



Online Transfer Learning with MLP-Assisted Graph Convolution Network for Traffic Flow Forecasting: A Solution for Edge Intelligent Devices

Jingru SUN^{†1}, Chendingying LU¹, Yichuang SUN², Hongbo JIANG¹, Zhu XIAO¹

¹College of Computer Science and Electronic Engineering, Hunan University, Changsha, 410082, People's Republic of China

²School of Engineering and Computer Science, University of Hertfordshire, Hatfield, AL10 9AB, United Kingdom

Abstract: Traffic flow prediction is crucial for intelligent transportation, aiding route planning and navigation. However, existing studies often focus on prediction accuracy improvement, while neglect external influences, and practical issues like resource constraints and data sparsity on edge devices. We propose an Online Transfer Learning with Multi-layer Perceptron (MLP) assisted Graph Convolution Network (GCN) framework (OTL-GM), consisting of two parts: transferring source domain features to edge devices and using online learning to bridge domain gaps. Experiments on 4 datasets demonstrate OTL's effectiveness, and compared with the model without OTL, convergence time of the OTL-model increased from 24.77% to 95.32%.

Key words: Online transfer learning; Traffic forecasting; Intelligent edge devices

<https://doi.org/>

CLC number: TP391.4

1 Introduction

The integration of Internet of Things (IoT) in Intelligent Transportation Systems (ITS) signifies a transformative era with unprecedented possibilities(Bojan et al., 2014). A widespread sensor network in urban traffic nodes enables real-time monitoring of variables like traffic flow and congestion, swiftly transmitting data for dynamic traffic management(Zhang and Lu, 2020). IoT's significance lies in processing data with advanced techniques, such as deep learning, providing a scientific foundation for optimized transportation systems, resulting in heightened efficiency and reduced congestion. Accurate traffic flow prediction is crucial for intelligent transportation, empowering advanced management, and optimizing resources for efficiency. How-

ever, predicting delay is a more important aspect for traffic managers and traveling vehicles(Hashemi and Abdelghany, 2015). Excessively long delays can lead to wrong decisions by traffic managers and mislead vehicles into planning their driving routes. The synergy of traffic flow prediction and IoT establishes an intelligent foundation for resilient, efficient, and secure urban transportation systems(Derawi et al., 2020). However, some contradictions have arisen during the integration of IoT and ITS.

On the one hand, IoT devices have evolved from simple magnetic and infrared sensors to the current generation of intelligent edge devices(Qadri et al., 2020). Initially, basic motion detection relied on magnetic and infrared technologies. Later, cameras were introduced to improve monitoring precision. However, integration among different sensors is still limited. Intelligent edge devices are now capable of integrating sensors, computing abilities, and connectivity to provide advanced real-time processing and

[†] Corresponding author

© Zhejiang University and Springer-Verlag GmbH Germany, part of Springer Nature 2020

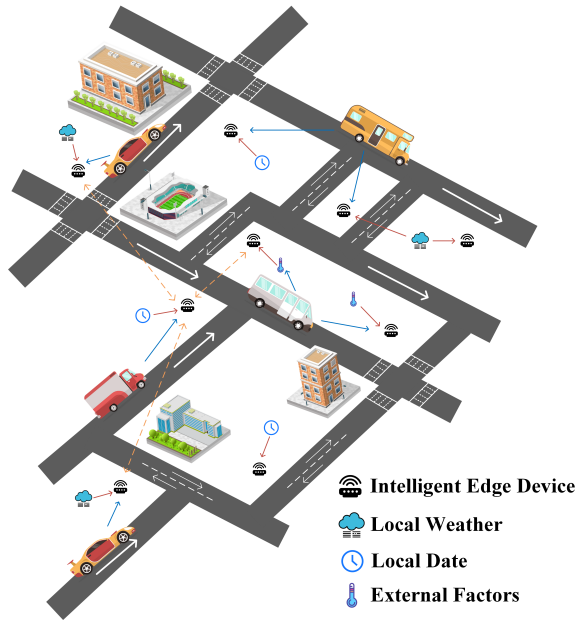


Figure 1 The real world scenario of traffic collection and prediction.

feedback functions. This integration is driving the Internet of Things (IoT) towards greater intelligence and efficiency. However, such multi-functional devices are often limited by resources, especially computing resources (Asim et al., 2020). Since they not only compute, but transmit wireless data (Cong et al., 2020) and road condition detection (Singh et al., 2021). Therefore, how to reduce computing resource consumption and provide as many computing resources as possible for other aspects has become an urgent problem to be solved. Furthermore, collecting traffic flow data can be challenging in extreme weather conditions (Datla and Sharma, 2008).

On the other hand, as the core of traffic flow prediction problems, prediction accuracy has always been a hot topic in academic research. In the 1990s, traffic flow prediction based on micro-simulation and macro-models began to attract attention (Bando et al., 1995). During the 2010s, there was increased recognition of the significance of data-driven approaches and real-time technology (Nellore and Hancke, 2016). Currently, deep learning has emerged as an essential technology for predicting traffic flow, highlighting the superiority of neural network models in handling complex spatiotemporal relationships (Kashyap et al., 2022). Various models emerged for spatiotemporal prediction, such as Spatial-Temporal Graph Convolutional Network (STGCN) (Yu et al., 2017), Diffusion Convolu-

tional Recurrent Neural Network (DCRNN) (Li et al., 2017), and a graph attention method (Tang et al., 2020), all achieving excellent results in accuracy. With the pursuit of higher accuracy, comes the increased algorithmic complexity and prediction time (Zhu et al., 2021; Diao et al., 2019; Guo et al., 2022).

The increasing complexity of algorithms contradicts the limited computing resources of IoT intelligent edge devices. In the context of traffic flow prediction, this issue becomes even more apparent.

As shown in Fig. 1, numerous intelligent transportation facilities have been deployed on urban roads presently, such as camera modules, geomagnetic modules, temperature sensors, and 5G modules for communicating with one another (as shown in Fig. 1). They can accurately ascertain the traffic flow, detect temperature, and other information. From the real-world perspective, traffic flow is affected by multiple factors, for example, the road will encounter different types of weather, which lead to differences in data distribution. When facing extreme weather conditions such as blizzards and heavy fog, people living in cities surely tend to stay in buildings rather than go out and these extreme weathers decrease the accuracy of detection algorithms e.g. YOLO, which leads to data sparsity as well. Similarly, different times of period cause changes in data distribution as well (proven in Fig. 3). The rush hours in the morning and evening can lead to a sharp increase in traffic flow, i.e. nonlinear characteristic. Also, weekends and holidays can affect people's willingness to travel. Therefore, how to handle the nonlinear characteristics and alleviate the limited resources of edge intelligent devices has become a challenge.

Traditional traffic flow prediction algorithms often take all features in the dataset into account (Zhang and Patras, 2018), while this will introduce many irrelevant data features and lead to a decrease in prediction accuracy. Only a few studies (Sun et al., 2022) address issues related to regional structuring. Based on the above analysis, there remain two challenges in traffic flow prediction: 1) limitation on computation resources and 2) the sparsity of data caused by extreme weather and the accuracy of object detection algorithms. So existing literature is insufficient to address these emerging challenges and problems in the real-world scenario. To solve these problems, we proposed an online transfer

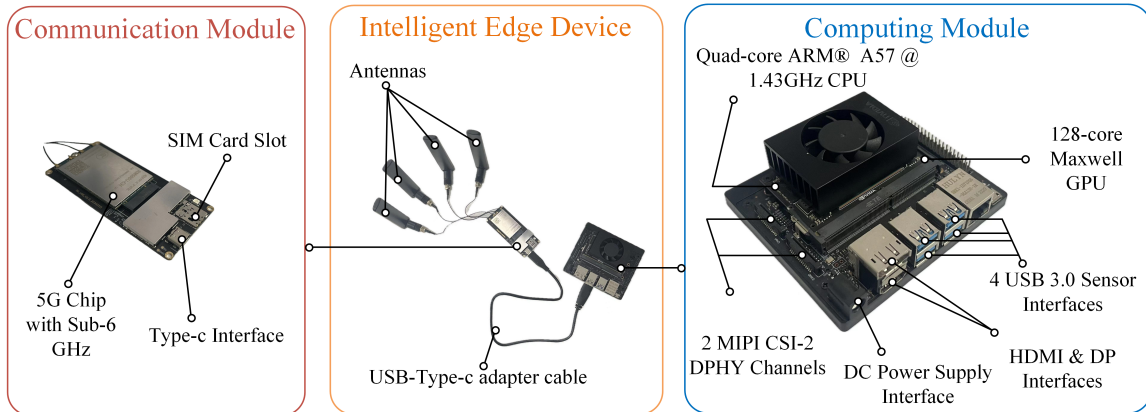


Figure 2 Intelligent edge device with 5G communication module and kinds of interfaces.

learning framework with MLP-assisted GCN (OTL-GM). The main contributions of this paper can be summarized as follows:

- To ensure the lightweight of models running on intelligent edge devices, we combined the temporal model multilayer perceptron with spatial model graph convolution network to obtain a GM model capturing spatio-temporal features.
- We combined GM with a novel Online Transfer Learning framework (OTL) for traffic forecasting, which not only relieves the tensed computational resources but also maintains prediction accuracy at current research levels due to opting for the proper models with smaller Euclidean distance.
- We validated the effectiveness of our OTL framework with other methods by conducting extensive experiments on real-world dataset PeMS04, which is classified into three datasets with different data types. Algorithm convergence time has been greatly improved. To the best of our knowledge, it is the first attempt to apply online transfer learning for traffic flow forecasting.

The remaining content of the article is organized as below. Section 2 provides some related work about online transfer learning and traffic flow forecasting with spatio-temporal correlation. Before introducing the model, the prerequisite knowledge in this paper and the definition of traffic flow prediction is introduced in detail in section 3. Section 4 mainly

introduces the details of model and algorithm. Section 5 validates the effectiveness of proposed OTL-GM and discusses the experimental results. Finally, section 6 summarizes the full paper.

2 Related work

2.1 Traffic flow forecasting

There are two main categories of methods for predicting traffic flow: model-driven approaches and data-driven approaches. Model-driven approaches use statistical theory to create a comprehensive model, which learns the time feature and analyzes the data. There are several typical model-driven approaches, such as the history average model (Liu and Guan, 2004), the auto-regressive moving average model (Williams and Hoel, 2003), and the vector auto-regressive model (Zivot and Wang, 2006). In particular, Hamed et al. (Hamed et al., 1995) consider the flow of each road aiming at obtaining a proper ARIMA model. However, model-driven approaches are insufficient in dealing with real-world traffic data as traffic flow is impacted by various factors and is nonlinear.

To tackle these issues, data-driven approaches have gained traction in recent decades. Additionally, data-driven approaches include machine learning and deep learning. As for machine learning, in time order, there are k-nearest neighbor (Van Lint and Van Hinsbergen, 2012) and support vector regression (Jeong et al., 2013). However, the fatal drawback of machine learning is excessively dependent on the setting of initial hyper-parameters i.e. prior knowledge, which determines the accuracy of

the prediction. And the complexity of traffic data in the real network is vital as well. Hence, the further literature has shifted towards research on deep learning: the current research results indicate that they can improve the accuracy of prediction facing the high complexity of traffic data and represent traffic features ignoring prior knowledge, such as deep belief network (DBN) (Jia et al., 2016), stacked auto-encoder (Lv et al., 2014) and Multi-layer Perceptron (Rumelhart et al., 1986).

2.2 Spatio-temporal correlation

Given that traffic flow forecasting is a time series prediction problem indeed. Some neural networks for time series forecasting problems are applied to traffic flow prediction, such as recurrent neural network (RNN) (Gräber et al., 2018) and the time complexity is $O(T)$, where T represents the length of the input sequence. Furthermore, to overcome the problem of vanishing gradient and exploding gradient of RNN, the derivative of RNN, such as long short-term memory (LSTM) (Zhang and Guo, 2020) neural network. Actually the time complexity of LSTM is $O(4T)$ since its three gates, but according to the rules of algorithm complexity, it is also still $O(T)$. And gated recurrent unit networks (GRU) (Sutskever et al., 2014) have been introduced with $O(T)$ time complexity. Hence, the temporal feature can be represented properly. However, LSTM's performance in predicting long sequences falls short of expectations and Transformer (Vaswani et al., 2017) was introduced in response to the long sequence challenge at the cost of the time complexity $O(T^2)$. Its parallelized self-attention mechanism enables efficient processing. However, the increased computational demands can not meet our requirement for limited computational resources. We use a multi-layer perceptron and adjusted its width and depth to reduce time complexity while ensuring a certain accuracy. The time complexity of GM is $O(\sum_{l=1}^L I_l O_l)$, where L is the number of layers, I_l is the input dimension of layer l and O_l is the output dimension of layer l . The final time complexity is constant level $O(1)$.

Actually, the traffic flow is influenced by not only time dimension but also spatial dimension, which is known as spatio-temporal correlation. To take the spatial feature into consideration, convolution neural network (CNN) (Krizhevsky et al., 2017) has been applied to extract the local feature. For

example, MGSTC (Chen et al., 2020) established multiple gated spatio-temporal CNN networks in order to extract multiple spatio-temporal dependency and S-TCN (Guo et al., 2023) is modeled with spatio-temporal convolution neural network to capture several correlation among different dimensions. However, CNN can only be applied to data structures with Euclidean properties, i.e. regular grid shapes, such as images, which is obviously not suitable for capturing spatial features in transportation networks due to complex network topology. Therefore, we adopt graph convolution network to capture the spatial features, which means dealing with non-Euclidean topology graph and reduce the parameters due to the same adjacency matrix. Excellent work in recent years includes primary GCN (Kipf and Welling, 2016), which find 2-3 layers can obtain the best results, spatial-temporal GCN (Yu et al., 2017) combined gated CNNs and GCN. And temporal graph convolution network (Zhao et al., 2019) combined GRUs and GCN. Different from these, we use a two-layer GCN and a three-layer MLP to ensure lightweight and capture the spatio-temporal features.

2.3 Transfer learning

The motivation of adopting transfer learning in some literature (Mallick et al., 2021) is to solve the problem of data sparsity due to the poor urban infrastructure. However, even with the development of cities with more intelligent roadside equipment, there will still be data sparse problems due to the existence of extreme weather such as heavy fogs and blizzards (Datla and Sharma, 2008). So we take these weather factors into account in our transfer learning. For effective transfer of knowledge from one trained model to another, we need to select a relatively similar model that has a similar data distribution. Nonetheless, transfer learning leads to prediction error to some extent. Several literatures use different methods to solve this problem. Yao et al. (Yao et al., 2023) proposes an adversarial domain adaptation to reduce the error caused by transfer learning. Kwon et al. (Kwon et al., 2021) use incremental learning to help deep learning models learn new tasks continuously without forgetting what was learned previously. Tzeng et al. (Tzeng et al., 2015) adopt approach to simultaneously optimize for domain invariance to facilitate domain transfer and use

a soft label distribution matching loss to transfer information between tasks. And we use Euclidean distance to characterize the similarity between data distributions and implement online learning to minimize transfer learning errors by utilizing intelligent roadside equipment to gather real-time data.

2.4 Online learning

Online learning has been investigated considerably during the past decades aiming at obtaining more accuracy of predicting. In this paper, we introduce the approach of online learning to eliminate the error caused by transfer learning. Much research focus on the criterion of maximum-margin such as (Li and Long, 1999; Gentile, 2000; Crammer et al., 2006). They are all devoted to minimizing the loss of algorithm. And one of the most classic algorithms is (Rosenblatt, 1958), the authors tend to constantly adjust the weight of the classifier with the latest samples. Later, a new algorithm named Double Updating Online Learning (DUOL) emerged (Zhao et al., 2011), which is designed to update not only the weights of current instances but also the weights of instances that already have been received. Recently, some of research (Han et al., 2020) adopt the approach of online learning in order to minimize the time of training model. However, these methods are inconsistent with our purpose of reducing the error caused by transfer learning. So we use real-time data collection to update the parameters of the last fully connected layer. Therefore, online learning can reduce prediction errors and training time, thus decreasing resource consumption.

3 Preliminary

This section introduces some notations and definitions that are used in the model, and defines the task of the prediction model. Some important notations are recorded in Table 1.

3.1 Traffic flow data

The traffic flow data $X_t \in \mathbb{R}^{N \times C}$ are collected by node (as shown in Fig. 1) at time step t , where the C represents the kinds of traffic data (generally speaking, it may be traffic volume, average speed or road occupancy) and N denotes the whole nodes. Considering a batch of data of length τ as input,

Table 1 Key Notations

Symbols	Description
G	The graph of the road network.
X	Data collected by the devices.
$\mathcal{X}$	A batch of collected data.
$\mathcal{Y}$	Predicting results.
τ	Length of $\mathcal{X}$.
λ	Predicting step size.
C	Numbers of kinds of data.
N	Numbers of the node in the network.
V	The set of nodes.
E	The set of edges between any two nodes.
A	Adjacent matrix of G .
T	Time dimension.
S	Spatial dimension.
$\hat{A}$	Graph matrix with added self-connection
I_N	The identity matrix
$\tilde{D}$	Degree matrix with added self-connection
D	Degree matrix
$H^{(l)}$	The output of l^{th} layer
$\theta^{(l)}$	The weight parameter
σ	Sigmoid activation function

we can obtain $\mathcal{X} = (X_{t_1}, X_{t_2}, \dots, X_{t_\tau}) \in \mathbb{R}^{\tau \times N \times C}$. After feeding the input into the network, we get the predicted results $\mathcal{Y} = (Y_{t_\tau+1}, Y_{t_\tau+2}, \dots, Y_{t_\tau+t_\lambda}) \in \mathbb{R}^{\lambda \times N \times C}$.

3.2 Road network structure

In this paper, the structure of the traffic network can be regarded as a weighted directed graph $G = (V, E, A)$, where V represents all the nodes in the road network. Generally, vertex, node, sensor, detector will be all seen as the same thing. For simplicity, we use node for the following description, which is denoted as $v_i \in V, i \in (1, 2, \dots, N)$; E is the set of the edges in the whole network, which indicates the edges between any two nodes; $A \in \mathbb{R}^{N \times N}$ denotes adjacent matrix of the graph G .

3.3 Definition of traffic forecasting

Since traffic flow prediction is essentially a time series prediction problem. Given the input $\mathcal{X} = (X_{t_1}, X_{t_2}, \dots, X_{t_\tau}) \in \mathbb{R}^{\tau \times N \times C}$, we aim at obtaining the result $\mathcal{Y} = (Y_{t_\tau+1}, Y_{t_\tau+2}, \dots, Y_{t_\tau+t_\lambda}) \in \mathbb{R}^{\lambda \times N \times C}$, where t_λ represents the predicting horizons. The

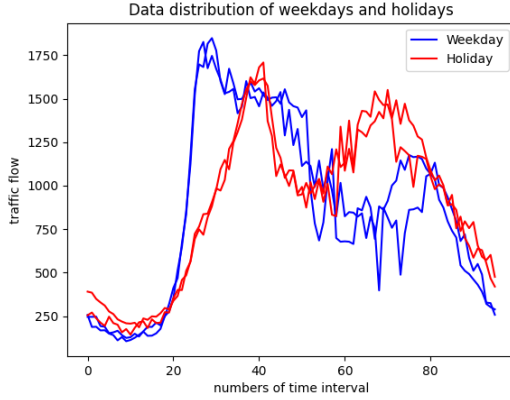


Figure 3 This dataset collects real-time traffic data every fifteen minutes, which means there are 96 collection points 24 hours a day corresponding to the numbers of time interval-axis. Traffic-flow-axis means how many cars are there in 15 minute period.

problem is defined as follows:

$$\mathcal{Y} = \arg \max_{Y_{groundTruth}} \log P(Y_{groundTruth}|\mathcal{X}), \quad (1)$$

where $Y_{groundTruth}$ represents the ground truth observed by intelligent devices and $P(\cdot|\cdot)$ is conditional probability function.

3.4 Transfer learning

In order to demonstrate the universality of the similarity in data distribution across various datasets, we extract similar data distributions in the data set (located near Heathrow Airport on the M25 highway in the UK, URL: <http://tris.highwaysengland.co.uk/detail/trafficflowdata>) and have some of the same type of data visualization. Taking weekday data and holiday data as examples, from a qualitative perspective, we find that the same type of data has similar distribution in amplitude and phase (as shown in Fig. 3), while different types of data have different phase and amplitude. From a quantitative perspective, we use Euclidean distance to determine the similarity of samples

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}, \quad (2)$$

where x_i and y_i represent different sample points at the same time, n is the total number of samples. In this case, after calculation, we get the Table II, in which the decimal part of the data has been omitted.

Table 2 Euclidean Distance Between Each Other

	Holi1	Holi2	Week1	Week2
Holi1	0	1602	3896	3582
Holi2	1602	0	4086	3739
Week1	3896	4086	0	1366
Week2	3582	3739	1366	0

Obviously, the Euclidean Distance of the same type data is much smaller than that of the different type of data. Based on the results above and the real-time interaction of the intelligent edge devices, we choose transfer learning to obtain the pretrained models and choose the most suitable one. Firstly, we train multiple models separately depending on the data type using edge servers or other servers full of computing resources. And a kind of model corresponds to a kind of data distribution. Then we obtain the signal (type of data and weather) from intelligent devices, and calculate the Euclidean Distance between the distribution of data of pretrained models and the data type represented by signal. Finally, we choose the smallest one to transfer.

3.5 Online Transfer learning

The transfer learning discussed in the previous section only involves transferring a model with the most suitable data distribution. However, the distribution of real-time data may not align closely with the transferred model, so it is necessary for smart edge devices to continuously collect real-time data and update some parameters of the model's layers to enhance prediction accuracy and timeliness.

4 Proposed model and algorithm

This section mainly introduces the details of model and algorithm steps.

4.1 The overview of GM framework

In this section, we elaborate on the proposed architecture of GM. As shown in Fig. 4, GM is composed of a spatial and a temporal model, which are used to extract spatial and temporal features respectively. The whole input $\mathcal{X} = (X_{t_1}, X_{t_2}, \dots, X_{t_r}) \in \mathbb{R}^{\tau \times N \times C}$ is divided into small batches $\mathcal{X}_B = (X_{t_1}, X_{t_2}, \dots, X_{t_B}) \in \mathbb{R}^{B \times t_B \times N \times C}$, B and t_B represent the numbers of batches and the size of each batch. The W_{Best-S} and W_{Best-T} are the parameters of the best model transferred by our

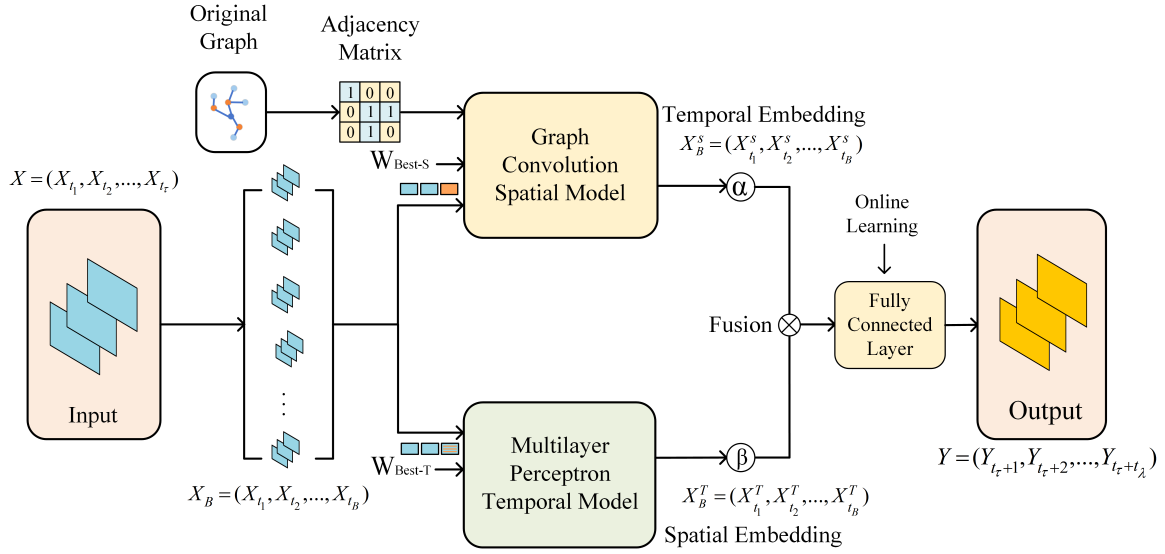


Figure 4 The whole architecture of our proposed GM.

online transfer learning framework. α and β represent the weight of time features and space features respectively. And the final predicting result $Y = \alpha Y^{Spatial} + \beta Y^{Temporal}$. The detail of each module and the online transfer learning algorithm are described as follows.

4.2 GCN for Spatial Feature

According to the previous discussions, spatial feature extraction is crucial for traffic flow prediction. And CNN aims at data with Euclidean properties, such as images and other regular grid, etc. However, the topological structure of both urban and highway structures have non-Euclidean properties, which means CNN no longer works in cases like this. Based on this problem, some literature (Kipf and Welling, 2016) gradually evolved CNN into GCN and gained much attention. GCN was originally used in many fields such as document classification (Defferrard et al., 2016) and image classification (Bruna et al., 2013). Observing the formula of graph convolution in (Kipf and Welling, 2016), we find that as long as there is an adjacency matrix and a feature matrix, a filter can be generated in the Fourier domain, which is able to extract spatial features between nodes by their first-order neighborhood. Guided by the prediction result, we can stack such filters to establish the GCN model, which is described as follows:

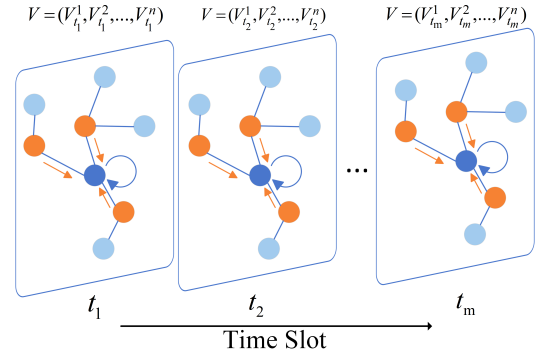


Figure 5 The process of graph convolution aggregation.

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \hat{A} \tilde{D}^{\frac{1}{2}} H^{(l)} \theta^{(l)}), \quad (3)$$

where $\hat{A} = A + I_N$ is graph matrix with added self-connection, A is the adjacency matrix of graph and I_N is the identity matrix. $\tilde{D} = D + I_N$ is D with added self-connection, $D = \sum_j \hat{A}_{i,j}$ is the degree matrix of graph, $H^{(l)}$ is the output of l^{th} layer, $\theta^{(l)}$ represents the weight parameter corresponding to that layer and $\sigma(\cdot)$ is the sigmoid function for a nonlinear activation:

$$\sigma(x) = \frac{e^x}{e^x + 1}, \quad (4)$$

As shown in Fig. 5, the notion above the figure $V = (V_{t_i}^1, V_{t_i}^2, \dots, V_{t_i}^n)$, $i \in (1, 2, \dots, m)$ represents the traffic value of each node in the graph at time t_i . Each node aggregates its first-order neighborhood

and itself every time interval, and obtain a result that only correlates with spatial features in the next time interval. Take the deep blue dot as an example, the orange node as its first-order neighborhood affects the result of it at the next time slot. And the deep blue node also affects itself. When only taking the points in the first-order neighborhood into consideration, the light blue points as the second-order neighborhood are not considered.

And for the input data $\mathcal{X} = (X_{t_1}, X_{t_2}, \dots, X_{t_\tau}) \in \mathbb{R}^{\tau \times N \times C}$, we construct an adjacency matrix $A \in \mathbb{R}^{N \times N}$ and degree matrix $D \in \mathbb{R}^{N \times N}$ based on the node relationships given by the PeMS04. According to (3), we get the output of hidden layer:

$$H^{(l)} = \sigma(\tilde{D}^{-\frac{1}{2}} \hat{A} \tilde{D}^{\frac{1}{2}} H^{(l-1)} \theta^{(l)}) \in \mathbb{R}^{\tau \times N \times C}, \quad (5)$$

and the output of GCN:

$$Y^{Spatial} = \sigma(\tilde{D}^{-\frac{1}{2}} \hat{A} \tilde{D}^{\frac{1}{2}} H^{(last-1)} \theta^{(last)}) \in \mathbb{R}^{\tau \times N \times C}. \quad (6)$$

4.3 MLP for temporal feature

As mentioned before, there are many excellent models for capturing temporal features, such as RNN, LSTM (Zhang and Guo, 2020), GRU (Sutskever et al., 2014) and Transformer (Vaswani et al., 2017). Among them, Transformer has become popular recently due to its potential to capture long-term sequences. However, the problem to be solved in this paper is the resource limitation of intelligent edge devices, and the computational complexity of Transformer is $O(n^2)$ which leads us to choose MLP with lower computational complexity $O(n)$.

Our model employs an MLP architecture primarily for its computational efficiency and lightweight nature, which are essential for real-time traffic prediction on resource-constrained devices. Unlike recurrent models such as LSTM or GRU, which are computationally expensive, our approach focuses on reducing the computational load while still capturing temporal dependencies effectively. To achieve this, we use a sliding window approach to directly encode historical traffic data into the input feature space (as detailed in Section 3.1). This method allows the model to treat temporal patterns as part of its static input features, thus avoiding the

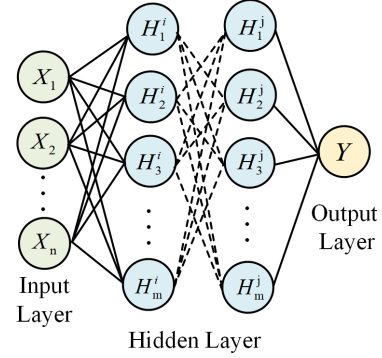


Figure 6 The temporal model with MLP.

complexity of recurrent structures while still capturing relevant time-series information. Experimental results confirm that our model achieves comparable accuracy to recurrent models, demonstrating that MLP can effectively capture temporal dependencies without the associated computational cost.

As shown in Fig. 6, H_m^j and H_m^i represent node values of each hidden layers, i and j represents the i th and j th hidden layers. For the input data $\mathcal{X} = (X_{t_1}, X_{t_2}, \dots, X_{t_\tau}) \in \mathbb{R}^{\tau \times N \times C}$, which represents the temporal information of the node. We divide it into $N \times \mathcal{X}^{MLP} = (X_{t_1}, X_{t_2}, \dots, X_{t_\tau}) \in \mathbb{R}^{\tau \times C}$ as the input of MLP. With $H^{(k)}$, $W^{(k)}$ and $b^{(k)}$ denoting the node values, parameter matrix and bias parameter of the k th hidden layer, we can obtain the output of the first hidden layer as follows:

$$H^{(1)} = \text{ReLU}(X^{MLP} W^{(1)} + b^{(1)}) \in \mathbb{R}^{\tau \times C}. \quad (7)$$

And the output of l th hidden layer is as follows:

$$H^{(l)} = \text{ReLU}(H^{(l-1)} W^{(l)} + b^{(l)}) \in \mathbb{R}^{\tau \times C}, \quad (8)$$

where $\text{ReLU}(\cdot)$ represents the activation function:

$$\text{ReLU}(x) = \max(x, 0). \quad (9)$$

Then we get the output of MLP Y :

$$Y^{MLP} = \text{ReLU}(H^{(last-1)} W^{(last)} + b^{(last)}) \in \mathbb{R}^{\tau \times C}. \quad (10)$$

Finally, we concatenate all the N outputs:

$$Y^{Temporal} = \text{Concat}(Y_1^{MLP}, Y_2^{MLP}, \dots, Y_n^{MLP}) \in \mathbb{R}^{\tau \times N \times C}. \quad (11)$$

Algorithm 1 Pre-trained Task

```

1: Input: sourceLoader, sourceModel, criterion,
   optimizer, epochs
2: Output: convergenceTime, loss
3: startTime  $\leftarrow$ 
4: for epoch  $\leftarrow$  1 to epochs do
5:   for each (inputs, labels)  $\in$  sourceLoader do
6:     optimizer.zero_grad()
7:     outputs  $\leftarrow$  sourceModel(inputs)
8:     loss  $\leftarrow$  criterion(outputs, labels)
9:     loss.backward()
10:    optimizer.step()
11:   end for
12: end for
13: endTime  $\leftarrow$ 
14: convergenceTime  $\leftarrow$  endTime - startTime
15: return convergenceTime, loss

```

4.4 Details of online transfer learning algorithm

As shown in Algorithms 1 and Fig. 7, we first feed and train the model with the classified data sets (as will be described in Section 5) to obtain the model convergence time and performance metrics *loss*, which includes MAE, MAPE, RMSE. All the Pre-trained Model corresponds to Fig. 4 with different parameters.

Then, as shown in Algorithm 2, based on the signals collected by the intelligent edge device in real time, the device calculate the Euclidean distance between the collected data type and the pre-trained model. Select a pre-trained model corresponding to the smallest Euclidean distance to transfer. This step corresponds to Pre-trained Model to Transfer and Opting Model in Fig. 7. After the transferred model loading into the current model, we freeze the parameters of feature extraction layer(i.e. all layers except fully connected layer), which means online learning is used at the fully connected layer in Fig. 4. Then adjust the learning rate of the fully connected layer and continuously run the online learning algorithm in the current environment to obtain the model convergence time and *loss*. This corresponds to the Real-time Data to Online Learning Model in Fig. 7.

The *startTime* $\leftarrow$ and *endTime* $\leftarrow$ represent to obtain the starting/ending time before/after the algorithm converges. Due to space limitations, we will write smallest Euclidean distance as *smallestED*.

Algorithm 2 Online Transfer Learning Task

```

1: Input: targetLoader, transferredModel,
   epochs criterion, optimizer, curModel, signal
2: Output: convergenceTime, loss
3: EuclideanDistance()  $\leftarrow$  signal
4: transferredModel  $\leftarrow$  smallestED
5: curModel.loadStateDict(
6:   transferredModel.stateDict())
7: parametersFreezingExceptFinalLayer
8: optim.Adam(finalLayer.parameters())
9: startTime  $\leftarrow$ 
10: for epoch  $\leftarrow$  1 to epochs do
11:   for each (inputs, labels)  $\in$  targetLoader do
12:     optimizer.zero_grad()
13:     outputs  $\leftarrow$  curModel(inputs)
14:     loss  $\leftarrow$  criterion(outputs, labels)
15:     loss.backward()
16:     optimizer.step()
17:   end for
18: end for
19: endTime  $\leftarrow$ 
20: convergenceTime  $\leftarrow$  endTime - startTime
21: return ConvergenceTime, loss

```

And *lr* is the learning rate.

5 Experiments**5.1 Dataset**

- **Traffic Data:** The datasets are collected by the Caltrans Performance Measurement System (PeMS) in real time every 30 seconds. The traffic data are aggregated into every 5-minute interval from the raw data. This paper verified the effectiveness of OTL-GM on the PeMS04 dataset provided by Los Angeles, California, USA. PeMS continuously collected data by the sensor for 59 days from January 1, 2018, which contains data from 307 nodes. And sensors collected data every five minutes. So there are 16992 records in total. And three-dimensional data features are flow, occupy and speed respectively. According to the visualization results, this paper adopt flow to perform experiments. We divide it into two parts, 45 days for training and 14 days for test. Based on the above, we start to reorganize PeMS04 into classified datasets.

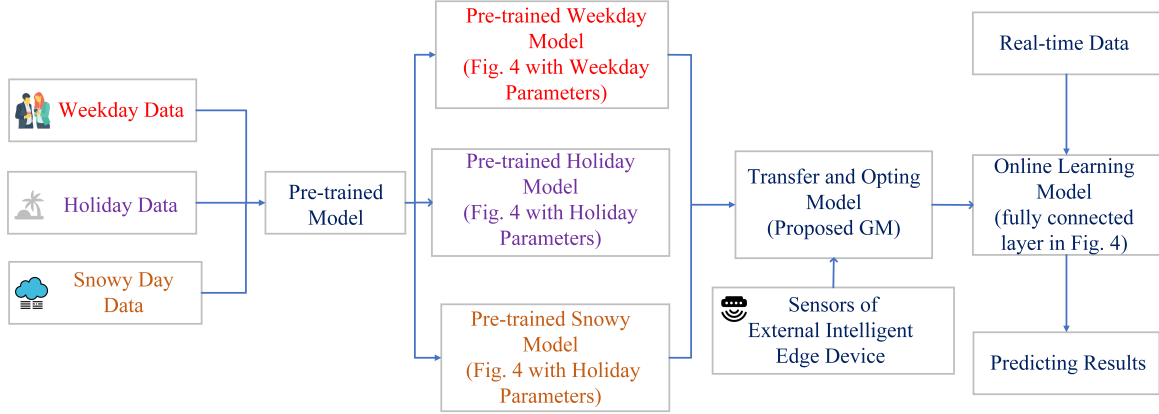


Figure 7 The online transfer learning framework and the data flow.

- The date type: Based on the Google Calendar (<https://calendar.google.com>), we get the weekend and weekday dates for the above 59 days. And we reassemble PeMS04 into weekday and weekend datasets. There are 16 weekend days and 43 working days.
- The weather type: Based on the National Weather Service (<https://www.weather.gov>), entering the downtown area of Los Angeles, we get the actual snowy days during the 59 days, they are January 25th, February 18th, 19th, 20th, 26th and 27th.

For ablation experiments, when the online transfer learning algorithm is not used, the original data set will be disrupted on a daily basis, and use a dataset with the same sample length as the classified dataset using the online transfer learning algorithm.

5.2 Experiment setting

The experiment is carried out in the following hardware configuration. As shown in Fig. 1, the edge intelligent device NVIDIA JETSON NANO B01 is equipped with an ARM (R) Quad-core A57 CPU @ 1.43 GHz and 128 core Maxwell GPU 4GB 64-bit LPDDR4. We adopt the MSE function as loss function and the Adam optimizer as optimizer. The hyper-parameter settings are as follows: To ensure the efficiency of model training, the learning rate of Adam optimizer is 0.01, the batch size of input data is 64 and the training epoch is based on the convergence time of various data sets. The hyper-parameters involved in OTL-GM mainly include the numbers of MLP layers Num_M , the numbers of GCN

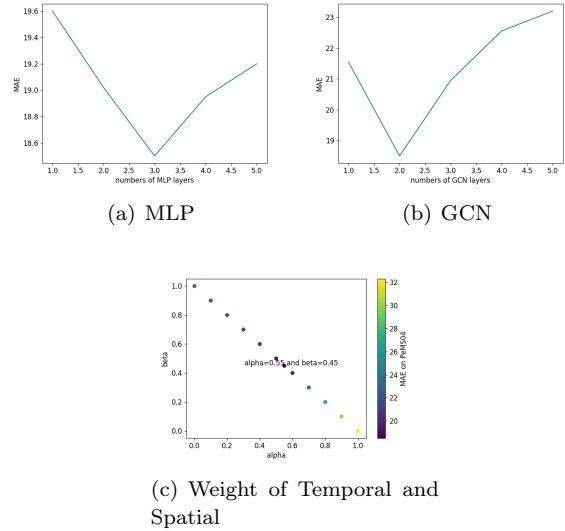


Figure 8 Parameter adjustment process on the validation set.

layers Num_G , the weight of spatio features α and temporal features β . These parameters are adjusted on the validation set to achieve the best performance of the OTL-GM ($Num_M = 3$, $Num_G = 2$, $\beta = 0.45$, $\alpha = 0.55$).

As shown in Fig. 8, the numbers of MLP layer and GCN layer achieve the best performance when they are 3 and 2; the performance reaches the best when the spatial and temporal feature weight α and β are 0.55 and 0.45.

5.3 Evaluation metrics

Since traffic flow forecasting is essentially a regression problem, we use the classical indicators of regression problems to evaluate the effectiveness of

the proposed method. They are Mean Absolute Error (MAE), Mean Absolute Error Percent (MAPE), and Root Mean Square Error (RMSE), which are defined as:

$$MAE = \frac{1}{TS} \sum_{j=1}^S \sum_{i=1}^T |y_{ij} - x_{ij}|, \quad (12)$$

$$MAPE = \frac{1}{TS} \sum_{j=1}^S \sum_{i=1}^T \left| \frac{y_{ij} - x_{ij}}{y_{ij}} \right|, \quad (13)$$

$$RMSE = \sqrt{\frac{\sum_{j=1}^S \sum_{i=1}^T (y_{ij} - x_{ij})^2}{TS}}, \quad (14)$$

where x_{ij} is the ground truth and y_{ij} is the predicting results. And T represents the time dimension, while S stands for spatial dimension.

The metric *Time* records the total time from training to convergence of the model:

$$Time = Time_{end} - Time_{start}, \quad (15)$$

where $Time_{start}$ stands for the real time before training and $Time_{end}$ represents the real time after convergence.

5.4 Ablation study of proposed GM

In this part, we conduct a comprehensive ablation experiment to verify the performance of GM. This part does not involve the comparison of algorithm usage for online transfer learning. The following baselines are widely used for time series forecasting or proposed in recent years and all hyperparameters have been tuned to their optimal values:

- **History Average (HA)**: which uses the average of historical observations to estimate future observations.
- **Vector Auto-Regression (VAR)**: which is a time series model designed to capture spatial correlations among various traffic series. (Zivot and Wang, 2006)
- **DCRNN**: which integrates graph convolution with the diffusion process and combines Graph Convolutional Networks (GCN) with recurrent models in an encoder-decoder fashion for multi-step prediction. (Li et al., 2017)

- **STGCN**: which is a spatio-temporal graph convolutional network that utilizes Graph Convolutional Networks (GCN) and temporal convolution to capture spatial and temporal correlations, respectively. (Yu et al., 2017)
- **AGCRN**: which is designed to capture fine-grained spatial and temporal correlations in traffic series automatically based on the two modules and recurrent networks. (Bai et al., 2020)
- **Cy2Mixer**: which is a novel spatiotemporal GNN based on topological non-trivial invariants of spatio-temporal graphs with gated multi-layer perceptrons. (Choi et al., 2024)
- **STAEformer**: which can yield outstanding results with vanilla transformers. (Liu et al., 2023)

The experimental results of MAE, MAPE, RMSE over 12 predictive steps on PeMS04 are shown in Table 3.

Table 3 Metrics of All Methods on PeMS04

	MAE	MAPE(%)	RMSE
HA	38.85	28.22	58.86
GCN	32.30	32.40	46.08
VAR	24.02	17.55	37.21
MLP	23.32	15.52	35.07
DCRNN	21.56	14.88	34.85
STGCN	21.32	13.83	35.02
AGCRN	19.16	12.56	31.56
Cy2Mixer	18.14	12.13	28.22
STAEformer	18.22	12.85	28.85
GM	18.50	13.41	27.95

In Table 3, we can see that the deep learning method is better than the traditional model-driven method in terms of accuracy, and there is a little difference in convergence time between deep learning methods. As mentioned before, the accuracy of a model that only considers temporal or spatial characteristics is not as accurate as a model that considers both spatio-temporal characteristics, which is well reflected in GCN, MLP, and GM. This also explains why most studies now consider both temporal and spatial features. The results also show that on the dataset PeMS04, GM has reached the current research level, which enables the proposed GM to play a role in online transfer learning.

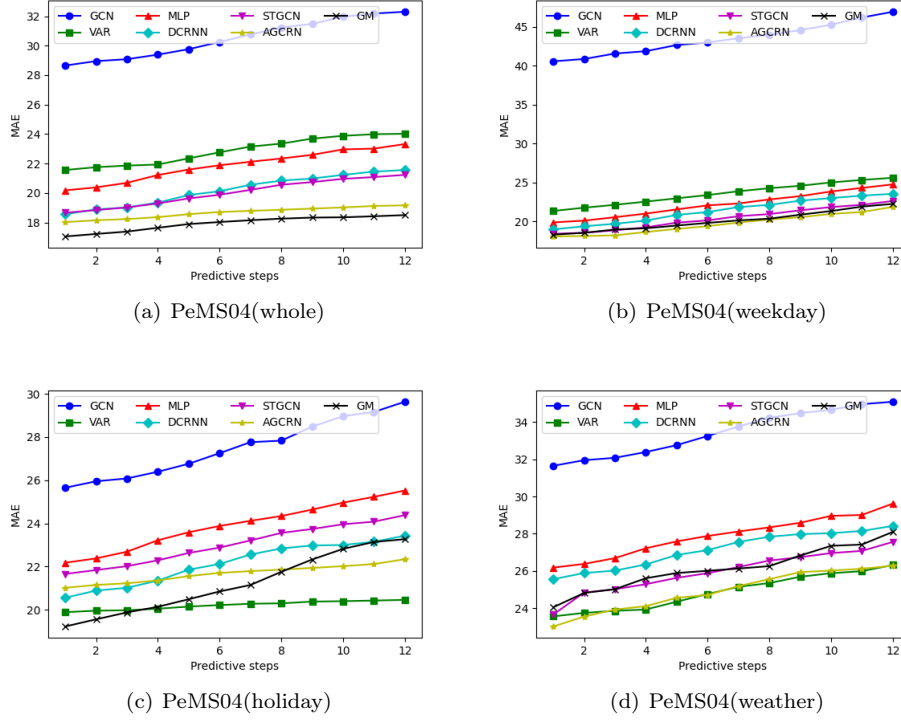


Figure 9 Accuracy Comparison as the Prediction Horizon Varies.

We can see that the metrics of proposed GM is not the best, but close to the best. And We will explain the reason by combining Table 4 and Fig. 8.

Table 4 Time and Parameters of All Methods on PeMS04

	Time(s)	Parameters
HA	125.58	-
GCN	134.58	9.12×10^4
VAR	152.25	9.86×10^4
MLP	155.55	2.85×10^5
DCRNN	164.42	3.88×10^5
STGCN	178.85	4.21×10^5
AGCRN	180.56	7.44×10^5
Cy2Mixer	280.43	6.28×10^6
STAEformer	240.98	4.37×10^6
GM	158.81	2.57×10^5

As shown in Table 4, some of the latest papers have indeed improved in accuracy, but at the expense of algorithm complexity (discussed in Section 2.2) and algorithm convergence time. GCN, VAR and MLP do also reach the low convergence time and parameters, but they fail to reach the current level of accuracy as mentioned above. However,

our proposed GM takes the spatio-temporal correlation into account while simplifies the models and number of layers to do a trade-off between accuracy and time complexity. As shown in Fig. 8 (a) and (b), on one hand, increasing the model complexity can improve accuracy on the training set; on the other hand, overly complex models can lead to overfitting and increase the algorithm's convergence time, which contradicts our goal of a lightweight model.

5.5 Stability study of proposed GM

As shown in Fig. 9, when the prediction horizon increases, the predictive accuracy of each method on every dataset decreases, which is consistent with the trends observed in deep learning. However, when the prediction horizon reaches 12 steps (1 hour), our proposed model GM maintains a certain degree of superiority in predictive accuracy compared to other baseline models. This provides more time for IoT traffic managers for effective path planning and navigation. And when the horizon reaches 1 step (5 mins), all the accuracy improved and proposed GM still holds among the best in PeMS04(whole) and PeMS04(holiday). Therefore, proposed GM can

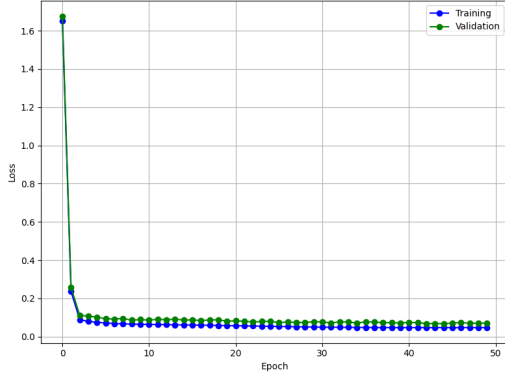


Figure 10 Loss During Training and Validation Across Epochs.

achieve both short-term traffic flow prediction and long-term traffic flow prediction.

To further evaluate the model's training dynamics, Fig. 10 shows the training and validation loss across epochs. This analysis provides insight into the potential risk of overfitting and the convergence behavior of the model. During training, we monitored the loss values for both training and validation datasets at each epoch. The loss function used was normalized MSE. And the model converges quickly and the loss of the validation set is slightly higher than that of the training set.

5.6 Ablation study of proposed Transfer Learning

For ablation experiments in transfer learning, We compared the convergence time of algorithms using transfer learning and not using transfer learning under varying degrees of data sparsity. Taking data set PeMS04(weekday) with our proposed GM as an example, the results are shown in Table 5.

Table 5 Convergence Time Under Varying Degrees of Data Sparsity

Degrees	Time with TL(s)	Time without TL(s)
10%	4.41	60.54
20%	4.52	68.22
30%	4.93	75.91
40%	5.68	123.58
50%	6.72	None
60%	8.64	None
70%	11.56	None

We can see that algorithm convergence time with transfer learning slowly increases as data spar-

sity increases, while algorithm convergence time without transfer learning increases dramatically. More importantly, when the data sparsity reaches 50%, the algorithm without transfer learning no longer converges, which results in traffic flow predictions being unavailable.

However, one problem with transfer learning is that when the local data distribution changes significantly after migration, the prediction accuracy will decrease. Therefore, we proposed an online learning method to continuously update certain parameters of the model through real-time collected data to eliminate this decline in accuracy caused by differences in data distribution.

5.7 Time and Space Complexity Analysis

Due to the use of adjacency matrix storage, the space complexity of GCN is $O(N)$, where N represents each node in the actual transportation network and the time complexity of each layer of GCN is $O(N)$, so it is still $O(N)$ after ignoring the constants. The space and time complexity of a VAR model primarily depend on the number of time steps T , the number of nodes N and the lag order P . In particular, the time complexity is $O(T^2 N^2 P^2)$, and the space complexity is $O(PN^2 + TN)$. Since P and T are treated as constants, the final time complexity is $O(N^2)$ and space time is $O(N^2)$. Since MLP is a time model, the time and space complexity is related to the input data scale, so in this article the time and space complexity is $O(1)$. DCRNN consists of Diffusion Convolution and RNN. After ignoring constants and lower-order variables, space complexity of DCRNN is $O(N^2)$ and time complexity is $O(N)$. STGCN consists of GCN and GRU, and the time complexity is $O(N)$, space complexity is $O(N^2)$. AGCRN consists of GCN, Adaptive Graph Convolution and RNN. Time complexity is $O(N)$ and space complexity is $O(N^2)$. Cy2Mixer consists of GCN and Mixer attention and time complexity is $O(N)$, space complexity is $O(N^2)$. STAEformer consists of Self-Attention, Spatio-Temporal Self-Attention, Transformer and Position Coding. And the space and time complexity are both $O(N^2)$. And our proposed GM uses sparse matrix to store traffic nodes, so the space complexity is $O(N)$. And the MLP have been discussed above, so the time complexity is $O(1)$.

According to the above analysis we can see that GM offer significant advantages in handling resource

Table 6 Performance and Inference Time Comparison

Methods	MAE	MAPE(%)	RMSE	Time(s)
ST-GFSL	18.41	16.56	29.13	57.43
ST-DAAN	18.33	15.15	28.00	65.43
OTL-GM	18.38	15.14	28.18	2.20

limited issue like traffic flow prediction on edge intelligent devices due to their efficient space and time complexities.

5.8 Comparative Analysis with Transfer Learning Approaches

To further evaluate our method, we conducted experiments comparing it with two state-of-the-art transfer learning techniques:

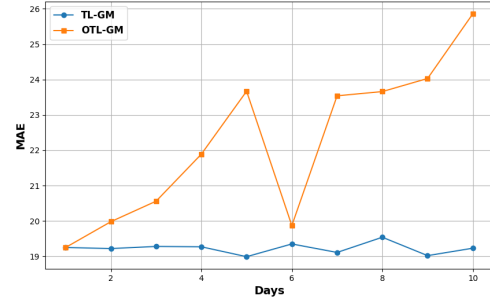
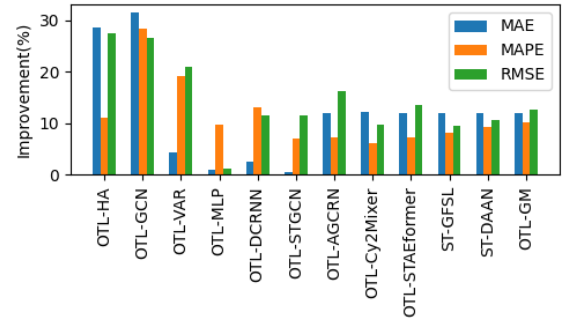
- **ST-GFSL**: which proposes to generate non-shared parameters based on node-level meta knowledge.(Lu et al., 2022)
- **ST-DAAN**: which maps the raw spatio-temporal data of source domain and then domain adaptation is adopted.(Wang et al., 2022)

The experiments were conducted on the PeMS04(holiday), using the same training/test split and evaluation metrics. Table 6 summarizes the results, where our method achieves comparable performance while maintaining lower convergence time i.e. lower computational cost.

The results demonstrate that while transfer learning methods like (Lu et al., 2022) and (Wang et al., 2022) achieve high prediction accuracy, they require substantial computational resources. (Lu et al., 2022) involves weighted matrix calculation parameters and reconstructed graphs and (Wang et al., 2022) involves 3D convolution. While our approach, on the other hand, achieves a balance between accuracy and efficiency by just calculating Euclidean distance and adjusting the last fully connected layers, making it more practical for real-world deployment in resource-constrained scenarios.

5.9 Ablation study of proposed Online Transfer Learning

In the previously mentioned transfer learning, the problem of data sparsity is improved. However, if the algorithm only uses transfer learning, the accuracy of predictions will decrease when the distribution of data changes significantly over time.

**Figure 11 Comparison between transfer learning and online transfer learning****Figure 12 Improvement of metrics on PeMS04(weekday).**

Therefore, it is necessary to use online learning to continuously collect real-time data to update certain parameters of the model to maintain the accuracy of predictions.

This part of the experiment was completed when the data sparsity was 30%. We still use the GM we proposed to conduct experiments on data set PeMS04(weekday). The results are shown in Fig. 11

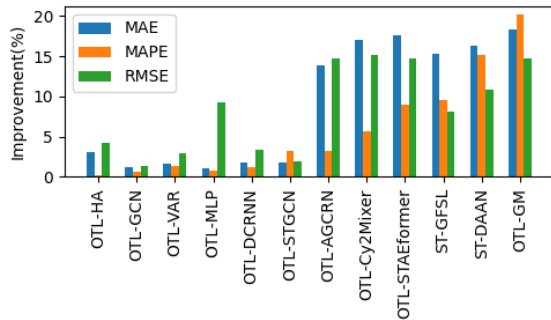
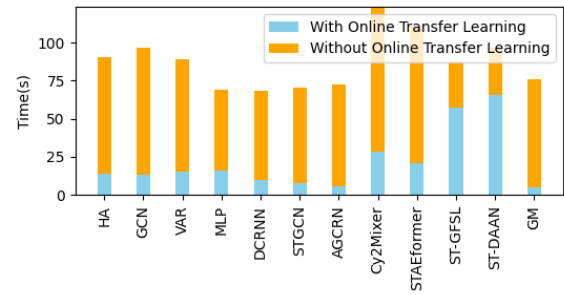
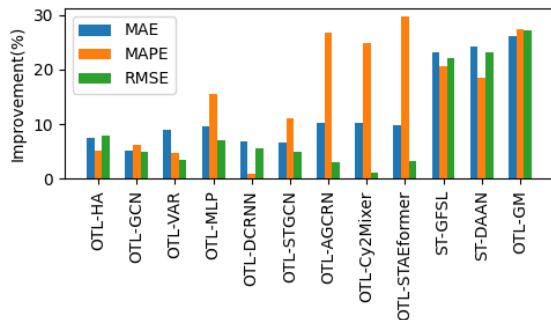
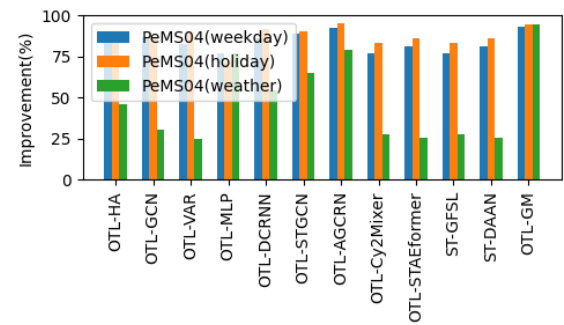
As time goes by and the data distribution changes continuously, solely relying on transfer learning can lead to a gradual decline in prediction accuracy over the overall trend. However, the incorporation of online learning helps maintain the accuracy at a consistent level. It is worth noting that on the sixth day, the accuracy of transfer learning is close to that of the first day, which is due to the Euclidean distance between the data distribution on the sixth day and the first day being very small.

5.10 All results of methods with Online Transfer Learning

As shown in Fig. 12, Fig. 13, Fig. 14 and Table 7, for the improvement of predicting metrics (MAE, MAPE and RMSE), the analysis of all the methods

Table 7 Metrics of All Methods on the Weekday, Holiday and Weather of PeMS04

	PeMS04(weekday)				PeMS04(holiday)				PeMS04(weather)			
	MAE	MAPE(%)	RMSE	Time(s)	MAE	MAPE(%)	RMSE	Time(s)	MAE	MAPE(%)	RMSE	Time(s)
HA	48.42	38.55	65.43	90.35	33.57	27.99	53.11	48.85	40.52	48.23	60.23	26.16
GCN	46.92	36.75	62.69	96.90	29.63	32.40	42.29	32.33	35.09	27.73	49.77	19.17
VAR	25.62	21.58	38.02	89.26	20.46	17.25	31.18	48.37	26.32	27.47	39.22	20.63
MLP	24.77	18.36	36.28	69.38	25.52	18.60	38.29	21.48	29.62	36.57	39.03	12.14
DCRNN	23.56	16.28	35.62	68.58	23.43	17.43	37.24	42.24	28.42	32.46	36.12	18.33
STGCN	22.64	15.08	36.64	70.58	24.38	16.42	38.25	40.25	27.56	31.45	36.14	19.58
AGCRN	21.88	13.43	35.21	72.33	22.34	15.52	34.26	50.25	26.28	29.43	32.48	20.69
Cy2Mixer	21.74	13.08	33.44	123.56	21.85	15.18	33.18	80.33	25.98	30.14	38.23	25.48
STAEformer	21.78	12.96	34.02	111.35	21.98	15.27	34.01	76.48	26.03	29.00	38.88	23.19
GM	22.27	15.69	33.28	75.91	23.27	18.97	33.07	39.25	34.60	35.93	49.74	19.37
OTL-HA	34.52	34.23	47.44	14.02	32.54	27.93	50.85	5.56	37.49	45.72	55.42	14.02
OTL-GCN	32.12	26.27	45.96	13.00	29.25	32.21	41.73	4.12	33.28	26.01	47.42	13.28
OTL-VAR	24.52	17.43	30.02	15.52	20.12	17.02	30.25	5.02	23.98	26.17	37.85	15.52
OTL-MLP	24.55	16.58	35.86	16.01	25.23	18.45	34.74	5.93	26.77	31.08	36.23	2.79
OTL-DCRNN	22.98	14.18	31.52	9.65	23.01	17.21	35.98	3.58	26.47	32.18	34.12	8.32
OTL-STGCN	22.53	14.02	32.28	7.56	23.95	15.89	37.52	3.99	25.74	27.93	34.38	6.78
OTL-AGCRN	19.24	12.44	29.51	5.43	19.25	15.03	29.21	2.35	23.58	21.49	31.51	4.3
OTL-Cy2Mixer	19.10	12.28	30.18	28.12	18.14	14.33	28.20	13.34	23.33	22.65	37.78	18.63
OTL-STAEformer	19.19	12.01	29.41	20.55	18.23	14.02	29.01	10.56	23.46	20.38	37.62	17.32
ST-GFSL	19.23	13.89	29.34	113.35	18.41	16.56	29.13	57.43	23.38	24.51	36.47	68.43
ST-DAAN	19.14	13.31	29.28	98.72	18.33	15.15	28.00	65.43	23.35	23.31	34.58	80.72
OTL-GM	19.25	14.09	29.04	4.93	18.38	15.14	28.18	2.20	25.54	26.09	36.26	1.10

**Figure 13 Improvement of metrics on PeMS04(holiday).****Figure 15 Comparison of convergence times of all methods on PeMS04(weekday).****Figure 14 Improvement of metrics on PeMS04(weather).****Figure 16 Improvement of convergence time on all the classified datasets.**

and datasets are as follows (the data sparsity is also 30%):

- **PeMS04(weekday)**: As shown in Fig. 12, the accuracy metrics of each method has been improved to a certain extent. For the MAE, the accuracy has increased from 0.48% (OTL-STGCN) to 31.54% (OTL-GCN); for the MAPE, the accuracy has increased from 7.03% (OTL-STGCN) to 28.51% (OTL-GCN); for the RMSE index, the accuracy has increased from 1.16% (OTL-MLP) to 27.50% (OTL-HA). The proposed method OTL-GM has an improvement of 13.56%, 10.20% and 12.74%, respectively.
- **PeMS04(holiday)**: As shown in Fig. 13, the results are similar to the above. For the MAE, the accuracy has increased from 1.13% (OTL-MLP) to 18.26% (OTL-GM); for the MAPE, the accuracy has increased from 0.21% (OTL-HA) to 20.19% (OTL-GM); for the RMSE index, the accuracy has increased from 1.32% (OTL-GCN) to 15.10% (OTL-Cy2Mixer). The proposed method OTL-GM has an improvement of 18.26%, 20.19% and 14.78%, respectively.
- **PeMS04(weather)**: As shown in Fig. 14, the results are similar to the above as well. For the MAE, the accuracy has increased from 5.16% (OTL-GCN) to 26.18% (OTL-GM); for the MAPE, the accuracy has increased from 0.86% (OTL-CGN) to 29.72% (OTL-STAEformer); for the RMSE index, the accuracy has increased from 3.00% (OTL-AGCRN) to 27.10% (OTL-GM). The proposed method OTL-GM has an improvement of 26.18%, 27.38% and 27.10%, respectively.

Compared to the entire PeMS04 dataset, the sample size of three classified datasets decreased (43, 16, and 6 days, respectively), and the accuracy of all algorithms decreased, which is in line with the rules of deep learning. Based on the above results, we can draw the conclusion that, compared to merging heterogeneous datasets with chaotic distributions, utilizing online transfer learning algorithms to merge datasets with similar data distributions, i.e. the smaller Euclidean distance, has led to a certain degree of improvement in predictive accuracy, which makes the prediction results valid. **And the most important point is that our proposed OTL-GM achieves**

the best performance in terms of algorithm convergence time.

Taking Fig. 15 as an example, the convergence time of all methods on PeMS04(weekday) has gained considerable improvement. As shown in Fig. 16, we detail the convergence time of all algorithms, the analysis of all the datasets is as follows:

- **PeMS04(weekday)**: The convergence time of all algorithms has been significantly improved, increasing from 76.92% (OTL-MLP) to 93.51% (OTL-GM).
- **PeMS04(holiday)**: The convergence time of all algorithms has seen a substantial enhancement, progressing from 72.40% (OTL-MLP) to 95.32% (OTL-AGCRN).
- **PeMS04(weather)**: The convergence time for all algorithms has experienced a substantial improvement, elevating from 24.77% (OTL-VAR) to 94.32% (OTL-GM).

Through an analysis combining predictive accuracy and algorithm convergence time, we observe that algorithms adopting online transfer learning not only enhance accuracy due to the similarity in learned data distributions but also significantly reduce convergence time, which can be attributed to pre-trained tasks algorithm.

As we mentioned before, many academic papers on traffic flow prediction nowadays focus on improving accuracy, often achieved by introducing complex components (Transformer) or increasing algorithm complexity. While accuracy is undoubtedly crucial for traffic flow prediction, the practical scenarios need to take the limited resources of devices into account. The model and methods we propose aim to maintain accuracy at the current research level while, to some extent, reducing algorithm convergence time and increasing the practicality of traffic forecasting.

Although we perform the pre-trained tasks on our intelligent devices (as shown in Fig. 1), in the real-world scenario, pre-training on the edge servers with rich computing resources is a very good choice. The online transfer learning with smaller Euclidean distance addresses, to some extent, the issue of data sparsity resulting from extreme weather conditions, and the shortened algorithm convergence time addresses limitations in computational resources on intelligent edge devices at the same time. Hence, we

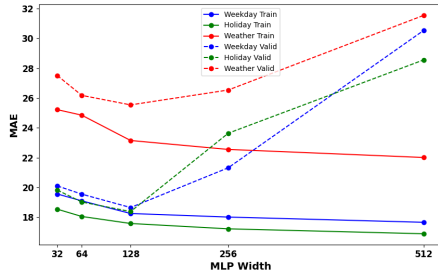


Figure 17 Variation of MAE with Different MLP Widths.

have addressed the issues of data sparsity and limited computational resources as initially posed in the introduction, rendering traffic flow prediction more practically meaningful and closer to real-world applicability. And the online transfer learning method we proposed can not only be applied to the GM, but future advanced models proposed thereafter.

5.11 Sensitivity analysis to the width of the MLP

Given that it was already mentioned in section 5.2, the optimal depths for GCN and MLP are 2 and 3. And the width of GCN is the number of traffic nodes, so this section will discuss the sensitivity of MLP width.

In Fig. 17, we can observe that as the MLP width increases, the results on the training set gradually improve. However, for the PeMS04(weekday) and PeMS04(holiday) datasets, when the MLP width exceeds 128, the results on the test set begin to increase, indicating overfitting. In contrast, for the PeMS04(weather) dataset, since we only collected a small amount of data (only 6 days), overfitting occurs when the MLP width exceeds 64. With sufficient data, our MLP model can generalize well to traffic flow prediction under different conditions.

5.12 Analysis of scalability and performance of GM under operational conditions.

In real-world applications, scalability is a crucial aspect of deploying the OTLGM framework. As traffic grow in size and the time steps increases, which may introduce significant computational challenges. However, the OTLGM framework is designed to handle such scenarios efficiently.

As shown in the Table 7, the data volume of the three data sets PeMS04(weekday), PeMS04(holiday)

and PeMS04(weather) is gradually increasing. Judging from the time growth, from small to large, they are 1.10s, 2.20s and 4.93s respectively. The amount of data in PeMS04(weekday) is 7 times that of PeMS04(weather), but the time has only increased by about 4 times. So GM is well-suited for transportation network with increasing data volume. And as shown in Fig.9, when the step size increases, the accuracy gradually decreases, but it is still better than most methods and maintains a current level.

Data sparsity is one of the goals of this article. We explored the convergence time of the GM algorithm under different data sparsity conditions. In Table 5, we can see that as data sparsity increases, algorithm convergence time also increases. When the data sparsity reaches 50%, GM is no longer able to converge. However, transfer learning can very well make up for the sparseness of real-time data.

6 Conclusion

In this paper, within the context of the development of the IoT, we address the challenges of traffic flow prediction in the real world by highlighting two issues: the limited computational resources of intelligent edge devices and the data sparsity resulting from extreme weather conditions. Utilizing Euclidean distance, we verify the similarity in distribution of the same type of data, which confirms the effectiveness of transfer learning. Considering the constraint of limited computational resources, we select GCN models capable of capturing spatial features effectively and MLP models with lower complexity. We propose a novel online transfer learning algorithmic framework, integrating our proposed models and methods. Extensive validation experiments and ablation studies on the real-world PeMS04 dataset confirm the effectiveness of our proposed OTL-GM, addressing the limitations of computational resources and data sparsity on intelligent edge devices. This enhances the practical significance of traffic flow prediction.

We couldn't gather enough data under extreme weather conditions, which affected the accuracy of our predictions to some extent. However, our experiments show that the online transfer learning approach we proposed still works well. Despite this, ablation studies indicate that the proposed online transfer learning approach remains effective. If a

diverse set of data samples is available in sufficient quantity, it has the potential to improve prediction accuracy and reduce algorithm convergence time. We perform experiments only on our intelligent device and utilize online transfer learning in real-world scenarios. Pre-training on edge servers with rich computing resources is a wise choice that is well combined with IoT technology. In the future, it will be a challenge to consider multiple types of data distributions simultaneously to achieve better prediction accuracy.

References

- Asim M, Wang Y, Wang K, et al., 2020. A review on computational intelligence techniques in cloud and edge computing. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(6):742-763.
- Bai L, Yao L, Li C, et al., 2020. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems*, 33:17804-17815.
- Bando M, Hasebe K, Nakayama A, et al., 1995. Dynamical model of traffic congestion and numerical simulation. *Physical review E*, 51(2):1035.
- Bojan TM, Kumar UR, Bojan VM, 2014. An internet of things based intelligent transportation system. 2014 IEEE international conference on vehicular electronics and safety, p.174-179.
- Bruna J, Zaremba W, Szlam A, et al., 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, .
- Chen C, Li K, Teo SG, et al., 2020. Citywide traffic flow prediction based on multiple gated spatio-temporal convolutional neural networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(4):1-23.
- Choi YY, Lee M, Park SW, et al., 2024. A gated mlp architecture for learning topological dependencies in spatio-temporal graphs. *arXiv preprint arXiv:2401.15894*, .
- Cong P, Zhou J, Li L, et al., 2020. A survey of hierarchical energy optimization for mobile edge computing: A perspective from end devices to the cloud. *ACM Computing Surveys (CSUR)*, 53(2):1-44.
- Crammer K, Dekel O, Keshet J, et al., 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7(3).
- Datla S, Sharma S, 2008. Impact of cold and snow on temporal and spatial variations of highway traffic volumes. *Journal of Transport Geography*, 16(5):358-372.
- Defferrard M, Bresson X, Vandergheynst P, 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.
- Derawi M, Dalveren Y, Cheikh FA, 2020. Internet-of-things-based smart transportation systems for safer roads. 2020 IEEE 6th World Forum on Internet of Things (WF-IoT), p.1-4. <https://doi.org/10.1109/WF-IoT48130.2020.9221208>
- Diao Z, Wang X, Zhang D, et al., 2019. Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting. Proceedings of the AAAI conference on artificial intelligence, 33(01):890-897.
- Gentile C, 2000. A new approximate maximal margin classification algorithm. *Advances in Neural Information Processing Systems*, 13.
- Gräber T, Lupberger S, Unterreiner M, et al., 2018. A hybrid approach to side-slip angle estimation with recurrent neural networks and kinematic vehicle models. *IEEE Transactions on Intelligent Vehicles*, 4(1):39-47.
- Guo G, Yuan W, Liu J, et al., 2023. Traffic forecasting via dilated temporal convolution with peak-sensitive loss. *IEEE Intelligent Transportation Systems Magazine*, 15(1).
- Guo S, Lin Y, Wan H, et al., 2022. Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 34(11):5415-5428. <https://doi.org/10.1109/TKDE.2021.3056502>
- Hamed MM, Al-Masaeid HR, Said ZMB, 1995. Short-term prediction of traffic volume in urban arterials. *Journal of Transportation Engineering*, 121(3):249-254.
- Han P, Wang S, Leung KK, 2020. Adaptive gradient sparsification for efficient federated learning: An online learning approach. 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), p.300-310. <https://doi.org/10.1109/ICDCS47774.2020.00026>
- Hashemi H, Abdelghany K, 2015. Real-time traffic network state prediction for proactive traffic management: Simulation experiments and sensitivity analysis. *Transportation Research Record*, 2491(1):22-31.
- Jeong YS, Byon YJ, Castro-Neto MM, et al., 2013. Supervised weighting-online learning algorithm for short-term traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1700-1707.
- Jia Y, Wu J, Du Y, 2016. Traffic speed prediction using deep learning method. 2016 IEEE 19th international conference on intelligent transportation systems (ITSC), p.1217-1222.
- Kashyap AA, Raviraj S, Devarakonda A, et al., 2022. Traffic flow prediction models—a review of deep learning techniques. *Cogent Engineering*, 9(1):2010510.
- Kipf TN, Welling M, 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, .
- Krizhevsky A, Sutskever I, Hinton GE, 2017. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84-90.
- Kwon YD, Chauhan J, Mascolo C, 2021. Fastcarl: Fast incremental classifier and representation learning with efficient budget allocation in audio sensing applications. *arXiv preprint arXiv:2106.07268*, .
- Li Y, Yu R, Shahabi C, et al., 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, .
- Li Y, Long P, 1999. The relaxed online maximum margin algorithm. *Advances in Neural Information Processing Systems*, 12.
- Liu H, Dong Z, Jiang R, et al., 2023. Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting. Proceedings of the 32nd ACM international conference on information and knowledge management, p.4125-4129.

- Liu J, Guan W, 2004. A summary of traffic flow forecasting methods. *Journal of highway and transportation research and development*, 21(3):82-85.
- Lu B, Gan X, Zhang W, et al., 2022. Spatio-temporal graph few-shot learning with cross-city knowledge transfer. Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, p.1162-1172.
- Lv Y, Duan Y, Kang W, et al., 2014. Traffic flow prediction with big data: A deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):865-873.
- Mallick T, Balaprakash P, Rask E, et al., 2021. Transfer learning with graph neural networks for short-term highway traffic forecasting. 2020 25th International Conference on Pattern Recognition (ICPR), p.10367-10374.
- Nellore K, Hancke GP, 2016. A survey on urban traffic management system using wireless sensor networks. *Sensors*, 16(2):157.
- Qadri YA, Nauman A, Zikria YB, et al., 2020. The future of healthcare internet of things: a survey of emerging technologies. *IEEE Communications Surveys & Tutorials*, 22(2):1121-1167.
- Rosenblatt F, 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.
- Rumelhart DE, Hinton GE, Williams RJ, 1986. Learning representations by back-propagating errors. *nature*, 323(6088):533-536.
- Singh R, Sharma R, Akram SV, et al., 2021. Highway 4.0: Digitalization of highways for vulnerable road safety development with intelligent iot sensors and machine learning. *Safety science*, 143:105407.
- Sun J, Peng M, Jiang H, et al., 2022. Hmian: A hierarchical mapping and interactive attention data fusion network for traffic forecasting. *IEEE Internet of Things Journal*, 9(24):25685-25697.
- Sutskever I, Vinyals O, Le QV, 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.
- Tang C, Sun J, Sun Y, et al., 2020. A general traffic flow prediction approach based on spatial-temporal graph attention. *IEEE Access*, 8:153731-153741.
- Tzeng E, Hoffman J, Darrell T, et al., 2015. Simultaneous deep transfer across domains and tasks. Proceedings of the IEEE international conference on computer vision, p.4068-4076.
- Van Lint J, Van Hinsbergen C, 2012. Short-term traffic and travel time prediction models. *Artificial Intelligence Applications to Critical Transportation Issues*, 22(1):22-41.
- Vaswani A, Shazeer N, Parmar N, et al., 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- Wang S, Miao H, Li J, et al., 2022. Spatio-temporal knowledge transfer for urban crowd flow prediction via deep attentive adaptation networks. *IEEE Transactions on Intelligent Transportation Systems*, 23(5):4695-4705. <https://doi.org/10.1109/TITS.2021.3055207>
- Williams BM, Hoel LA, 2003. Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results. *Journal of transportation engineering*, 129(6):664-672.
- Yao Z, Xia S, Li Y, et al., 2023. Transfer learning with spatial-temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, .
- Yu B, Yin H, Zhu Z, 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, .
- Zhang C, Patras P, 2018. Long-term mobile traffic forecasting using deep spatio-temporal neural networks. Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing, p.231-240.
- Zhang H, Lu X, 2020. Vehicle communication network in intelligent transportation system based on internet of things. *Computer Communications*, 160:799-806.
- Zhang T, Guo G, 2020. Graph attention lstm: A spatiotemporal approach for traffic flow forecasting. *IEEE Intelligent Transportation Systems Magazine*, 14(2):190-196.
- Zhao L, Song Y, Zhang C, et al., 2019. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE transactions on intelligent transportation systems*, 21(9):3848-3858.
- Zhao P, Hoi SC, Jin R, 2011. Double updating online learning. *Journal of Machine Learning Research*, 12:1587.
- Zhu J, Wang Q, Tao C, et al., 2021. Ast-gcn: Attribute-augmented spatiotemporal graph convolutional network for traffic forecasting. *IEEE Access*, 9:35973-35983. <https://doi.org/10.1109/ACCESS.2021.3062114>
- Zivot E, Wang J, 2006. Vector autoregressive models for multivariate time series. *Modeling financial time series with S-PLUS®*, :385-429.