memory is then calculated by adding the output memory vectors and weighting them according to the input's attention score.

$$o_t = \sum_{i=1}^{(n_d+n_w+n_h)T} c_i p_t, i \in R^{N \times d}$$

After computation of this output, we may combine local spatiotemporal data with global multiscale information to produce enriched traffic embeddings:

$$h_t = (qt \| o_t) W_h + b_h \in R^{N \times d}$$

4.3 Dynamic Graph Construction

By utilising the parameters, such as the distance between the cities, etc., a pre-defined graph is formed. However, the preconfigured graph won't have much of an impact. Dynamic graphs are created with rich traffic embeddings at each spatiotemporal block, which allows effective capture of spatial interactions between nodes while taking into account missing values and the type of data in traffic data.

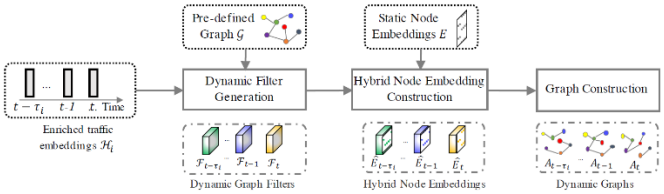

Figure 3: Dynamic Graph Construction [1]

Given h_t, the traffic embeddings at time t, the dynamic filters will be generated as

$$F_t = \sum_{k=0}^{K} P_k h_t W_k \in R^{N \times d}$$

Where, $P_k = A_G / rowsum(A_G)$

A_G is the Adjacency Matrix.

Static node embeddings refer to fixed representations of nodes in a graph, where each node is assigned a numerical vector or embedding. These embeddings capture the structural and/or semantic information about the nodes in the graph.

There are various methods to generate static node embeddings, some of which include Node2Vec [8], etc.

Now, two random static node embeddings, corresponding to source and target vectors is taken, and the dynamic filters are applied on the embeddings.

$$\hat{E}_t^1 = \tanh(\propto (F_t^1 \odot E^1)) \in R^{N \times d}$$
$$\hat{E}_t^2 = \tanh(\propto (F_t^1 \odot E^2)) \in R^{N \times d}$$

Where $\odot$ denotes hadamard product, which ensures combining of both static and dynamic features of the traffic data.

Finally, the dynamic graph is constructed using these embeddings, as

$$A_t = ReLU(\tanh(\propto (\hat{E}_t^1 \hat{E}_t^{2^T} - \hat{E}_t^2 \hat{E}_t^{1^T}))) \in R^{N \times N}$$

4.4 Temporal Convolution Module

High level temporal information can be extracted using the Temporal Convolution Module's numerous dilated convolution layers. We adopt the temporal convolution module over the enhanced traffic embeddings while taking into account the temporal dynamic data in the traffic. Hi. To output the temporal information, one dilated convolution block is followed by a tangent hyperbolic activation function. A sigmoid activation function serves as a gate to the other block, determining the percentage of information that can travel to the following module. Non-linearity is incorporated into the architecture through sigmoid gates as well.

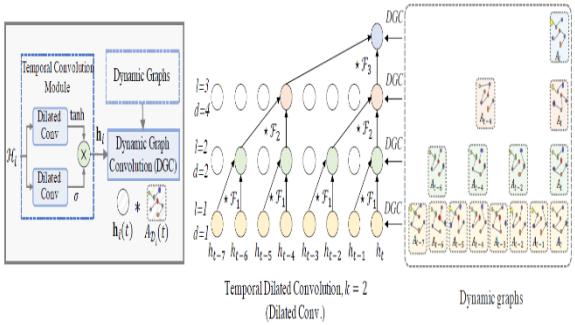

Figure 4: Temporal Convolution Module [1]

Given the traffic Embeddings Hi, a filter F, the dilated convolution operation is

$$H_i \star F_i(t) = \sum_{s=0}^{K} F_i(s) H_i(t - d \times s) \in R^{N \times d \times T_{i+1}}$$

There are 2 Temporal Convolution Blocks – 1 followed by a tangent hyperbolic function, and other with sigmoid function, as a gate to ratio of information that can pass to next module. Output of the TCN Module will be

$$h_i = \tanh(W_{F1} \star H_i) \odot \sigma(W_{F^2} \star H_i) \in R^{N \times d \times T_{i+1}}$$

4.5 Dynamic Graph Convolution

This is applied on output of temporal convolution module. Temporal features aggregate spatial information according to dynamic graphs generated. Vanishing Gradient problem will be taken care of by using Residual Connection. At each time step, the graph convolution will produce the aggregated spatial data.

$$H_i'(t) = \sum_{k=0}^{K} \left(A_{D_i}(t)\right)^k h_i(t) W_k \in R^{N \times d}$$

Residual connections are adopted to avoid the vanishing gradient problem.

The input of $(i+1)^{th}$ ST-Block is :

$$H_{i+1}(t) = H_i(t) + H_i'(t)$$

4.6 Output Forecasting Module

The output h_i from the middle of the Temporal Convolution Module, and the last output H_i will be considered for predicting the output.

$$O = (h_0 W_s^0 + b_s^0) \| \dots \| (h_i W_s^i + b_s^i) \| \dots \| (h_{l-1} W_s^{l-1} + b_s^{l-1}) \| (h_l W_s^l + b_s^l)$$

Using this, the final output is generated.

$$\hat{Y} = \left(ReLU\left(O W_{fc}^1 + b_{fc}^1\right)\right) W_{fc}^2 + b_{fc}^2 \in R^{N \times T_p}$$

The output will be predicted by outputs of the Temporal Convolution Module, and the last Spatial-Temporal Layer Output. Skip connections are added in each hidden layer. All the outputs will be concatenated to get the final output

V. EXPERIMENTS

5.1 Datasets

We perform experiments on the public traffic dataset METR-LA [4]. The METR-LA dataset contains values from 207 sensors, in 5 minutes intervals, which contains the average traffic speed of the traffic at the moment.

The METR-LA dataset is a widely recognized and extensively used traffic forecasting dataset that contains valuable traffic flow information. Collected from loop detectors installed throughout the Los Angeles County area, the dataset offers a comprehensive view of traffic patterns. With a temporal resolution of 5 minutes, it provides detailed historical traffic flow measurements and timestamps, enabling researchers to analyze and predict traffic conditions at various time intervals. Covering highways and arterial roads, the dataset encompasses over 200 loop detectors strategically placed across the road network. Each data point in the dataset includes attributes such as traffic flow measurements, occupancy, and speed, shedding light on congestion levels and traffic behavior.

To support model development and evaluation, the METR-LA dataset is typically split into training, validation, and testing sets. Researchers and practitioners in transportation and traffic engineering utilize this dataset to build and assess traffic prediction models, ultimately aiding in traffic management and transportation planning endeavors. The dataset can be accessed from its official repository [4], providing preprocessed data suitable for training and evaluating traffic forecasting algorithms.

5.2 Preprocessing

The preprocessed METR-LA dataset is divided into 70% training, 10% validation, and 20% testing. The missing values are introduced into the data by using 2 methods random, and mixed methods, with a specified miss ratio, like 10%, 20%, 60%, etc.

5.3 Evaluation Metrics

Three metrics are mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE) used to assess the correctness of each model.

$$MAE(Y,\hat{Y}) = \frac{1}{NT_p}\Sigma_{n=1}^{N}\Sigma_{t=1}^{T_p}|\hat{Y}_t^n - Y_t^n|$$

$$RMSE(Y,\hat{Y}) = \sqrt{\frac{1}{NT_p}\Sigma_{n=1}^{N}\Sigma_{t=1}^{T_p}|\hat{Y}_t^n - Y_t^n|^2}$$

$$MAPE(Y,\hat{Y}) = \frac{1}{NT_p}\Sigma_{n=1}^{N}\Sigma_{t=1}^{T_p}|\frac{\hat{Y}_t^n - Y_t^n}{Y_t^n}|$$

5.4 Execution and Parameter Settings

A learning rate of 0.001 is used to train the Graph Convolution Networks with Modified Attention Score Calculation model. L and S are set to 12 and 5 in the multi-scale memory module, respectively. nh, nd, and nw are all set to 2. The Temporal Convolution module has two dilated layers with a dilation factor of d [1, 2] in four ST blocks. The value of d, the embedding dimension, is 32.

VI. RESULTS

When the model is trained by given the input, the following output is generated.

Horizon	MAE	RMSE	MAPE
1	20.91	22.89	0.4801
2	34. 90	36.61	0.7250
3	18.82	20.70	0.4375
4	17.53	20.09	0.4167
5	30.01	31.85	0.6532
6	12.38	14.92	0.3016

7	17.37	19.58	0.4004
8	25.71	27.40	0.5671
9	16.48	19.59	0.3935
10	19.40	21.61	0.4503
11	28.20	30.56	0.6323
12	9.86	13.34	0.2765
Average	**20.96**	**19.75**	**0.45**

Table 1: Generated Evaluation Metrics when trained on Dataset with Missing Ratio 60%

VII. SUMMARY AND CONCLUSION

When this model Graph Convolution Network with modified attention score calculation is trained on the METR-LA dataset, it has achieved better accuracy than existing model GCN-M. This project successfully addressed the challenge of traffic prediction by leveraging graph convolutional networks.

The utilization of the graph structure of transportation networks through GCNs proved to be a valuable approach for capturing spatial and temporal dependencies in traffic data. This is important in forecasting the traffic status for the future and helps in efficient management and control of the traffic in any city.

This is also useful for preparing the city for the future. Overall, this project contributes to the growing body of research on traffic prediction and underscores the potential of graph convolutional networks as a powerful tool for understanding and forecasting traffic patterns. The findings pave the way for further advancements in traffic prediction methodologies and offer valuable insights for the development of intelligent transportation systems.

VIII. REFERENCES

[1]. Graph convolutional networks for traffic forecasting with missing values, Jingwei Zuo, Karine Zeitouni, Yehia Taher & Sandra Garcia-Rodriguez, published in DMKD 2022.

[2]. Informer: Beyond efficient transformer for long sequence time-series forecasting, Zhou H, Zhang S, Peng J, et al, published in AAAI 2021.

[3]. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, by Li Y, Yu R, Shahabi C, et al published in International Conference on Learning Representations (ICLR), 2018.

[4]. Graph wavenet for deep spatial-temporal graph modeling by Wu Z, Pan S, Long G, et al published in Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), pp 1907 – 1913, 2018.

[5]. Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution by Li F, Feng J, Yan H, et al published in ACM Transactions on Knowledge Discovery from Data (TKDD) , 2021.

[6]. Che Z, Purushotham S, Cho K, et al Recurrent neural networks for multivariate time series with missing values published in Scientific reports 8 – 12 , 2018.

[7]. Dataset : METR-LA, downloaded from https://drive.google.com/drive/folders/10FOTa6HXPqX8Pf5WRoRwcFnW9BrNZEIX

[8]. Node2Vec: Scalable Feature Learning for Networks by Aditya Grover, Jure Leskovec published in proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

[9]. Semi-Supervised Classification with Graph Convolutional Networks by Thomas N. Kipf, Max Welling, published in ICLR, 2017.

[10]. Deep Learning on Graphs: A Survey, by Zhang, Z., Cui, P., Zhu, W., & Wang, X. published in IEEE Transactions on Knowledge and Data Engineering, 30(12), 2321-2333., 2018.

[11]. Graph Convolutional Long Short-Term Memory Network for Traffic Flow Forecasting by Lv, Z., Zhu, Y., & Wu, L., published in IEEE Transactions on Intelligent Transportation Systems, 21(3), 1074-1084., 2020.

[12]. Traffic Prediction with Graph Convolutional Recurrent Neural Network by Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., & Wang, Y., published in the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 615-623), 2019.

Cite this article as :

Bathina Bhanukowshik , "Traffic Prediction Using Graph Convolution Networks", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 9, Issue 4, pp.57-65, July-August-2023.