ODMixer: Fine-Grained Spatial-Temporal MLP for Metro Origin-Destination Prediction

Yang Liu, *Member, IEEE*, Binglin Chen, Yongsen Zheng, Lechao Cheng, *Member, IEEE*, Guanbin Li, *Member, IEEE*, and Liang Lin, *Fellow, IEEE*

***Abstract*—Metro Origin-Destination (OD) prediction is a crucial yet challenging spatial-temporal prediction task in urban computing, which aims to accurately forecast cross-station ridership for optimizing metro scheduling and enhancing overall transport efficiency. Analyzing fine-grained and comprehensive relations among stations effectively is imperative for metro OD prediction. However, existing metro OD models either mix information from multiple OD pairs from the station's perspective or exclusively focus on a subset of OD pairs. These approaches may overlook fine-grained relations among OD pairs, leading to difficulties in predicting potential anomalous conditions. To address these challenges, we learn traffic evolution from the perspective of all OD pairs and propose a fine-grained spatial-temporal MLP architecture for metro OD prediction, namely ODMixer. Specifically, our ODMixer has double-branch structure and involves the Channel Mixer, the Multi-view Mixer, and the Bidirectional Trend Learner. The Channel Mixer aims to capture short-term temporal relations among OD pairs, the Multi-view Mixer concentrates on capturing spatial relations from both origin and destination perspectives. To model long-term temporal relations, we introduce the Bidirectional Trend Learner. Extensive experiments on two large-scale metro OD prediction datasets HZMOD and SHMO demonstrate the advantages of our ODMixer.**

***Index Terms*—Correlation learning, spatial-temporal learning, origin-destination prediction, metro system.**

I. Introduction

WITH rapid urbanization, cities are experiencing significant population growth, resulting in a steady rise in vehicle numbers [1], [2], [3], [4]. To improve daily commuting efficiency, metros have become a preferred mode of transportation. Cities are actively expanding their metro networks to increase transport capacity, reduce traffic congestion, and enhance residents' quality of life. In major metropolises like Beijing and Shanghai, daily metro ridership approaches 10 million trips, presenting substantial operational challenges. Accurate forecasting of future ridership is thus essential for effective metro scheduling and route planning. Moreover, to ensure operational efficiency and seamless passenger movement, real-time monitoring of passenger flow and prediction of future trends are critical for addressing emergencies. Consequently, metro OD prediction, which estimates passenger flow between all stations in the near future, has emerged as a vital task for metro scheduling and flow management.

Metro OD prediction poses unique challenges compared to general OD prediction [5], [6], [7]. First, the time gap between passenger entry and exit often causes these events to occur in different time intervals. As a result, the OD matrix for the current time interval cannot fully represent passengers still within the system, leading to incompleteness. Second, the high number of metro stations results in a high-dimensional OD matrix, complicating model design. Additionally, uneven passenger flow distribution creates significant disparities between OD pairs, contributing to matrix sparsity. These factors make metro OD matrix contain highly complex spatiotemporal relations, making metro OD prediction exceptionally challenging.

For metro OD prediction, recent years have seen the rise of deep learning models leveraging powerful nonlinear modeling capabilities [6], [7], [8], [9]. These models employ architectures such as RNNs, TCNs, and Transformers to capture temporal relations, while GCNs and Transformers are commonly utilized to learn spatial dependencies. However, most of these approaches adopt a station-based perspective, as illustrated in Fig. 1(a). This method encodes data for each station during each time interval, causing the flow data of different OD pairs with the same origin to be mixed at the input layer. As a result, the extracted features combine information from multiple OD pairs, making it challenging for the model to distinguish between them when learning spatiotemporal relations. This limitation makes the station-based perspective a coarse-grained modeling method. As depicted in Fig. 2(a), while the overall flow trends of two OD pairs with the same origin may appear similar over time, significant differences in their flow values, particularly during peak hours, are evident. Moreover, in some cases, as shown in Fig. 2(b), the flow trends of OD pairs with the same origin can be entirely opposite. Mixing these OD pairs at the input layer overlooks such differences, hindering the model's ability to effectively learn spatiotemporal relations. To address this,

Received 17 July 2024; revised 6 May 2025; accepted 9 June 2025. Date of publication 12 June 2025; date of current version 24 July 2025. This work was supported in part by the National Key R&D Program of China under Grant 2021ZD0111601, in part by the National Natural Science Foundation of Chin under Grant 62436009, Grant 62322608, and Grant 62472139, and in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2025A1515011874 and Grant 2023A1515011530. Recommended for acceptance by K. Zheng. *(Corresponding author: Liang Lin.)*

Yang Liu, Binglin Chen, Yongsen Zheng, Guanbin Li, and Liang Lin are with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510275, China, and also with the Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou 510275, China (e-mail: liuy856@mail.sysu.edu.cn; chenblin9@mail2.sysu.edu.cn; z.yongsensmile@gmail.com; liguanbin@mail.sysu.edu.cn; linliang@ieee.org).

Lechao Cheng is with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230601, China (e-mail: chenglc@hfut.edu.cn).

Our code is available at https://github.com/KLatitude/ODMixer.

Digital Object Identifier 10.1109/TKDE.2025.3579370

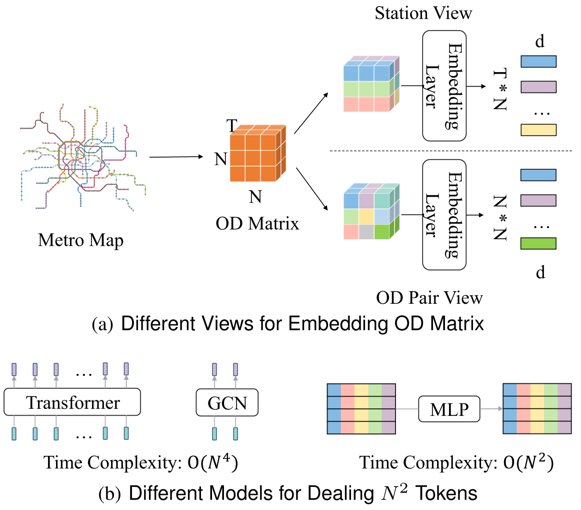


Fig. 1. Comparison of different views and models for metro OD prediction. (a) illustrates the difference in encoding the OD Matrix from the station view and the OD pair view. N is the number of stations, T is the number of time intervals. (b) shows various models for processing the N^2 tokens.

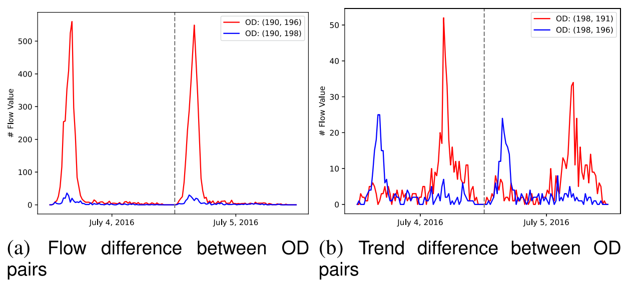


Fig. 2. Comparison of flow changes between OD pairs with the same origin. The x-axis represents July 4 and 5, 2019, the y-axis represents the flow value between OD pairs. The OD: (190,196) in the legend means the origin is station 190 and the destination is station 196. Figure (a) shows that although the OD pairs with the same origin have similar change trends, the flow values are very different. Figure (b) illustrates that the flow rate trends of two OD pairs with the same origin exhibit significant differences over time.

we propose an OD pair-based modeling perspective, as shown in the Fig. 1(a). This approach models each OD pair independently, allowing the model to differentiate between OD pairs and accurately capture their complex spatiotemporal relations. Compared to the station-based perspective, the OD pair-based modeling approach provides a fine-grained representation of the OD matrix, enabling the model to learn more fine-grained and accurate patterns in the data. Additionally, from Fig. 2(a) and (b), we can see that for each OD pair, the OD traffic at adjacent day exhibits a very similar trend. However, existing methods [6] do not consider this similarity and fail to incorporate historical traffic information, leading to sub-optimal model performance.

Actually, there exist some models that predict the metro OD matrix from the OD pair view [10], [11], but they only focus on a subset of OD pairs in the metro. This constrained approach neglects the richness of information among all OD pairs, resulting in the model's inability to learn fine-grained relations across the entirety of OD pairs. Consequently, it hampers the model's capability to address potential emergencies and comprehend the dynamics of traffic changes, making it difficult to provide effective support for metro management. To achieve more comprehensive metro OD prediction, it is imperative to consider information from all OD pairs. However, considering the number of OD pairs as N^2, the time complexity needed to calculate the relations among N^2 OD pairs using Transformer or to construct a virtual graph among OD pairs and subsequently employ Graph Convolution Network (GCN) for information aggregation is $O(N^4)$, as shown in Fig. 1(b). Therefore, balancing the model's parameters and computation time while considering all OD pairs remains challenging.

Inspired by the efficiency of the MLP architecture [12], [13], we propose ODMixer, a simple yet effective fine-grained spatiotemporal MLP architecture. Unlike existing metro OD prediction models that adopt coarse-grained station-based modeling, ODMixer employs a fine-grained OD pair-based modeling approach to comprehensively capture the complex spatial-temporal relations in metro OD prediction. To address the issue of incomplete input data in metro OD prediction, we introduce the OD Matrix Process module, which leverages periodicity to fill missing values in the OD matrix. Once the OD matrix is complete, we independently model each OD pair from an OD pair-based perspective. This approach not only considers each OD pair separately but also mitigates the challenges posed by uneven distribution in the OD matrix. To capture the intricate spatiotemporal relationships within the OD matrix, we design specialized modules. For temporal relations, we account for both short-term and long-term dependencies. The Channel Mixer operates on the most recent features of each OD pair to capture short-term temporal relations. To model long-term trends, such as periodicity inherent in metro traffic, we employ a dual-branch structure coupled with a Bidirectional Trend Learner. Regarding spatial relations, we treat each OD pair as a node. Given the computational challenges posed by models like Transformers and GCNs in processing all OD pairs, we adopt a decomposition strategy. Specifically, we decompose spatial relations based on the origin and destination perspectives and design the Multi-view Mixer module. This module separately calculates spatial relations from the origin and destination perspectives, utilizing a Mixer structure for each. This design enhances the model's efficiency and its ability to learn spatial relationships across all OD pairs.

Our contributions are summarized as follows:

- We propose a fine-grained spatial-temporal MLP architecture named ODMixer from the perspective of the OD pairs, to comprehensively capture OD relations and achieve accurate and efficient metro OD prediction.
- To effectively learn the spatial and temporal dependencies between metro flows, we propose two specific modules, the Channel Mixer and the Multi-view Mixer.
- To empower the model with the capability to perceive long-term flow changes, the Bidirectional Trend Learner (BTL) is introduced in the ODMixer.
- Extensive experiments on two large-scale metro OD datasets demonstrate our promising OD prediction accuracy. Our ODMixer outperforms the state-of-the-art models in wMAPE for HZMOD and SHMOD datasets by a considerable margin of 5% and 7%, respectively.

II. Related Work

A. Traffic Flow Prediction

Traffic flow prediction is a very important task in smart city systems. Its main task is to use historical traffic information to accurately predict future traffic flow. There are many methods to solve this problem. Initially, researchers used traditional statistical methods to analyze traffic changes, such as ARIMA [14]. Later, some machine learning-based methods began to emerge, such as SVM [15] and decision trees [16].

With the rapid development of deep learning, neural networks have been used to solve traffic prediction problems. These representative works include many structural types, mainly including CNN [17], [18], RNN [19], [20], Attention [21], [22], etc. DeepSD [23] proposed an end-to-end deep learning framework and integrates external environment information to realize the automatic discovery of complex supply-demand patterns from the car-hailing service data. Based on three temporal relations, MDL [24] used two three-stream fully convolutional networks (3S-FCNs) to achieve flow prediction for points and edges. DeepCrowd [25] divided the area into multiple fine-grained grids, and combines ConvLSTM and attention mechanisms to analyze and fuse flow information at multiple times to achieve Large-Scale Citywide Crowd Density and Flow Prediction. Later, the GCN was popular to process the relation between nodes because of the natural network structure in the traffic flow prediction problem [26], [27], [28]. STDGRL [29] designed a spatio-temporal dynamic graph relation learning model. STWave+[30] utilized decoupling technology to partition one-hour traffic data into stable trends and fluctuating events. This approach employs a dual-channel spatiotemporal network to separately model these components, leveraging self-supervised learning and contrastive loss to propagate long-term trend information to hourly trends.

However, these methods focus solely on the total flow at each node, they fail to analyze the flow relations between nodes. This limitation restricts the information that can be obtained, making it less suitable for practical applications such as traffic analysis and control. To address this issue, we concentrate on the relations between nodes. Compared to focusing only on the total flow at each node, this OD relation provides a more fine-grained analysis. Additionally, the total flow at a node can be derived from the OD flow, highlighting the greater value of OD flow data.

B. Origin-Destination Prediction

OD prediction [31], [32], [33], [34] involves the accurate estimation of traffic flow between two regions over a given period of time, presenting a formidable challenge due to its intricate and dynamic nature. As a pivotal aspect of urban traffic management, OD prediction has garnered extensive attention.

As a special type of OD prediction, the metro OD prediction [5], [6], [7], [8], [10], [35], [36], [37] presents unique challenges compared to the general OD prediction, primarily because the precise destinations of passengers are not accurately known until they reach their destinations. This leads to incomplete data at the current moment, making it challenging to effectively depict the real-time traffic distribution. To address this issue and obtain more comprehensive data on traffic distribution, several studies focused on predicting unfinished orders to supplement the incomplete OD matrix. For instance, in HIAM [6], long and short historical distributions were utilized to estimate the distribution of unfinished orders at the current time. This study integrated the completed and unfinished distributions in the feature space. Gong et al. [7] created an indication matrix, with values assigned based on the assumption that the travel time of each OD pair at each timestamp follows a normal distribution. However, this approach overlooked the unfinished OD pair data. MVPF [5] proposed a multi-view passenger flow evolution trend based OD matrix prediction method that combines individual station and cross-station learning, uses GRU and EGAT models to learn the spatial-temporal-dependent representation of stations, and defines a transfer matrix to capture passenger mobility patterns.

Different from previous approaches that neglect the richness of information among all OD pairs, we consider all OD pairs from the OD pair view and propose a novel model base on the spatial-temporal MLP architecture to achieve more fine-grained and comprehensive relation modeling.

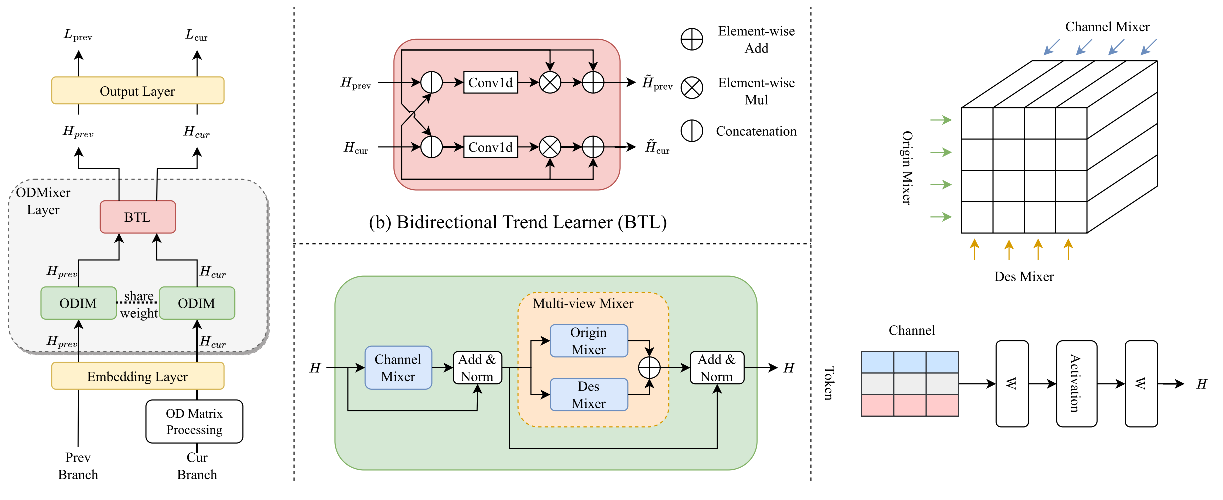
C. MLP-Based Model

With the popularity of the Transformer, its exceptional performance has led to its widespread adoption across various fields. However, several MLP-based models have attracted attention for addressing diverse problems recently. In computer vision, MLP-Mixer [12] addressed vision challenges through an entirely MLP-based implementation. Another notable model gMLP [38] outperformed some Transformer-based models by leveraging channel projection, spatial projection, and gating mechanisms. ResMLP [39] introduced a residual structure atop MLP, incorporating training methods like self-supervision for efficient training. sMLPNet [40] focused on reducing computational costs while maintaining performance. For sequential recommendation problems, MLP4Rec [41] employed a tri-directional mixing MLP model, while MLPST [42] introduced SpatialMixer and TemporalMixer to address spatio-temporal dependencies and temporal variations in traffic prediction. For time series forecasting task, Ekambaram et al. [43] and Chen et al. [44] proposed MLP-based models dedicated to efficient time series forecasting.

Unlike existing MLP-Mixer models that neglect fine-grained spatial-temporal relations, we focus on metro OD prediction task that requires fine-grained and comprehensive OD relations. In this paper, we propose a fine-grained spatial-temporal MLP model ODMixer, which excels in capturing spatial relations of metro flows, along with short-term and long-term temporal dependencies.

III. Preliminaries

In this section, a concise introduction is provided for the notation and problem definitions. Table I presents a comprehensive list of commonly used symbols for reference.



(a) ODMixer (c) Origin-Destination Interact Module (ODIM) (d) Channel Mixer, Origin Mixer, Des Mixer

(b) Bidirectional Trend Learner (BTL)

Fig. 3. The overall framework of the ODMixer and its essential modules. (a) illustrates the architecture of double-branch ODMixer, both branches learn features using ODIM. Subsequently, the two branches interact with each other using the BTL, contributing to the final output. (b) depicts the BTL module, which aims to jointly model the features from both branches, thereby enhancing the information exchange. (c) represents the ODIM module, which models the temporal attributes of the OD pair features using the Channel Mixer, while the comprehensive relations among OD pairs are learned using the Multi-view Mixer. (d) shows the Channel Mixer, Origin Mixer, Des Mixer.

TABLE I
NOTATION TABLE

Notations	Description
t	the current time interval
d	the current day
T	the number of time intervals of input
N	the number of metro stations
$\text{unf}_{t-T+i}^{d} \in \mathbb{R}^{N}$	the unfinished order vector ($i = 1, \cdots, T$)
$\text{OD}_{t-T+i}^{d} \in \mathbb{R}^{N\times N}$	the complete OD matrix ($i = 1, \cdots, T$)
$\text{UOD}_{t-T+i}^{d} \in \mathbb{R}^{N\times N}$	the unfinished OD matrix ($i = 1, \cdots, T$)
$\text{IOD}_{t-T+i}^{d} \in \mathbb{R}^{N\times N}$	the incomplete OD matrix ($i = 1, \cdots, T$)

Definition 1: Incomplete OD Matrix $\text{IOD}_t^d \in \mathbb{R}^{N\times N}$: $\text{IOD}_t^d(i,j)$ is the number of passengers entering from station i and exiting from station j at the tth time interval of the dth day.

Definition 2: Unfinished OD Matrix $\text{UOD}_t^d \in \mathbb{R}^{N\times N}$: $\text{UOD}_t^d(i,j)$ signifies the number of passengers entering station i at the tth time interval and exiting from station j after the tth time interval of the dth day.

Definition 3: Complete OD Matrix $\text{OD}_t^d \in \mathbb{R}^{N\times N}$: $\text{OD}_t^d(i,j)$ denotes the number of passengers at the tth time interval entering station i and subsequently exiting from station j at and after the tth time interval of the dth day. $\text{OD}_t^d = \text{IOD}_t^d + \text{UOD}_t^d$.

Definition 4: Unfinished Order Vector $\text{unf}_t^d \in \mathbb{R}^N$: $\text{unf}_t^d(i)$ represents the number of passengers at the tth time interval entering station i but not exiting from any station at the tth time interval of the dth day. $\text{unf}_t^d(i) = \sum_{j=1}^{N} \text{UOD}_t^d(i,j)$.

Problem 1: Considering the historical complete OD matrix (OD), alongside the incomplete OD matrix (IOD) and the unfinished orders (unf) from the most recent time period, we determine a function f to predict the complete OD flow at the subsequent time step.

$$\text{OD}_{t+1}^{d} = f\left(\left\{\text{OD}^{d-1}, \text{IOD}^{d}, \text{unf}^{d}\right\}_{t-T+i}\right) \qquad (1)$$

where $i = 1, 2, \ldots T$, OD_{t+1}^d represents the OD matrix at the $(t+1)$th time interval of the dth day.

IV. METHODOLOGY

This section gives a detailed description of our ODMixer.

A. Architecture Overview

Our proposed ODMixer comprises two branches that operate similarly and share parameters, as shown in Fig. 3. Initially, in the Embedding Layer, we embed the input based on the OD pair view. Subsequently, we employ ODIM to learn the relations between features. Specifically, we utilize Channel Mixer to learn the short-term temporal relations of features, followed by employing Multi-view Mixer to comprehensively learn the relations among OD pairs from both Origin and Destination perspectives. Subsequently, we use BTL to interact the features of the two branches, enhancing the long-term temporal attributes of the features. This augmentation ensures that the features contain more information, ultimately contributing to increase prediction accuracy. Finally, the Output Layer is employed to output the future OD matrics.

B. Dual-Branch Structure

As shown in Fig. 2, the OD flow on adjacent days exhibits certain similarities. Therefore, we integrate this similarity into our model by designing a dual-branch structure. The Prev Branch learns historical information, while the Cur Branch learns information at the current time. Since we focus on the similarity between adjacent days, the input to the Prev Branch is the OD matrix from the same time on the previous day. Since yesterday's order information has been completed, its input is a complete OD matrix and no operation is required. For the Cur branch, the OD matrix is incomplete as some orders have not yet been completed.

If we directly use the incomplete OD matrix, discrepancies between the inputs of the Prev Branch and Cur Branch arise, hindering the model's ability to effectively perceive complete information for the current time. Therefore, we introduce the OD Matrix Processing module in the Cur branch.

Specifically, we use an Embedding layer to encode the OD matrix based on the OD pair perspective for the inputs of both branches. Subsequently, the ODIM module learns the relations between OD pairs. We then introduce the BTL module to learn the relations between the two branches, supplementing the Cur Branch with historical information. This design captures the similarities between adjacent days and completes the incomplete information for the current time, improving the model's prediction accuracy and reliability.

This dual-branch structure not only captures the complex relation between OD traffic and temporal dynamics but also enhances the prediction accuracy of current OD traffic by leveraging historical information. Consequently, our model can better handle the large and complex OD matrix in the subway system, utilizing historical data for accurate prediction, thereby demonstrating significant potential and practical value in real-world applications.

C. OD Matrix Processing

In this subsection, we introduce the approach for supplementing the OD matrix and estimating the current UOD distribution by leveraging the historical UOD distribution.

1) Unfinished Orders Processing: As the destination information is only available until passengers have exited the station, the OD matrix for the current time period is incomplete. To address this issue and enhance the completeness of the input OD matrix for the model, preprocessing steps are implemented before integrating the matrix into the model.

Acknowledging the temporal similarity characteristic of metro traffic, where the current OD traffic distribution closely resembles that of the previous time, we estimate the current UOD distribution by referencing historical UOD distributions. Our approach specifically incorporates two temporal scales: long-term and short-term UOD distributions. The calculation is outlined as follows:

$$\mathrm{US}^d_{t-T+i}(j,k) = \mathrm{unf}^d_{t-T+i}(j) * \frac{\mathrm{UOD}^{d-1}_{t-T+i}(j,k)}{\sum_{k=1}^N \mathrm{UOD}^{d-1}_{t-T+i}(j,k)}$$

$$\mathrm{UL}^d_{t-T+i}(j,k) = \mathrm{unf}^d_{t-T+i}(j) * \frac{\mathrm{UOD}^{d-7}_{t-T+i}(j,k)}{\sum_{k=1}^N \mathrm{UOD}^{d-7}_{t-T+i}(j,k)}$$

$$\widetilde{\mathrm{UOD}}^d_{t-T+i}(j,k) = \frac{1}{2}\left(\mathrm{US}^d_{t-T+i}(j,k) + \mathrm{UL}^d_{t-T+i}(j,k)\right) \quad (2)$$

Here, $i = 1,\ldots,T$, $\widetilde{\mathrm{UOD}}^d_{t-T+i}$ represents an estimation of unfinished OD matrix at the $(t-T+i)$th time interval of the dth day, the symbol $*$ denotes the multiplication of the corresponding elements. $\mathrm{UOD}^{d-1}_{t-T+i}$ represents the short-term UOD distribution, corresponding to the UOD distribution at the same time yesterday, and $\mathrm{UOD}^{d-7}_{t-T+i}$ represents the long-term UOD distribution, aligned with the UOD distribution observed at the corresponding time on the same day of the previous week. This computation aims to generate a more comprehensive and accurate estimate of the UOD distribution at the current time by combining both short-term and long-term UOD distributions.

Subsequently, we derive the OD matrix for the current time, which serves as input to the Embedding Layer:

$$\widetilde{\mathrm{OD}}^d_{t-T+i}(j,k) = \mathrm{IOD}^d_{t-T+i}(j,k) + \widetilde{\mathrm{UOD}}^d_{t-T+i}(j,k) \quad (3)$$

where $i = 1,\ldots,T$, $\widetilde{\mathrm{OD}}^d_{t-T+i}$ is the estimation of complete OD matrix at the current time, IOD^d_{t-T+i} represents the matrix of actual completed orders, and $\widetilde{\mathrm{UOD}}^d_{t-T+i}$ denotes the matrix of unfinished orders computed from the historical short-term and long-term UOD distributions. This approach ensures the model receives comprehensive and accurate input information by integrating data from both completed and ongoing orders, thereby enhancing the accuracy of passenger destination predictions.

D. Embedding Layer

Unlike the previous models for handling OD matrix, we consider it from the OD pair view. We encode each OD pair sequence using the Embedding Layer, which ultimately yields fine-grained features of the OD matrix:

$$\boldsymbol{H}(i,j) = \boldsymbol{W}^{\boldsymbol{E}}\mathrm{OD}_{t-T+1:t}(i,j) \quad (4)$$

where $i = 1,\ldots,N, j = 1,\ldots,N$, $\boldsymbol{W}^{\boldsymbol{E}} \in \mathbb{R}^{d\times T}$ is learnable weights of Embedding layer, $\boldsymbol{H} \in \mathbb{R}^{N\times N\times d}$ represents the obtained feature representation after the Embedding Layer. This approach differs from the previous methods that treat the OD matrix as a whole. Instead, we emphasize modeling each OD pair independently, to effectively capture the characteristics and dynamics of each OD pair. After performing embedding layer on both Prev and Cur branches, we obtain the feature $\boldsymbol{H}_{\text{prev}}, \boldsymbol{H}_{\text{cur}} \in \mathbb{R}^{N\times N\times d}$.

By modeling OD sequences, we can distinguish the difference among OD pairs at a fine-grained manner, which enhances the sensitivity and accuracy of the model in predicting OD matrix. Moreover, this approach is more realistic, considering significant differences among various OD pairs. Modeling these OD pairs individually better reflects the inherent variability within the data.

E. Origin-Destination Interact Module (ODIM)

For the metro OD prediction task, the spatio-temporal relations plays a crucial role. First, passenger flow information in adjacent time periods is correlated, indicating that current passenger flow can be influenced by flows in preceding and succeeding periods. Second, each metro station has unique attributes, and stations with similar attributes typically exhibit similar passenger flow trends. Therefore, by learning and capturing the spatial relations of OD pairs, the model's prediction performance can be significantly enhanced, leading to more accurate predictions of future passenger flow distribution. To address this, we design the ODIM module to effectively capture the spatio-temporal relations of OD pairs. ODIM conducts spatio-temporal interaction of features encoded by the Embedding Layer. It comprises two modules: Channel Mixer, which considers the short-term temporal relations of features within OD pairs, and Multi-view Mixer, which models the relations among OD pairs.

1) Channel Mixer: In the domain of metro traffic, temporal attributes play a pivotal role, and there exist discernible correlations between traffic patterns at different time. Consequently, the model must adeptly capture the temporal characteristics of flow to ensure accurate OD predictions. Leveraging the Embedding Layer, we encode each OD sequence as features. To effectively capturing the time-varying relations within the flow, it is imperative to compute the inter-dependencies among these features. Thus, we introduce the Channel Mixer structure, facilitating the fusion of attributes across different time to discern and learn the temporal patterns inherent in the flow. This yields a feature representation enriched with temporal attributes.

The Channel Mixer structure can be expressed as follows:

$$\boldsymbol{H}_{\boldsymbol{i}}^{\boldsymbol{C}} = LayerNorm\left(\boldsymbol{H}_{\boldsymbol{i}} + \boldsymbol{W}^{\boldsymbol{C}_2}\sigma\left(\boldsymbol{W}^{\boldsymbol{C}_1}\boldsymbol{H}_{\boldsymbol{i}}\right)\right) \quad (5)$$

where, $i = 1, \ldots, N^2$, σ is the activation function, $\boldsymbol{W}^{\boldsymbol{C}_1} \in \mathbb{R}^{d_C \times d}$ is learnable weights of the first fully connected layer in the Channel Mixer, $\boldsymbol{W}^{\boldsymbol{C}_2} \in \mathbb{R}^{d \times d_C}$ is learnable weights of the second fully connected layer, $\boldsymbol{H}^{\boldsymbol{C}} \in \mathbb{R}^{N \times N \times d}$ represents the feature after temporal attributes have enhanced by the Channel Mixer. We employ layer normalization (LayerNorm) [45] and residual connection [46] in the Channel Mixer. The Channel Mixer structure is specifically designed to merge attributes from diverse time intervals within the feature representation, enhancing the model's ability to capture temporal traffic patterns. Most importantly, the Channel Mixer parameters are shared across all OD pairs, implicitly fostering temporal interactions between distinct OD pairs without necessitating an increase in parameter count. This shared parameter design contributes to the model's efficiency in learning global temporal patterns, thereby enhancing its capacity to comprehend and predict variations in traffic over time.

2) Multi-View Mixer: The diverse relations among different OD pairs, such as the similarity in traffic variations among OD pairs connecting similar types of stations, hold the potential to enhance the accuracy of OD predictions. However, calculating the relations among N^2 OD pairs using attention models, such as Transformer, demands a considerable amount of time. Moreover, the direct calculation of relations among N^2 OD pairs may pose challenges in discerning the relations crucial for actual prediction. Therefore, efficiently computing the relations among OD pairs is a crucial and challenging problem.

To address this issue, we introduce the Multi-view Mixer module by analyzing relations among OD pairs from multiple perspectives. Recognizing that each OD pair represents the flow between two stations, which is correlated with the origin and destination, we propose the Multi-view Mixer module, consisting of two distinct parts: Origin Mixer and Des Mixer.

Origin Mixer Module: From the perspective of the origin within an OD pair, this module evaluates all OD pairs sharing the same origin. Concretely, we apply a dimensional transformation to the input feature $\boldsymbol{H}^{\boldsymbol{C}}$ to obtain the transformed feature $\boldsymbol{H}^{\boldsymbol{C_O}} \in \mathbb{R}^{(N \times d) \times N}$, where the first N is the number of origins, the second N is the number of destinations. Then we calculate the relations among OD pairs as indicated by:

$$\boldsymbol{H}_{\boldsymbol{i}}^{\boldsymbol{O}} = \boldsymbol{W}^{\boldsymbol{O}_2}\sigma\left(\boldsymbol{W}^{\boldsymbol{O}_1}\boldsymbol{H}_{\boldsymbol{i}}^{\boldsymbol{C_O}}\right) \quad (6)$$

where $i = 1, \ldots, N \times d$, $\boldsymbol{W}^{\boldsymbol{O}_1} \in \mathbb{R}^{d_O \times N}$ and $\boldsymbol{W}^{\boldsymbol{O}_2} \in \mathbb{R}^{N \times d_O}$ are the learnable weights of the Origin Mixer, $\boldsymbol{H}_{\boldsymbol{O}} \in \mathbb{R}^{N \times N \times d}$ denotes the output of the module.

Des Mixer Module: Analogous to the Origin Mixer, this module concentrates on relations among OD pairs from the destination perspective. To be more specific, we initially interchange the two dimensions of origin and destination, and subsequently transform the dimensions in the same way as the Origin Mixer to derive $\boldsymbol{H}^{\boldsymbol{C_D}} \in \mathbb{R}^{(N \times d) \times N}$, where the first N is the number of destinations and the second N is the number of origins. The calculation is:

$$\boldsymbol{H}_{\boldsymbol{i}}^{\boldsymbol{D}} = \boldsymbol{W}^{\boldsymbol{D}_2}\sigma\left(\boldsymbol{W}^{\boldsymbol{D}_1}\boldsymbol{H}_{\boldsymbol{i}}^{\boldsymbol{C_D}}\right) \quad (7)$$

Where, $i = 1, \ldots, N \times d$, $\boldsymbol{W}^{\boldsymbol{D}_1} \in \mathbb{R}^{d_D \times N}$ and $\boldsymbol{W}^{\boldsymbol{D}_2} \in \mathbb{R}^{N \times d_D}$ are the learnable weights of the Des Mixer, $\boldsymbol{H}^{\boldsymbol{D}} \in \mathbb{R}^{N \times N \times d}$ is the output of the Des Mixer.

Then, to obtain features after comprehensive interaction with the remaining OD pairs, a fusion approach is employed to maintain simplicity while reducing computation time:

$$\boldsymbol{H}^{\boldsymbol{F}} = LayerNorm\left(\boldsymbol{H}^{\boldsymbol{O}} + \boldsymbol{H}^{\boldsymbol{D}} + \boldsymbol{H}^{\boldsymbol{C}}\right) \quad (8)$$

where $\boldsymbol{H}^{\boldsymbol{F}} \in \mathbb{R}^{N \times N \times d}$.

This design enables more efficient modeling of relations among complex OD pairs, thereby enhancing the model's accuracy in OD prediction. Simultaneously, by considering relations from different perspectives, the model comprehensively captures the intricate dynamics among stations.

F. Bidirectional Trend Learner (BTL)

Due to the regularity observed in people's daily routines, metro traffic exhibits temporal similarities. To effectively leverage this temporal consistency and provide a reference for the trend in OD traffic changes at the current time, we introduce a BTL. This learner discerns the temporal similarities and change trends in the flow in two directions: from the past to the present and from the present to the past, respectively.

The bidirectional trend learner comprises two symmetric components, one of which is described below. Initially, we aggregate two temporal features and subsequently interact these features using 1D convolution to obtain the feature $\boldsymbol{H}_{\boldsymbol{f}}$, representing the perceptual flow features over time:

$$\boldsymbol{H}_{\boldsymbol{f}} = \text{Conv1d}\left([\boldsymbol{H}_{\text{prev}}; \boldsymbol{H}_{\text{cur}}]\right) \quad (9)$$

$[;]$ denotes the concatenation of features, $\boldsymbol{H}_{\text{prev}}, \boldsymbol{H}_{\text{cur}} \in \mathbb{R}^{N\times N\times d}$ represent the features of two branches, respectively, $\boldsymbol{H_f} \in \mathbb{R}^{N\times N\times d}$ is the feature after the interaction.

Subsequently, the impact of the perceived change on $\boldsymbol{H}_{\text{prev}}$, denoted as $\boldsymbol{H_g} \in \mathbb{R}^{N\times N\times d}$, is obtained through the Sigmoid activation function:

$$\boldsymbol{H_g} = \delta\left(\boldsymbol{W^g H_f}\right) \tag{10}$$

where $\boldsymbol{W^g} \in \mathbb{R}^{d\times d}$ is the learnable weights, δ denotes the Sigmoid activation function.

Finally, we apply this effect to $\boldsymbol{H}_{\text{prev}}$ to obtain $\tilde{\boldsymbol{H}}_{\text{prev}} \in \mathbb{R}^{N\times N\times d}$, capturing the trend of the flow from the present to the past:

$$\tilde{\boldsymbol{H}}_{\text{prev}} = \boldsymbol{H_g} * (\boldsymbol{W^p H}_{\text{prev}}) + \boldsymbol{H}_{\text{prev}} \tag{11}$$

where $\boldsymbol{W^p} \in \mathbb{R}^{d\times d}$, $*$ is the Hadamard product.

With the bidirectional trend learner, we derive features that perceive the traffic trend in both directions over time:

$$\tilde{\boldsymbol{H}}_{\text{prev}}, \tilde{\boldsymbol{H}}_{\text{cur}} = \text{BTL}\left(\boldsymbol{H}_{\text{prev}}, \boldsymbol{H}_{\text{cur}}\right) \tag{12}$$

This design empowers the model to comprehensively grasp temporal similarities and trends in both the past-to-present and present-to-past directions, thereby enhancing the accuracy of OD flow predictions at the current time.

G. Optimization

In this subsection, we first introduce the double-branch architecture of the model, followed by a detailed explanation of the model's loss function. Subsequently, we perform time complexity analysis for each component and the overall model.

1) Loss Function: To fully exploit the temporal relations within OD traffic, we establish two branches to model the past and present variations in OD traffic. Here, 'Prev' represents the OD traffic at the same time yesterday, and 'Cur' represents the OD traffic at the current time. Notably, the layer parameters of these two branches are shared. We formulate two loss functions to simultaneously optimize both branches, thereby further enriching the temporal attributes embedded in the learned features.

Ultimately, the loss function of the ODMixer is defined as:

$$\begin{aligned}
\mathcal{L} &= \mathcal{L}_{prev} + \mathcal{L}_{cur} \\
\mathcal{L}_{prev} &= \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} |\widetilde{OD}_{t+1}^{d-1}(i,j) - OD_{t+1}^{d-1}(i,j)|}{N^2} \\
\mathcal{L}_{cur} &= \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} |\widetilde{OD}_{t+1}^{d}(i,j) - OD_{t+1}^{d}(i,j)|}{N^2}
\end{aligned} \tag{13}$$

where $\widetilde{OD}_{t+1}^{d-1}, \widetilde{OD}_{t+1}^{d}$ and $OD_{t+1}^{d-1}, OD_{t+1}^{d}$ represent the predicted and true values of the OD matrix at the $(t+1)$th time interval of $(d-1)$th or dth day, respectively.

This loss function is designed to prompt the model to learn trends in both yesterday's and today's OD traffic, facilitating a more robust capture of temporal relations. Moreover, this approach aims to enhance the accuracy of OD forecasts by better understanding and predicting the evolving patterns of OD traffic over time.

TABLE II
DETAILS OF THE DATASETS

Dataset	HZMOD	SHMOD
Region	Hangzhou	Shanghai
Stations	80	288
OD Pairs	80*80	288*288
Time Interval	15 minutes	15 minutes
Training Set	1/1/2019 - 1/18/2019	7/1/2016 - 8/31/2016
Validation Set	1/19/2019 - 1/20/2019	9/1/2016 - 9/9/2016
Testing Set	1/21/2019 - 1/25/2019	9/10/2016 - 9/30/2016

2) Model Complexity: The ODMixer model, comprising two fundamental components, namely ODIM and BTL, is analyzed for its computational complexities.

ODIM consists of two components: Channel Mixer and Multi-view Mixer. For Channel Mixer, the computational cost primarily arises from matrix multiplication in the linear layer, with intermediate dimensions being $2d$. This results in approximately $4N^2d^2$ operations. Since d is chosen independent of N, the time complexity of the Channel Mixer is $O(N^2)$.

The Multi-view Mixer module comprises two components: the Origin Mixer and the Des Mixer, with both components sharing a similar structure. Taking Origin Mixer as an example, the number of operations in the module is around $4N^2d^2$, resulting in a time complexity of $O(N^2)$.

For the BTL module, it contains two symmetric parts. Focusing on the top half, it comprises a 1D convolution layer followed by two linear layers. The computation required for the 1D convolution is $2N^2d$ operations, and each linear layer requires N^2d^2 operations. Consequently, the time complexity of the BTL module is $O(N^2)$.

Combining these analyses, the ODMixer model demonstrates a time complexity of $O(N^2)$. The training and inference time of both the baseline and ODMixer models are evaluated through experiments, as showed in Tables VII and VIII.

V. Experiment

In this section, we validate the effectiveness of our model across two extensive datasets. First, we describe the experimental configuration, including construction of the datasets, implementation details and evaluation metrics. Subsequently, we compare our proposed ODMixer with both basic and state-of-the-art models. We then conduct ablation studies to verify the influence of each component on the model's performance. Additionally, we also analyze the hyper-parameter settings and the efficiency of the model. Finally, we evaluate ODMixer's scalability, robustness and performance in cross-city experiments to demonstrate its capabilities.

A. Experimental Configuration

1) Dataset Description: We conduct experiments on two real-world datasets [6], as shown in Table II. HZMOD records transaction data for 80 stations from January 1, 2019, to January 25, 2019. SHMOD records transaction data for 288 stations from July 1, 2016, to September 30, 2016. These transaction data are segmented into 15-minute interval, and the complete dataset is partitioned into training, validation, and test set following a specific ratio. Compared with previous subway datasets, these two datasets contain billions of transaction records from the Shanghai and Hangzhou subway systems in China, fully reflecting various real-world scenarios. The SHMOD dataset, in particular, includes 288 stations, resulting in 288$*$288 OD pairs, a substantial number. This scale introduces significant differences in the OD matrix, making it an excellent benchmark for assessing the model's ability to handle large and complex OD matrices.

2) Implementation Details: We implement our model in PyTorch and conduct both training and testing using 8 RTX 2080Ti GPUs. The initial learning rate for the model is set at 0.001. To optimize the model loss, we employ the Adam optimizer. The batch size is set to 32 for both datasets. The input sequence T is set to 4. The input data and the ground-truth of output are normalized with Z-score normalization before fed into the model. The feature dimension d is 16 and ODMixer layer L is 5. And we set $d_C = d_O = d_D = 2 \times d$.

3) Quantitative Evaluation Metrics: We have selected three widely employed performance metrics within the domain of traffic flow prediction for evaluating the model's effectiveness. These metrics include Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Weighted Mean Absolute Percentage Error (wMAPE). MAE is simple and intuitive, insensitive to outliers, suitable for scenarios that require robustness. RMSE penalizes large errors, provides information about the distribution of prediction errors, and is suitable for scenarios that need to emphasize the severity of errors. wMAPE expresses errors in percentage form, suitable for comparison of datasets of different sizes, and can be weighted to reflect the importance of data points.

$$\text{MAE} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}|\widetilde{OD}(i,j) - OD(i,j)|$$

$$\text{RMSE} = \sqrt{\frac{1}{N^2}\sum\nolimits_{i=1}^{N}\sum\nolimits_{j=1}^{N}(\widetilde{OD}(i,j) - OD(i,j))^2}$$

$$\text{wMAPE} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}|\widetilde{OD}(i,j) - OD(i,j)|}{\sum_{i=1}^{N}\sum_{j=1}^{N}OD(i,j)} \tag{14}$$

B. Comparison With State-of-the-Art Methods

We conduct a comparative analysis between our proposed model and a selection of classical as well as contemporary models. STCNN [10] and ST-VGCN [11] that consider OD pairs aren't included because they have no publicly accessible codes and have not been evaluated on identical datasets.

- *Historical Average (HA):* Future projections are derived by directly utilizing historical flow averages for each metro station.
- *LSTM [47]:* This model utilizes a Seq2Seq approach, employing two LSTM layers to predict future metro OD.
- *GRU [48]:* The architecture of this model closely resembles the previous one, with a key distinction being the use of GRU layers instead of LSTM layers.
- *Graph WaveNet [49]:* The model acquires adaptive dependency matrices within the data by embedding nodes, and it captures long-range temporal dependencies through the use of stacked dilated 1D convolutions.
- *DCRNN [50]:* DCRNN captures spatial dependencies through the utilization of bi-directional random walks on the graph. Additionally, it employs encoder-decoder structures with scheduled sampling to model temporal information.
- *STG2Seq [51]:* STG2Seq incorporates a hierarchical graph convolution structure to effectively capture both spatial and temporal correlations.
- *PVCGN [52]:* PVCGN comprehensively learns complex spatio-temporal relations between metro stations by constructing physical, similarity, and correlation graphs.
- *DGSL [53]:* DGSL optimizes the average performance of the graph distribution by leveraging probabilistic graph models when the graph structure is unknown, thus facilitating the learning of the graph structure.
- *Informer [54]:* Informer is a transformer-based model specifically designed for the task of long sequence time-series forecasting. It is notable for its efficient ProbSparse self-attention mechanism, self-attention distillation, and generative-style decoder. For our metro OD prediction, we build our model using the official code of Informer as a foundation, and we customize it to suit our specific requirements.
- *HIAM [6]:* HIAM integrates diverse information to jointly learn the evolutionary patterns of OD and OD, featuring two branch for OD and DO estimation separately and a DIT for enhanced OD-DO correlation modeling.
- *HSTN [55]:* HSTN designs a HSM to capture three types of spatial relations and a HTM to quantify the influence of the input sequence on the target result.
- *MVPF [5]:* MVPF considers multiple views of real-time traffic information including individual station and cross-station flow information.
- *C-AHGCSP [56]:* C-AHGCSP completes unfinished trips based on real-time mobility evolution and inter-station time costs to enhance dynamic adaptability and considers the destination distributions of the passengers departing from a station are correlated with other stations sharing similar attributes.

We have conducted experiments on two datasets, and the experimental results for all methods are presented in Tables III and IV. We use the same data inputs as specified in the original papers when evaluating all baselines, provided the relevant data is available. For example, HIAM utilizes additional information, such as the Destination-Origin (DO) matrix, which we incorporate in our experiments. First, the simplest method, HA, can partially reflect the flow range of the training set by directly averaging it. However, lacking generalization ability, it fails to capture temporal variations in traffic. Consequently, its performance tends to be inferior to that of ODMixer, with the performance gap widening as the dataset size increases. After analysis, it is evident that the performance of LSTM, GRU, Graph WaveNet, STG2Seq, HSTN and MVPF models

TABLE III
PERFORMANCE OF OD PREDICTION ON HZMOD DATASET

Models	Spatial Learning	Temporal Learning	MAE (↓)	RMSE (↓)	wMAPE (↓)
HA	-	-	1.355	2.917	48.354%
LSTM	-	LSTM	1.387	3.458	49.500%
GRU	-	GRU	1.427	3.593	50.906%
Graph WaveNet	GCN	TCN	1.717	4.431	62.489%
HSTN	GCN+ Attention	GRU+ Attention	1.692	5.558	60.392%
MVPF	GAT	GRU	1.805	4.749	64.414%
DCRNN	GCN	GRU	1.269	2.944	46.203%
STG2Seq	GCN	Attention	1.578	4.355	56.302%
DGSL	GCN	GRU	1.244	2.906	45.269%
Informer	Attention	Attention	1.272	2.756	45.393%
PVCGN	GCN+MLP	GRU	1.241	2.697	44.290%
C-AHGCSP	GCN	LSTM	1.239	2.694	44.200%
HIAM	GCN	GRU	1.196	2.581	42.690%
ODMixer (Ours)	MLP	MLP	**1.131**	**2.367**	**40.358%**

TABLE IV
PERFORMANCE OF OD PREDICTION ON SHMOD DATASET

Models	Spatial Learning	Temporal Learning	MAE (↓)	RMSE (↓)	wMAPE (↓)
HA	-	-	0.515	1.429	70.388%
LSTM	-	LSTM	0.507	1.789	69.261%
GRU	-	GRU	0.526	1.920	71.887%
Graph WaveNet	GCN	TCN	0.560	2.049	76.851%
HSTN	GCN+ Attention	GRU+ Attention	0.560	2.530	76.520%
MVPF	GAT	GRU	0.587	2.271	80.185%
DCRNN	GCN	GRU	0.442	1.392	61.357%
STG2Seq	GCN	Attention	0.651	2.703	88.887%
DGSL	GCN	GRU	0.431	1.238	59.820%
Informer	Attention	Attention	0.482	1.118	65.883%
PVCGN	GCN+MLP	GRU	0.441	1.229	60.184%
C-AHGCSP	GCN	LSTM	0.564	2.096	77.039%
HIAM	GCN	GRU	0.441	1.226	60.268%
ODMixer (Ours)	MLP	MLP	**0.408**	**1.096**	**55.691%**

is inferior to that of HA, potentially due to their oversimplified design that neglects the spatial-temporal dependencies among metro stations. C-AHGCSP addresses the issue of incomplete OD matrices by introducing a dedicated module for completion. However, its approach involves sequentially filling in each time interval step by step. And it primarily utilizes the flow information from the most recent moment and uses simple module to learn dependencies. Therefore it performs well in smaller-scale metro networks, it faces limitations when dealing with larger-scale metro networks. The incorporation of spatial-temporal relations in models like DCRNN and Informer has resulted in better performance. DGSL outperforms other baselines on the SHMOD dataset because it learns the graph structure rather than using the provided structure, which allows for a more effective capture of relations on complex dataset. PVCGN, which considers three types of graphs and comprehensively accounts for the inter-station relations, exhibits good performance across both datasets. HIAM, which integrates OD and DO information and considers the incompleteness of OD matrices, achieves commendable results but performs less effectively than DGSL on the SHMOD dataset, possibly because the large scale of SHMOD requires a more fine-grained approach than can be provided by neighborhood graph aggregation alone to capture the complex interactions among numerous stations.

TABLE V
PERFORMANCE OF DIFFERENT VARIANTS ON HZMOD DATASET

Variants	MAE (↓)	RMSE (↓)	wMAPE (↓)
ODMixer-w/o OMP	1.159	2.487	41.366%
ODMixer-w/o CM	1.138	2.390	40.614%
ODMixer-w/o OM	1.149	2.409	41.015%
ODMixer-w/o DM	1.154	2.428	41.177%
ODMixer-w/o MM	1.196	2.534	42.675%
ODMixer-w/o BTL	1.132	2.375	40.397%
ODMixer-w/o PB	1.137	2.388	40.581%
ODMixer (Ours)	**1.131**	**2.367**	**40.358%**

TABLE VI
PERFORMANCE OF DIFFERENT VARIANTS ON SHMOD DATASET

Variants	MAE (↓)	RMSE (↓)	wMAPE (↓)
ODMixer-w/o OMP	0.418	1.126	57.064%
ODMixer-w/o CM	0.410	1.107	56.039%
ODMixer-w/o OM	0.414	1.133	56.602%
ODMixer-w/o DM	0.416	1.126	56.862%
ODMixer-w/o MM	0.420	1.152	57.319%
ODMixer-w/o BTL	0.410	**1.090**	55.979%
ODMixer-w/o PB	0.412	1.100	56.257%
ODMixer (Ours)	**0.408**	1.096	**55.691%**

In contrast, ODMixer models the OD matrix with fine granularity from the perspective of OD pairs. Subsequently, it leverages spatial-temporal relations among OD pairs through the ODIM module and integrates long-term temporal dynamics with the BTL module, thereby achieving superior performance.

C. Ablation Study

In this subsection, we aim to validate the effectiveness of essential modules within ODMixer. By systematically removing each module, we derive variants of ODMixer. Comparing the performance of these variants against the original ODMixer allows us to understand the specific contributions of each module to the overall model. The following variants will be evaluated on two datasets:

- *ODMixer-w/o OMP:* Remove OD Matrix Processing module, eliminating the impact of unfinished orders.
- *ODMixer-w/o CM:* Remove Channel Mixer, eliminating short-term temporal relations modeling.
- *ODMixer-w/o OM:* Remove Origin Mixer, eliminating the Origin perspective in OD pair relations modeling.
- *ODMixer-w/o DM:* Remove Des Mixer, eliminating the Destination perspective in OD pair relations modeling.
- *ODMixer-w/o MM:* Remove Multi-view Mixer, eliminating relations modeling among OD pairs.
- *ODMixer-w/o BTL:* Remove BTL, eliminating the perception of long-term traffic changes.
- *ODMixer-w/o PB:* Remove Prev Branch and BTL, eliminating the usage of historical information.

The results are presented in the Tables V and VI. ODMixer-w/o OMP removes the OD Matrix Processing module, the Cur

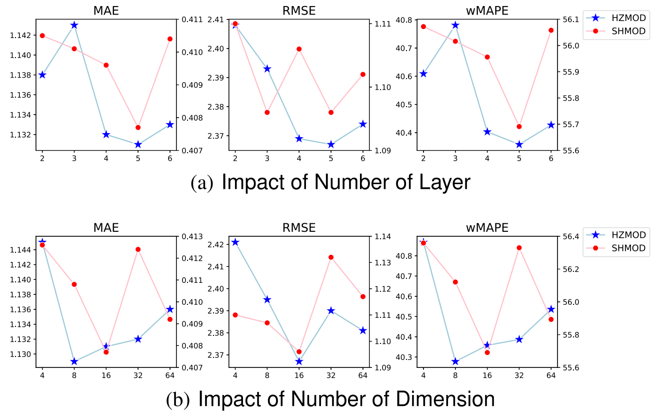


Fig. 4. Impact of Hyper-parameters. The two figure illustrate the impact of different parameters on prediction results across two datasets. Figure (a) displays the effect of varying the number of layers on performance, while Figure (b) shows the influence of different feature dimensions on performance.

Branch receives only the incomplete OD matrix as input, without any completion processing. This incompleteness results in the model lacking critical information needed to capture the traffic characteristics at the current time, leading to an inaccurate reflection of the actual traffic situation. Consequently, the model struggles to make effective predictions, and its performance drops significantly. ODMixer-w/o CM exhibits inferior performance compared to ODMixer, signifying the effectiveness of Channel Mixer in modeling short-term temporal relations of OD pairs. ODMixer-w/o MM performs less effectively than both ODMixer-w/o OM and ODMixer-w/o DM, highlighting the necessity of both Origin and Destination perspectives. The results indicate that these two perspectives complement each other in considering the relations among OD pairs. The Multi-view Mixer's effectiveness can be attributed to the fact that each OD pair is associated with both an origin and a destination, and the attributes of these locations influence the passenger flow dynamics. For example: If the origin is a residential area and the destination is a business district, there is typically a morning peak in passenger flow, with lower flows during other times. These patterns validate the effectiveness and rationale of modeling spatial relations from both dimensions. ODMixer-w/o BTL and ODMixer-w/o PB show some performance losses, suggesting that accounting for the temporal similarity of metro flows can enhance the model's performance. Due to the difficulty of accurately modeling long-term dependencies for large-scale metro data, ODMixer performs slightly worse than ODMixer-w/o BTL in RMSE metrics. These experimental findings affirm that all modules contribute to the ODMixer.

D. Hyper-Parameters Analysis

We conduct experiments to optimize the hyperparameters of ODMixer, with the results depicted in Fig. 4. The hyperparameters for ODMixer include the number of layers L and the number of feature dimensions d. Notably, when L is small, the model exhibits weak feature extraction capabilities, resulting in comparatively inferior performance. Conversely, a large L introduces fluctuations in performance. After comprehensive experimentation, we establish the optimal setting as $L = 5$. Regarding the dimension of the hidden layer, experimental findings indicate that $d = 16$ is a better choice for achieving desirable model performance.

TABLE VII
EFFICIENCY STUDY OF ODMIXER ON HZMOD DATASET

Models	MAE	Parameters (M)	Train (s)	Infer (s)
LSTM	1.387	1.77	2.14	0.30
GRU	1.427	1.33	2.18	0.31
Graph WaveNet	1.717	1.21	1.82	0.16
HSTN	1.692	2.18	6.51	1.86
MVPF	1.805	2.51	3.55	1.66
DCRNN	1.269	6.54	13.40	1.93
STG2Seq	1.578	1.06	2.17	0.23
DGSL	1.244	8.55	12.44	1.78
Informer	1.272	88.69	2.21	2.76
PVCGN*	1.241	55.84	94.11	10.43
C-AHGCSP	1.239	2.17	3.98	1.72
HIAM	1.196	13.89	13.21	1.25
ODMixer (Ours)	**1.131**	2.12	5.61	1.18

* The Batch Size in PVCGN is 16.

TABLE VIII
EFFICIENCY STUDY OF ODMIXER ON SHMOD DATASET

Models	MAE	Parameters (M)	Train (s)	Infer (s)
LSTM	0.507	2.24	55.44	9.91
GRU	0.526	1.70	52.19	10.48
Graph WaveNet	0.560	1.44	43.74	3.88
HSTN	0.560	6.98	80.81	10.93
MVPF	0.587	3.81	40.22	8.67
DCRNN	0.442	28.17	165.19	27.55
STG2Seq	0.651	4.07	27.66	4.48
DGSL	0.431	35.33	240.37	39.76
Informer	0.482	637.43	-	-
PVCGN	0.441	178.15	640.56	90.72
C-AHGCSP	0.564	2.31	61.36	14.01
HIAM	0.441	28.01	286.68	36.19
ODMixer (Ours)	**0.408**	26.75	247.91	26.86

E. Efficiency Analysis

In this subsection, we quantify the number of parameters for the various models. To ensure a fair comparison, all models are evaluated on a single NVIDIA GeForce RTX 2080 Ti GPU with a consistent batch size. The batch size is 8 for SHMOD and 32 for HZMOD, respectively.

Tables VII and VIII present a comparative analysis of all methods. The LSTM, GRU, Graph WaveNet, STG2Seq, HSTN and MVPF models exhibit poorer performance than HA, despite their shorter training and inference times. Because C-AHGCSP completes the OD matrix step by step introduces additional computational overhead, it needs longer inference times. Despite Informer's short training and inference times on the HZMOD dataset, it has a substantial number of parameters. Regarding the SHMOD dataset, Informer cannot be trained, even when the batch size is 1. Furthermore, the PVCGN model is unable to be trained on a single GPU with a batch size of 32 for HZMOD, requiring a reduction to 16 for testing. Despite its good

TABLE IX
VALIDITY VERIFICATION ON HZMOD DATASET

Models	wMAPE (↓)	Train (s)	Test (s)	Layer	Dim	BatchSize	GPUs
ODTrans[1]	47.422%	19.02	3.92	2	16	8	8
ODMixer	**40.609%**	2.85	0.88	2	16	32	1
Δ	14.37%	×6.67	×4.45	-	-	-	-
ODTrans[1]	47.007%	27.80	4.70	3	16	8	8
ODMixer	**40.781%**	3.75	0.98	3	16	32	1
Δ	13.24%	×7.41	×4.80	-	-	-	-

[1] For the Transformer, we set the batch size to 8, the number of layers to 2 or 3, and use 8 GPUs to ensure proper execution.

performance, the PVCGN model requires significantly longer training and inference times. In contrast, the HIAM model, which serves as the best-performance baseline, has a higher number of parameters, as well as longer training and inference times compared to ODMixer. Our proposed ODMixer not only achieves superior performance but also maintains reasonable training and inference times.

F. Validity Verification

Since our method is based on the perspective of OD pairs, while most existing methods are station-based, we aim to verify whether our model structure can effectively address the issues in existing methods from the same perspective. To this end, we replace the ODIM module in ODMixer with a standard Attention module, naming the modified model ODTrans. The experimental results are presented in the following Table IX.

As described in the introduction, the number of input tokens is N^2 from the perspective of OD pairs, resulting in space and time complexities of N^4 for the attention mechanism. Consequently, ODTrans cannot run with the same number of GPUs and batch size as ODMixer when using the same model layers and dimensions. For a fair performance comparison, we reduce the number of model layers and batch size and employe multiple GPUs for ODTrans. We also use the same number of layers for training ODMixer to ensure fairness.

The experiments show that the performance of ODMixer outperforms ODTrans by 14.37% and 13.24% at layers 2 and 3, respectively. Furthermore, by comparing training and testing times, we find that ODMixer's training time and testing time at layer 2 are accelerated by 6.67 times and 4.45 times, respectively. At layer 3, training and testing times are accelerated by 7.41 times and 4.80 times, respectively. These results demonstrate that the ODMixer module, tailored for the metro OD prediction task, effectively captures the spatiotemporal relations in traffic data. This significantly enhances the model's operational efficiency, reduces resource requirements, and greatly improves its deployability.

G. Case Study

In this subsection, we use two practical examples to demonstrate why our ODMixer achieves better results.

As shown in Fig. 5(a), the OD traffic on Saturday and Sunday (Sep 10-11) is low and stable, allowing both DGSL and ODMixer to fit the real traffic well. However, during the peak periods of Monday and Tuesday (Sep 12-13), the traffic between

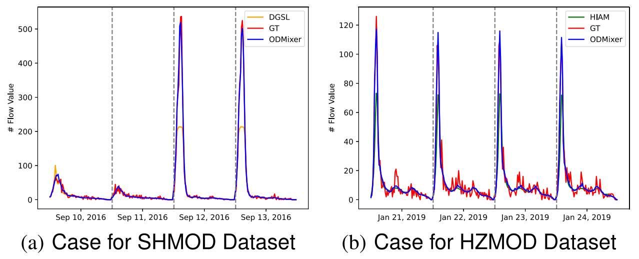


Fig. 5. Case for SHMOD and HZMOD Dataset. (a) shows the prediction results of the DGSL and ODMixer models from September 10 to 13, with the 12th being a Monday. (b) shows the prediction results of the HIAM and ODMixer models from January 21 to 24, with the 21th being a Monday. It can be observed that when traffic flow changes rapidly, the performance of HIAM and DGSL is significantly worse than that of ODMixer.

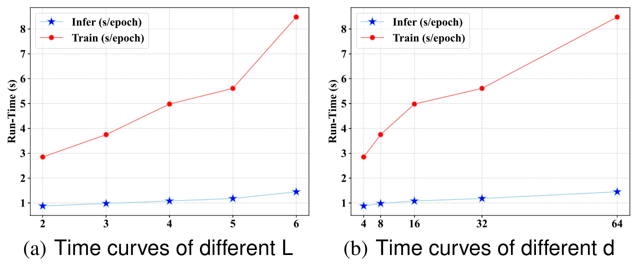


Fig. 6. Comparasions between running time and various L, d.

OD pairs changes rapidly. Although DGSL senses the rise and fall of traffic, it underestimates the peak values. Conversely, ODMixer quickly detects these rapid changes and accurately estimates the peak value of the OD pair's traffic. This is mainly due to ODMixer's input modeling based on the OD pair perspective. The Channel Mixer captures the short-term temporal relations, the Multi-view Mixer learns the spatial relations between OD pairs, and BTL senses long-term traffic similarity. These designs enable ODMixer to handle rapid changes and complex relations in traffic effectively, significantly improving the model's prediction accuracy and adaptability. A similar situation is observed with HIAM and our ODMixer in Fig. 5(b).

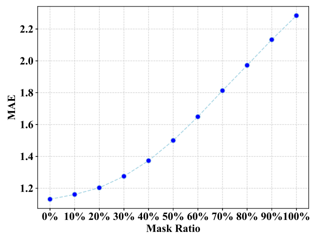
H. Scalability

Scalability tests are performed to assess the impact of various hyperparameters on model efficiency, with a primary focus on the number of layers L and the hidden layer dimension d. First, we evaluate the effect of the number of layers. As shown in Fig. 6(a), both training and inference times increase as the number of layers grows, although the increase in inference time is relatively smaller. Considering the trade-off between performance (Fig. 4(a)) and computational cost, we select 5 layers to strike a balance between effectiveness and efficiency.

Next, we examine the effect of the hidden layer dimension, as depicted in Fig. 6(b). The results indicate that training time varies with the hidden layer dimension, while inference time remains relatively stable. Based on overall performance metrics (Fig. 4(b)), we choose a hidden layer dimension of 16. Overall, the model's inference time exhibits minimal variation,

TABLE X
PREDICTION ERROR OF MODELS ON ORIGINAL DATA(1ST ROW), GAUSSIAN-NOISE POLLUTED DATA(2ND ROW) AND THE RELATIVE INCREMENT RATIO OF THE ERROR(3RD ROW)

Model	HZMOD			SHMOD		
	MAE	RMSE	wMAPE	MAE	RMSE	wMAPE
HIAM	1.196	2.581	42.690%	0.441	1.226	60.268%
+ $\mathcal{N}(0,1)$	1.214	2.638	43.337%	0.454	1.279	62.049%
+Δerrors	1.51%	2.21%	1.52%	2.95%	4.32%	2.96%
ODMixer	1.131	2.367	40.358%	0.408	1.096	55.691%
+ $\mathcal{N}(0,1)$	1.136	2.380	40.538%	0.413	1.122	56.384%
+Δerrors	0.44%	0.55%	0.45%	1.23%	2.37%	1.24%


(a) HZMOD

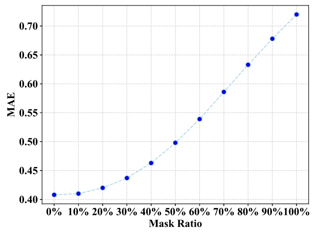

(b) SHMOD

Fig. 7. MAE curve for different missing ratio on HZMOD and SHMOD.

and training time remains within acceptable limits. These findings confirm that ODMixer can be efficiently trained on larger datasets and deployed in real-time applications. Thus, ODMixer demonstrates strong stability and adaptability under different hyperparameter settings, making it highly suitable for real-time scenarios.

I. Robustness

In real-world metro systems, hardware or software failures such as sensor malfunctions or network issues may result in missing order data, leading to discrepancies between the obtained traffic data and the actual data. To assess the robustness of the model under such conditions, we evaluate its performance under noisy and missing data scenarios. For the noisy data case, we add Gaussian noises into the raw data of test dataset and test the model on the noisy test dataset using the trained model [57]. The results in Table X show the increasing errors of ODMixer are much less than SOTA for the noise data, verifying the robustness of ODMixer.

For the missing data scenario, we perform spatiotemporal masking on the test dataset, randomly masking a portion of the data to simulate potential missing data situations. The results in Fig. 7 show the prediction errors is gradually increasing when mask ratio is increasing. When the mask ratio is relatively low (not greater than 30%), ODMixer still maintains good performance compared to the SOTA models, demonstrating that ODMixer has some resilience to missing data. Through experiments in both noisy and missing data scenarios, we further demonstrate that ODMixer exhibits robust performance, effectively handling common issues of noise and data missing in real-world applications. This ensures the model's stability and reliablity in real-world metro environments. Traffic prediction robustness focuses on enhancing a model's resilience to noise and handling missing data, which often referred to as the imputation task. Several studies have explored model robustness [57], [58]. Future work could integrate noise resilience and imputation modules to preprocess noisy or incomplete data, improving prediction quality.

TABLE XI
CROSS-CITY PERFORMANCE OF DIFFERENT METHODS

Method		SHMOD → HZMOD			HZMOD → SHMOD		
		MAE	RMSE	wMAPE	MAE	RMSE	wMAPE
FF	OAda	1.169	2.481	41.700%	0.436	1.163	59.553%
	TAda	1.169	2.462	41.732%	0.431	1.148	58.929%
OFA	OAda	1.212	2.580	43.243%	0.449	1.231	61.359%
	TAda	1.195	2.532	42.632%	0.435	1.165	59.357%

FF: full fine-tuning. OFA: only fine-tuning adapter. OAda: one adapter. TAda: two adapters.

J. Cross-City Generalizability

We conduct a comprehensive experiment using two real-world datasets with distinct characteristics. The results demonstrate that our model can effectively capture the complex spatiotemporal relations inherent in different datasets and is well-suited for deployment in metro systems with varying characteristics. To further validate the generalization capability of our model, we design the cross-city experiment, where the model is trained on the metro network of one city and fine-tuned and tested on a different city's network with distinct characteristics. However, due to differences in the number of stations across cities, the number of OD pairs also varies.

Therefore, to ensure the model operates properly, we map the number of stations at the input stage accordingly. We adopt two efficient mapping methods, OAda and TAda, to address differences in the number of stations between cities. These methods use Adapters to map the number of stations from the source city to the target city ($N_{source\_city} \rightarrow N_{target\_city}$), where N_{source_city} and N_{target_city} represent the numbers of metro stations in the source and target cities, respectively (e.g., HZMOD has 80 stations, while SHMOD has 288). Each Adapter is implemented as a two-layer MLP. In the OAda method, the origin and destination share a single Adapter, whereas in the TAda method, the origin and destination each use separate Adapters. We test the performance of two fine-tuning approaches: full model fine-tuning and Adapter-layer-only fine-tuning. The experimental results, shown in the Table XI, indicate that full model fine-tuning outperforms Adapter-layer-only fine-tuning and achieves better results than the current SOTA methods. Interestingly, Adapter-layer-only fine-tuning performs comparably to SOTA and even slightly surpasses it under the TAda configuration. Comparing the two mapping methods, TAda generally outperforms OAda, especially when only the Adapter layer is fine-tuned. These results demonstrate that ODMixer has strong cross-city generalization and adaptability, effectively transferring knowledge between metro networks with different attributes.

Recent studies focused on enhancing the generalization of traffic prediction models [59], [60]. They improved the model's ability to transfer knowledge across datasets with diverse attributes. The primary objective is to train models on large, data-rich datasets and fine-tune them on smaller, domain-specific datasets, addressing challenges related to data scarcity. In future work, our ODMixer could be extended to a more generalized metro OD prediction model.

VI. Conclusion

In this paper, we introduce ODMixer, a fine-grained spatial-temporal MLP architecture for metro OD prediction problem. Our ODMixer learns the short-term temporal relations of OD pairs by incorporating the Channel Mixer. The Multi-view Mixer efficiently captures OD pair relations from both origin and destination perspectives. With the integration of BTL, our ODMixer can perceive long-term traffic changes. Experimental results represent ODMixer's outstanding performance on two large-scale datasets. Future work could explore incorporating external factors such as weather, events, urban population distribution, regional composition, and Points of Interest (POIs). Additionally, enhancing the model's ability to learn general traffic flow patterns may improve its transferability and scalability. Deploying ODMixer in real-world metro systems could support the management and optimization of metro operations, thereby increasing transportation efficiency.

However, real-world deployment presents several key challenges, including efficient data processing, maintaining model performance under varying conditions, and integration with existing metro infrastructure. We also plan to conduct real-world testing to demonstrate the model's practical applicability.

References

[1] G. Jin et al., "Spatio-temporal graph neural networks for predictive learning in urban computing: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 10, pp. 5388–5408, Oct. 2024.

[2] K. Wang, L. Liu, Y. Liu, G. Li, F. Zhou, and L. Lin, "Urban regional function guided traffic flow prediction," *Inf. Sci.*, vol. 634, pp. 308–320, 2023.

[3] J. Lin, Y. Zhu, L. Liu, Y. Liu, G. Li, and L. Lin, "DenseLight: Efficient control for large-scale traffic signals with dense feedback," in *Proc. 32nd Int. Joint Conf. Artif. Intell.*, 2023, pp. 6058–6066.

[4] H. Li et al., "A survey on graph neural networks in intelligent transportation systems," 2024, arXiv: *2401.00713*.

[5] F. Zheng, J. Zhao, J. Ye, X. Gao, K. Ye, and C. Xu, "Metro OD matrix prediction based on multi-view passenger flow evolution trend modeling," *IEEE Trans. Big Data*, vol. 9, no. 3, pp. 991–1003, Jun. 2023.

[6] L. Liu, Y. Zhu, G. Li, Z. Wu, L. Bai, and L. Lin, "Online metro origin-destination prediction via heterogeneous information aggregation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3574–3589, Mar. 2023.

[7] Y. Gong, Z. Li, J. Zhang, W. Liu, and Y. Zheng, "Online spatio-temporal crowd flow distribution prediction for complex metro system," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 865–880, Feb. 2022.

[8] Y. Xu et al., "Adaptive feature fusion networks for origin-destination passenger flow prediction in metro systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 5, pp. 5296–5312, May 2023.

[9] L. Shen, J. Li, Y. Chen, C. Li, X. Chen, and D.-H. Lee, "Short-term metro origin-destination passenger flow prediction via spatio-temporal dynamic attentive multi-hypergraph network," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 8, pp. 9945–9957, Aug. 2024.

[10] Y. Zhang, K. Sun, D. Wen, D. Chen, H. Lv, and Q. Zhang, "Deep learning for metro short-term origin-destination passenger flow forecasting considering section capacity utilization ratio," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 8, pp. 7943–7960, Aug. 2023.

[11] J. Yang et al., "Spatiotemporal virtual graph convolution network for key origin-destination flow prediction in metro system," *Math. Problems Eng.*, vol. 2022, 2022, Art. no. 5622913.

[12] I. O. Tolstikhin et al., "MLP-mixer: An all-MLP architecture for vision," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 24261–24272.

[13] S. Lin et al., "MLP can be a good transformer learner," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 19489–19498.

[14] B. M. Williams, P. K. Durvasula, and D. E. Brown, "Urban freeway traffic flow prediction: Application of seasonal autoregressive integrated moving average and exponential smoothing models," *Transp. Res. Rec.*, vol. 1644, no. 1, pp. 132–141, 1998.

[15] Y. Sun, B. Leng, and W. Guan, "A novel wavelet-SVM short-time passenger flow prediction in Beijing subway system," *Neurocomputing*, vol. 166, pp. 109–121, 2015.

[16] W. Elleuch, A. Wali, and A. M. Alimi, "Intelligent traffic congestion prediction system based on ANN and decision tree using big GPS traces," in *Proc. 16th Int. Conf. Intell. Syst. Des. Appl.*, Springer, 2017, pp. 478–487.

[17] W. Zeng, C. Lin, K. Liu, J. Lin, and A. K. Tung, "Modeling spatial nonstationarity via deformable convolutions for deep traffic flow prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 2796–2808, Mar. 2023.

[18] H. Qu, Y. Gong, M. Chen, J. Zhang, Y. Zheng, and Y. Yin, "Forecasting fine-grained urban flows via spatio-temporal contrastive self-supervision," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 8, pp. 8008–8023, Aug. 2023.

[19] Q. Zhaowei, L. Haitao, L. Zhihui, and Z. Tao, "Short-term traffic flow forecasting method with M-B-LSTM hybrid network," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 225–235, Jan. 2022.

[20] C. Ma, G. Dai, and J. Zhou, "Short-term traffic flow prediction for urban road sections based on time series analysis and LSTM_BILSTM method," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 5615–5624, Jun. 2022.

[21] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 922–929.

[22] H. Hu, Z. Lin, Q. Hu, and Y. Zhang, "Attention mechanism with spatial-temporal joint model for traffic flow speed prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 16612–16621, Sep. 2022.

[23] D. Wang, W. Cao, J. Li, and J. Ye, "DeepSD: Supply-demand prediction for online car-hailing services using deep neural networks," in *Proc. IEEE 33rd Int. Conf. Data Eng.*, 2017, pp. 243–254.

[24] J. Zhang, Y. Zheng, J. Sun, and D. Qi, "Flow prediction in spatio-temporal networks based on multitask deep learning," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 3, pp. 468–478, Mar. 2020.

[25] R. Jiang et al., "DeepCrowd: A deep model for large-scale citywide crowd density and flow prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 276–290, Jan. 2023.

[26] X. Wang et al., "Traffic flow prediction via spatial temporal graph neural network," in *Proc. Web Conf.*, 2020, pp. 1082–1092.

[27] X. Zhang et al., "Traffic flow forecasting with spatial-temporal graph diffusion network," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 15008–15015.

[28] J. Ou et al., "STP-TrellisNets+: Spatial-temporal parallel TrellisNets for multi-step metro station passenger flow prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 7, pp. 7526–7540, Jul. 2023.

[29] P. Xie et al., "Spatio-temporal dynamic graph relation learning for urban metro flow prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 10, pp. 9973–9984, Oct. 2023.

[30] Y. Fang, Y. Qin, H. Luo, F. Zhao, and K. Zheng, "STWave+: A multi-scale efficient spectral graph attention network with long-term trends for disentangled traffic flow forecasting," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 6, pp. 2671–2685, Jun. 2024.

[31] M. Wang, Y. Zhang, X. Zhao, Y. Hu, and B. Yin, "Traffic origin-destination demand prediction via multichannel hypergraph convolutional networks," *IEEE Trans. Computat. Social Syst.*, vol. 11, no. 4, pp. 5496–5509, Aug. 2024.

[32] B. Huang, K. Ruan, W. Yu, J. Xiao, R. Xie, and J. Huang, "ODformer: Spatial–temporal transformers for long sequence origin–destination matrix forecasting against cross application scenario," *Expert Syst. Appl.*, vol. 222, 2023, Art. no. 119835.

[33] Y. He, Y. Zhao, and K.-L. Tsui, "Short-term forecasting of origin-destination matrix in transit system via a deep learning approach," *Transportmetrica A: Transport Sci.*, vol. 19, no. 2, 2023, Art. no. 2033348.

[34] R. Jiang et al., "Countrywide origin-destination matrix prediction and its application for COVID-19," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discov. Databases Appl. Data Sci. Track*, Springer, 2021, pp. 319–334.

[35] J. Zhang, H. Che, F. Chen, W. Ma, and Z. He, "Short-term origin-destination demand prediction in urban rail transit systems: A channel-wise attentive split-convolutional neural network method," *Transp. Res. Part C: Emerg. Technol.*, vol. 124, 2021, Art. no. 102928.
[36] J. Ye, J. Zhao, F. Zheng, and C. Xu, "Completion and augmentation-based spatiotemporal deep learning approach for short-term metro origin-destination matrix prediction under limited observable data," *Neural Comput. Appl.*, vol. 35, no. 4, pp. 3325–3341, 2023.
[37] G. Zhu, J. Ding, Y. Wei, Y. Yi, S. S.-D. Xu, and E. Q. Wu, "Two-stage OD flow prediction for emergency in urban rail transit," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 1, pp. 920–928, Jan. 2024.
[38] H. Liu, Z. Dai, D. So, and Q. V. Le, "Pay attention to MLPs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 9204–9215.
[39] H. Touvron et al., "ResMLP: Feedforward networks for image classification with data-efficient training," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 4, pp. 5314–5321, Apr. 2023.
[40] C. Tang, Y. Zhao, G. Wang, C. Luo, W. Xie, and W. Zeng, "Sparse MLP for image recognition: Is self-attention really necessary?," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 2344–2351.
[41] M. Li, X. Zhao, C. Lyu, M. Zhao, R. Wu, and R. Guo, "MLP4Rec: A pure MLP architecture for sequential recommendations," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, 2022, pp. 2138–2144.
[42] Z. Zhang et al., "MLPST: MLP is all you need for spatio-temporal prediction," in *Proc. 32nd ACM Int. Conf. Inf. Knowl. Manage.*, 2023, pp. 3381–3390.
[43] V. Ekambaram, A. Jati, N. Nguyen, P. Sinthong, and J. Kalagnanam, "TSMixer: Lightweight MLP-mixer model for multivariate time series forecasting," 2023, arXiv: *2306.09364*.
[44] S.-A. Chen, C.-L. Li, N. Yoder, S. O. Arik, and T. Pfister, "TSMixer: An all-MLP architecture for time series forecasting," 2023, arXiv: *2303.06053*.
[45] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, arXiv: *1607.06450*.
[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[47] L. S.-T. Memory, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 2010.
[48] K. Cho et al., "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1724–1734.
[49] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 1907–1913.
[50] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Representations*, 2018.
[51] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Q. Z. Sheng, "STG2Seq: Spatial-temporal graph to sequence model for multi-step passenger demand forecasting," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 1981–1987.
[52] L. Liu, J. Chen, H. Wu, J. Zhen, G. Li, and L. Lin, "Physical-virtual collaboration modeling for intra-and inter-station metro ridership prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 4, pp. 3377–3391, Apr. 2022.
[53] C. Shang, J. Chen, and J. Bi, "Discrete graph structure learning for forecasting multiple time series," 2021, arXiv: *2101.06861*.
[54] H. Zhou et al., "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 11106–11115.
[55] T. Chen, L. Nie, J. Pan, L. Tu, B. Zheng, and X. Bai, "Origin-destination traffic prediction based on hybrid spatio-temporal network," in *Proc. IEEE Int. Conf. Data Mining*, 2022, pp. 879–884.
[56] J. Ye, J. Zhao, F. Zheng, and C. Xu, "A heterogeneous graph convolution based method for short-term OD flow completion and prediction in a metro system," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 11, pp. 15614–15627, Nov. 2024.
[57] J. Jiang, B. Wu, L. Chen, K. Zhang, and S. Kim, "Enhancing the robustness via adversarial learning and joint spatial-temporal embeddings in traffic forecasting," in *Proc. 32nd ACM Int. Conf. Inf. Knowl. Manage.*, 2023, pp. 987–996.
[58] X. Chen, Z. Cheng, H. Cai, N. Saunier, and L. Sun, "Laplacian convolutional representation for traffic time series imputation," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 11, pp. 6490–6502, Nov. 2024.
[59] Y. Zhang, H. Lu, N. Liu, Y. Xu, Q. Li, and L. Cui, "Personalized federated learning for cross-city traffic prediction," in *Proc. 33rd Int. Joint Conf. Artif. Intell.*, 2024, pp. 5526–5534.
[60] J. Mo and Z. Gong, "Cross-city multi-granular adaptive transfer learning for traffic flow prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 11246–11258, Nov. 2023.

Yang Liu (Member, IEEE) received the PhD degree from Xidian University, in 2019. He is currently an associate professor working with the School of Computer Science and Engineering, Sun Yat-sen University. His current research interests include multi-modal reasoning, causality learning, and embodied AI. He is the recipient of the First Prize of the Third Guangdong Province Young Computer Science Academic Show. He has authorized and co-authorized more than 40 papers in top-tier academic journals and conferences such as *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Image Processing*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE/ASME Transactions on Mechatronics*, CVPR, ICCV, IJCAI, and ACM MM.

Binglin Chen received the BE degree from the School of Computer Science and Engineering, Jilin University, Changchun, China, in 2022, and he is currently working toward the master's degree with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. His current research interests include deep learning, data mining, and time series analysis.

Yongsen Zheng received the PhD degree in computer science and technology from Sun Yat-sen University, Guangzhou, China, in 2023. She is currently a research fellow with the College of Computing and Data Science, Nanyang Technological University, Singapore, and is also with Digital Trust Centre, Singapore (DTC). Her current research interests include human-AI dialogue system, conversational recommender system, trustworthy recommendation, NLP, trustworthy AI, AI safety, large language models, and causal reasoning.

Lechao Cheng (Member, IEEE) received the PhD degree from the College of Computer Science and Technology, Zhejiang University, Hangzhou, China, in 2019. He is currently an associate professor with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, China. He has contributed more than 50 papers to renowned journals and conferences such as the *International Journal of Computer Vision*, *IEEE Transactions on Medical Imaging*, *IEEE Transactions on Cybernetics*, *IEEE Transactions on Multimedia*, CVPR, ICML, ECCV, AAAI, IJCAI, and ACM MM. His research area centers around vision knowledge transfer in deep learning.

Guanbin Li (Member, IEEE) received the PhD degree from the University of Hong Kong, in 2016. He is currently a full professor with the School of Computer Science and Engineering, Sun Yat-Sen University. His current research interests include computer vision, image processing, and deep learning. He is a recipient of ICCV 2019 Best Paper Nomination Award. He has authorized and co-authorized on more than 70 papers in top-tier academic journals and conferences. He serves as an area chair for the conference of VISAPP. He has been serving as a reviewer for numerous academic journals and conferences such as *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *International Journal of Computer Vision*, *IEEE Transactions on Image Processing*, *IEEE Transactions on Multimedia*, *IEEE Transactions on Cybernetics*, CVPR, ICCV, ECCV, and NeurIPS.

Liang Lin (Fellow, IEEE) is a full professor of computer science with Sun Yat-sen University. He served as the executive director and distinguished scientist of SenseTime Group from 2016 to 2018, leading the R&D teams for cutting-edge technology transferring. He has authored or co-authored more than 300 papers in leading academic journals and conferences, and his papers have been cited by more than 40,000 times. He is an associate editor of *IEEE Transactions on Neural Networks and Learning Systems* and *IEEE Transactions on Multimedia*, and served as area chairs for numerous conferences such as CVPR, ICCV, SIGKDD and AAAI. He is the recipient of numerous awards and honors including Wu Wen-Jun Artificial Intelligence Award, the First Prize of China Society of Image and Graphics, ICCV Best Paper Nomination in 2019, Annual Best Paper Award by Pattern Recognition (Elsevier) in 2018, Best Paper Dimond Award in IEEE ICME 2017, Google Faculty Award in 2012. His supervised PhD students received ACM China Doctoral Dissertation Award, CCF Best Doctoral Dissertation, and CAAI Best Doctoral Dissertation. He is a fellow of the IAPR/IET.