applied sciences

MDPI

Article

Multi-Step Passenger Flow Prediction for Urban Metro System Based on Spatial-Temporal Graph Neural Network

Yuchen Chang [1], Mengya Zong [1], Yutian Dang [2] and Kaiping Wang [1,*]

[1] Department of Computer Science, Xi'an University of Architecture and Technology, Xi'an 710055, China; 1979457479cl@gmail.com (Y.C.); zongmy2000@outlook.com (M.Z.)
[2] College of Architecture, Xi'an University of Architecture and Technology, Xi'an 710055, China; dangyt@xauat.edu.cn
* Correspondence: wkp@xauat.edu.cn

Abstract: Efficient operation of urban metro systems depends on accurate passenger flow predictions, a task complicated by intricate spatiotemporal correlations. This paper introduces a novel spatiotemporal graph neural network (STGNN) designed explicitly for predicting multistep passenger flow within metro stations. In the spatial dimension, previous research primarily focuses on local spatial dependencies, struggling to capture implicit global information. We propose a spatial modeling module that leverages a dynamic global attention network (DGAN) to capture dynamic global information from all-pair interactions, intricately fusing prior knowledge from the input graph with a graph convolutional network. In the temporal dimension, we design a temporal modeling module tailored to navigate the challenges of both long-term and recent-term temporal passenger flow patterns. This module consists of series decomposition blocks and locality-aware sparse attention (LSA) blocks to incorporate multiple local contexts and reduce computational complexities in long sequence modeling. Experiments conducted on both simulated and real-world datasets validate the exceptional predictive performance of our proposed model.

Keywords: metro system; multistep passenger flow prediction; dynamic global attention; locality-aware sparse attention; graph convolutional network

check for updates

Citation: Chang, Y.; Zong, M.; Dang, Y.; Wang, K. Multi-Step Passenger Flow Prediction for Urban Metro System Based on Spatial-Temporal Graph Neural Network. *Appl. Sci.* **2024**, *14*, 8121. https://doi.org/10.3390/app14188121

Academic Editor: Christos Bouras

Received: 29 July 2024
Revised: 1 September 2024
Accepted: 4 September 2024
Published: 10 September 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the rapid growth of the urban population, immense traffic pressure has inevitably been imposed on transportation systems. Compared to other transportation modes, rail transit, with its efficiency and high passenger capacity, is integral to sustainable development. As the rail transit network expands and urban land remains limited, metro stations have become larger and deeper to alleviate traffic congestion. This has resulted in significant passenger flows, especially during peak hours. Passenger flow is a fundamental measurement reflecting the crowding level in the increasingly extensive and complex metro network system. Accurately and effectively predicting passenger flow enhances overall metro operational efficiency, alleviates peak-hour congestion, and reduces operating costs. Therefore, as a crucial component of intelligent transportation systems, scientifically forecasting passenger flow holds essential practical significance [1]. Passenger flow data refers to time series data consecutively recorded within a specific time interval. The analysis of passenger flow data is the core of metro passenger flow prediction, helping to find the changes in passenger flow and providing the scientific basis for prediction.

In recent decades, there has been a sustained emphasis on time series prediction in academic research, with widespread applications across diverse fields such as financial market prediction [2], climate prediction [3], and traffic prediction [4,5]. Existing studies can be generally categorized into two classes: traditional methods and deep learning methods. Traditional methods include statistical analysis and machine learning approaches. Statistical analysis-based methods often treat passenger flow data as a time series prediction

problem. For instance, the Autoregressive Integrated Moving Average (ARIMA) model, developed from autoregressive models, is one of the most typical time series forecasting models. Refs. [6–8] proposed a framework for analyzing short-term passenger flow during the special event based on the ARIMA model, demonstrating its superior accuracy and reliability. In addition to passenger flow, ARIMA is widely employed for stock price forecasting [9] and traffic prediction [10]. While statistical forecasting models such as ARIMA have proven effective in certain contexts, they often struggle with the nonlinear capabilities [11] inherent in metro passenger flow data. Machine learning methods provide a more flexible, data-driven alternative for learning temporal dynamics [12]. Traditional machine learning models, such as K-Nearest Neighbors [13], Support Vector Machine [14], and neural network models [15], are trained on historical data through supervised learning for predictive tasks. While these models can capture specific nonlinear structures within metro passenger flow data, they still face limitations in feature engineering and accuracy in complex scenarios. Deep learning, however, has emerged as a powerful tool, capable of leveraging potent nonlinear modeling capabilities, capturing more abstract latent features from extensive datasets, and reducing the need for manual feature engineering and model design [16].

Over the past few years, deep learning methods have made remarkable breakthroughs [17]. In studies of traffic flow or passenger flow using neural networks, previous studies are generally based on Multilayer Perceptron (MLP) and their variants [18,19]. It generates intermediate hidden layer feature representations through a series of non-linear layers to ensure ease of implementation. Yet, their accuracy tends to decrease when handling high-dimensional complex data [20]. Recurrent neural network (RNN) is another deep learning model suitable for processing sequential data. Unlike Multilayer Perceptron, RNN units integrate current data and memory capabilities. However, retaining long-term memory often leads to vanishing and exploding gradient problems, making it difficult to capture long-term dependencies [21]. To tackle this challenge, Long Short-Term Memory (LSTM) and gated recurrent unit (GRU) introduce memory cells, a distinctive form of hidden states designed to retain additional information [22], achieving improved efficiency in traffic flow prediction tasks. Meanwhile, road networks and metro networks encapsulate traffic and passenger flow data, offering datasets rich in both temporal and spatial location information. Previous studies mainly focused on temporal dependencies, overlooking valuable spatial information, which led to an inadequate capture of essential spatial dependencies. Convolutional neural networks (CNN) and graph convolutional networks (GCN) play a pivotal role in modeling spatial dependencies. Ref. [23] transformed traffic velocity across the entire road network into a sequence of static images, utilizing deep convolutional neural networks (DCNN) to proficiently capture spatial correlations. Similarly, Ref. [24] represented public transit data as spatiotemporal image matrices, showcasing the efficacy of the CNN-LSTM model in sequence predictive tasks.

However, it is noteworthy that CNNs are inherently designed for Euclidean domain data (regular grids, etc.) and may require enhancement when handling non-Euclidean domain data, such as topological structure. With the rise of graph neural networks, GCNs are increasingly adopted to replace CNNs, generalizing the applicable scope of traditional convolution from regular grid-based data to arbitrary graph-structured data. Ref. [25] embedded GCN into gated recurrent units (GRUs) and integrated graph convolutions into recurrent units to enhance dynamic correlation extraction. The message-passing paradigm of GCN models makes it challenging for them to have a global perspective on variations, and they are also insensitive to changes in the temporal dimension. Attention Mechanism is a general method that, by introducing learnable weights, allows the model to have a global perspective and dynamically adjust its focus based on the significance of different input parts. In recent years, the attention mechanism has recently been widely used in multivariate time series prediction. SAnD [26] and DSANet [27] have demonstrated the effectiveness of attention mechanisms in multivariate time series modeling. Furthermore,

Ref. [28] integrated LSTM and attention mechanism, achieving excellent results in the tasks of predicting short-term passenger flow data in large-scale metro systems.

Despite the excellent performance of the aforementioned deep learning methods in prediction tasks, further exploration is needed in capturing both local and global dependencies of metro passenger flow. In the spatial dimension, traditional graph neural network models primarily focus on local spatial dependencies of nodes, making it difficult to capture global spatial dependencies. Therefore, combining GCN with a global attention mechanism could further explore the balance between local and global information in the spatial dimension. In the temporal dimension, traditional attention mechanisms are often insensitive to the contextual relationships of observations at time points. Thus, integrating convolutional networks with attention mechanisms can enhance the utilization of local contextual information, while series decomposition can better capture the global characteristics of time series. This paper proposes a novel STGNN framework for multistep passenger flow prediction tasks. The primary contributions of this research can be summarized as follows:

1. A novel spatiotemporal graph neural network (STGNN) is proposed for multistep passenger flow prediction in metro stations.
2. A spatial modeling module is proposed, which consists of a dynamic global attention network (DAGN) and graph convolution network (GCN). DAGN implicitly captures the dynamic influence of passenger flow variation between global node pairs, and GCN can integrate the structural information of the input graph.
3. A temporal modeling module consisting mainly of series decomposition blocks and locality-aware sparse attention blocks (LSA). A series decomposition block is employed to better capture the global characteristics of time series. At the same time, the LSA block can extract multiple local contexts and reduce the computational complexity for long sequence modeling.
4. Both simulation and real-world datasets are used in the experiments. For simulation data, the simulation scenario is the real-world 3D architecture model of Chengdu Metro Line 10 Shuangliu International Airport Terminal 1 station. The passenger flow data are acquired based on AnyLogic pedestrian simulation. For large-scale real-world data, the Automatic Fare Collection (AFC) data in Hangzhou, China, are acquired. The experimental results demonstrate that our proposed STGNN outperforms all those ten baselines.

The remainder of the paper is organized as follows. Section 2 summarizes the related works. Section 3 formulates the multi-step metro passenger flow predicting problem and describes the structure and mathematical formulation of the proposed STGNN model. Section 4 compares the multi-step prediction performances of the proposed model with other benchmark models based on both simulation and real-world datasets and conducts and analyzes the ablation experiments. Finally, Section 5 concludes the paper.

2. Related Works

2.1. Attention-Based Models

Since the Transformer [29] has emerged as a pivotal advancement in deep learning, the attention mechanism has been widely applied to time series tasks. This mechanism converts the input into queries, keys, and values and dynamically assigns weights to each value based on the relationship between the query and its corresponding key. Informer [30] and Reformer [31] achieve sparse attention for transformers by using strategies where Informer selects representative query prototypes and Reformer compares only similar queries and keys. These approaches reduce computation time and may improve forecasting. WHENS [32] combined wavelet attention and DTW attention with BiLSTM, enhancing the model's predictive capabilities while addressing the distortion issues in multivariate time series. ConvTan [33] proposed a new framework for multivariate time series tasks, which builds on the Transformer by introducing absolute and relative positions and combining them with convolutional layers to improve the position and data embedding of time series data. ITransformer [34] revisits the architecture of the Transformer by adopting an inverted

perspective, treating the entire time series of each variate as a separate token. Although the aforementioned methods have modified the attention mechanism in the Transformer architecture and achieved good results, they all overlook the drawback of dot-product self-attention being insensitive to the local context. Therefore, before implementing self-attention at time points, we attempt to use causal convolution to learn the local context of each point to mitigate this drawback.

2.2. Spatial-Temporal Graph Neural Networks

Spatiotemporal graph neural networks (GNNs) have become a common model for traffic forecasting due to their ability to handle spatiotemporal relationships effectively. MT-MGNN [35] combined Graph Attention Networks and Diffusion Graph Neural Networks into a multi-graph neural network, extracting complex features from various graphs by inputting adjacency matrices of different types of metro networks. MST-GRT [36] proposed a multi-graph neural network that aggregated metro passenger flow data from smaller to larger time granularities, enhancing the model's predictive performance. Ref. [37] designed a novel framework for metro passenger flow prediction, STP-TrellisNets+. This network can capture spatial correlations in passenger flow and both long-term and short-term correlations, while also addressing the flow discontinuity issue. Ref. [38] introduced the Spatiotemporal Multi-Graph Convolutional Network (ST-MGCN) by encoding the non-Euclidean pairwise relationships between regions into multiple graphs and utilizing multi-graph convolution to explain the impact of distant regions on the current regions. Ref. [39] extended this idea by adding the correlation of graph structure between preceding and succeeding time steps, proposing the Spatiotemporal Synchronous Graph Convolutional Network (STSGCN). Ref. [40] proposed a multi-level attention network to model the dynamic spatial-temporal dependencies among multiple geographic sensor time series. Ref. [41] integrated spatial-temporal attention mechanism and spatial-temporal convolution to capture dynamic spatial-temporal correlations and spatial patterns. Ref. [42] proposed an adaptive graph spatial-temporal transformer network to model the cross-spatial-temporal correlations on the spatial-temporal graph directly using local multi-head self-attention. Ref. [43] proposed a spatial-temporal memory-augmented multi-level attention network to explicitly model both short-term and long-term dependencies at different spatial scales. These methods primarily focus on the local spatial dependencies of graph network nodes, neglecting dynamic global spatial dependencies, which may result in the loss of information from distant nodes. Therefore, we attempt to capture local spatial dependencies using GCN while employing dynamic global attention to implicitly capture the dynamic influence of any global node pairs. Additionally, we use series decomposition to further extract the temporal features of both long-term and recent-term passenger flows, aiming to explore the potential of guiding passenger flow prediction from the perspective of series decomposition.

3. Methodology

3.1. Problem Formulation

Generally, passenger information primarily comprises three main variables: passenger flow, density, and speed. In this paper, the goal of multistep pedestrian flow forecasting is to predict multistep passenger flow in a certain future period based on historical passenger flow.

Definition 1 (Pedestrian Flow Network)**.** *We define a pedestrian flow network as an undirected graph G = (V, E) and treat sensors as graph nodes, where V is a finite set of* $|V| = N$ *nodes and E is a set of* $|E| = M$ *edges. N and M represent the number of nodes and edges in the graph topology structure.* $A \in \mathbb{R}^{N \times N}$ *is the adjacency matrix of the link set accommodating the structural information of graph G. The variable* $A(l_i, l_j)$ *denotes whether link i and link j are connected, i.e.,* $A(l_i, l_j) = 1$ *indicates* l_i *and* l_j *are connected or* $A(l_i, l_j) = 0$*, otherwise.*

Definition 2 (Signal Matrix)**.** *The observation on the graph G at the time step t is regarded as matrix* $X_t = \left(x_{t,1}, x_{t,2}, \cdots, x_{t,N}\right)^T \in \mathbb{R}^{N \times C}$*, where* $x_{t,v} \in \mathbb{R}^C$ *represents the feature vector of node v at time t and C is the number of features (a collection of variables, e.g., passenger flow, density, speed).*

Given a spatiotemporal series of historical pedestrian flow signal matrices $X \in \mathbb{R}^{T_h \times N \times C}$ over the past T_h time steps, the goal is to learn a mapping function f to predict the spatiotemporal series of future signal matrices $Y = \left(X_{t+1}, X_{t+2}, \cdots, X_{t+T_p}\right) \in \mathbb{R}^{T_p \times N \times C}$ over the next T_p time steps. To accurately align with the long-term historical data and capture repeated patterns in passenger flow, such as peak hours, we aggregate recent-term and long-term historical passenger flow signal matrices. We divide the historical sequence T_h into two time series segments of lengths T_r and T_l along the time axis as the input of the recent-term and long-term, respectively. As illustrated in Figure 1, the segment $X_r = (X_{t-T_r+1}, X_{t-T_r+2}, \cdots, X_t) \in \mathbb{R}^{T_r \times N \times C}$ represents the recent-term segment. Additionally, we select data from the past d days in extensive historical records that are closest to the recent-term segment and correspond to the same time slice as the long-term segment $X_l = \left(X_1, X_2, \cdots, X_{T_l}\right) \in \mathbb{R}^{T_l \times N \times C}$, where T_l is d times T_p. As a result, the passenger flow prediction problem can be formulated as:

$$[X_l, X_r; A] \overset{f_\theta(\cdot)}{\rightarrow} Y \tag{1}$$

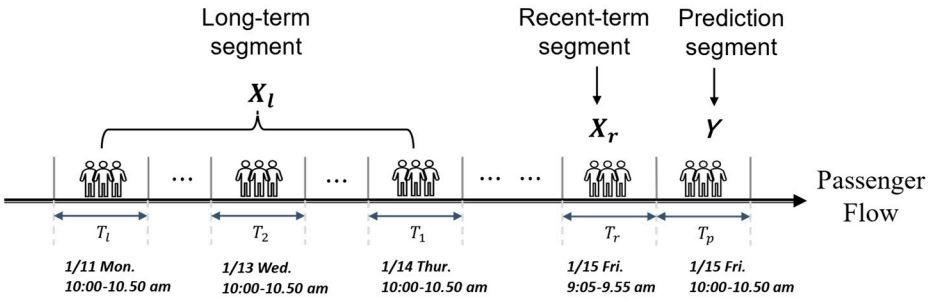


Figure 1. Input data of recent-term segment and long-term segment.

3.2. Overview

To address the multistep passenger flow prediction challenge, we propose a novel STGNN model to capture intricate spatiotemporal interactions in metro passenger flow prediction (MPFP) tasks. Figure 2 illustrates the overall pipeline of STGNN. The model mainly consists of four parts: long-term and recent-term spatial encoders, temporal encoders, and temporal decoders. The input of STGNN includes long-term passenger flow X_l, recent-term passenger flow X_r, and adjacent matrix A. X_l and X_r are separately input into the L-Spatial Encoder and R-Spatial Encoder to update the spatial features of nodes by dynamic global attention networks and graph convolution networks. The concatenated long-term and recent-term passenger flows are input into the temporal encoder of a stack of L layers to update the temporal features. The recent-term passenger flow is input into the temporal decoder of a stack of L' layers, after passing through a series decomposition block. The fusion gate accumulates the overall trend parts extracted from input data and hidden variables for prediction. Finally, the output of the temporal decoder is sent to the regression layer to obtain the future passenger flow.

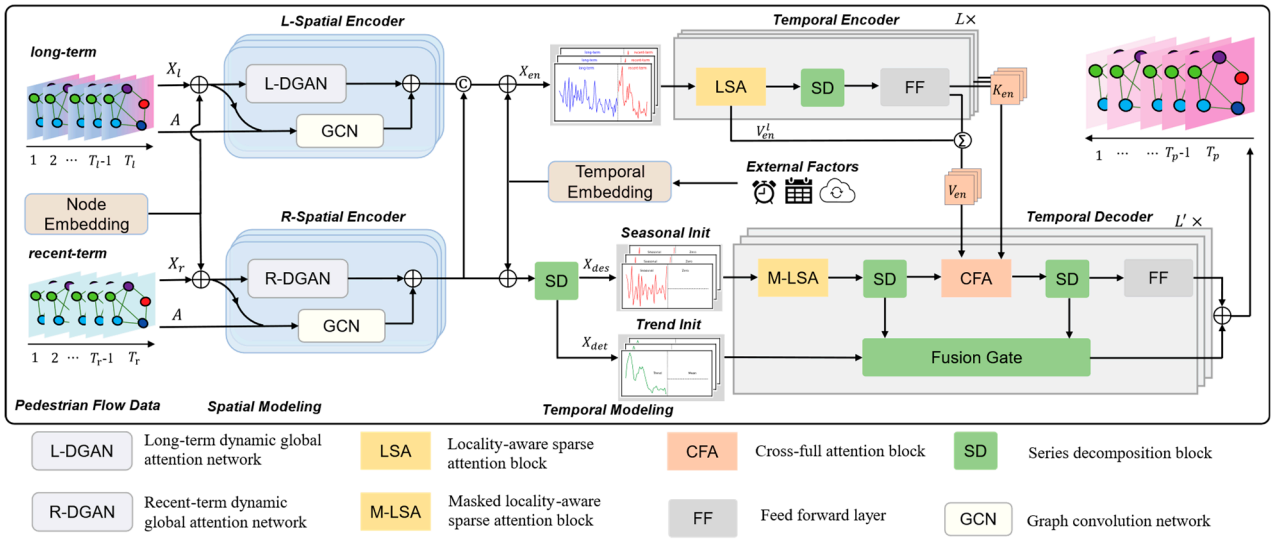


Figure 2. The architecture of STGNN. The architecture adopts the end-to-end trainable network for passenger flow prediction. The pipeline of STGNN is mainly divided into two parts: spatial modeling and temporal modeling. Spatial modeling is primarily composed of the L-Spatial Encoder and the R-Spatial Encoder, which update the spatial features of nodes through their respective DAGN and GCN. Temporal modeling is primarily composed of series decomposition blocks, a Temporal Encoder, and a Temporal Decoder. The Temporal Encoder primarily updates the seasonal features of the passenger flow sequence, while the Temporal Decoder updates these seasonal features and uses the trend component of the passenger flow sequence to assist in prediction. Node embedding and temporal embedding are respectively achieved through MLP, implementing the embedding of node features and time point features.

3.3. Spatial Modeling

In this section, we explain the architecture of the L-Spatial Encoder and the R-Spatial Encoder. GNNs are robust architectures well-suited for processing structured data, enabling the capture of complex and interrelated structural information. However, a reliance solely on GCN modules for capturing spatial features encounters limitations inherent to the characteristics of the input graph. Previous studies have highlighted the restricted expressive capacity of the message-passing paradigm. Although increasing the depth or breadth of GCNs can augment the receptive field, challenges such as optimization instability and information oversquashing persist [44,45].

To address these constraints, the global attention mechanism is embraced to extract spatial features not directly correlated with graph structure, thereby mitigating some of the limitations of GCNs. Additionally, it can be interpreted as a global message-passing mechanism facilitating the computation of pair-wise interactions between arbitrary node pairs. Consequently, the fusion of global attention and GCNs facilitates a comprehensive exploration of spatial features and captures implicit dependencies, such as sudden fluctuations in passenger flow at specific nodes or latent connections and long-range interactions between arbitrary node pairs. The resulting expression is depicted as follows:

$$X = (1-\alpha)\mathrm{DGAN}\left(X^{(0)}\right) + \alpha\mathrm{GCN}\left(X^{(0)}, A\right) \tag{2}$$

where α is a weight hyper-parameter, and A represents the adjacency matrix.

3.3.1. Dynamic Global Attention Network

The correlation among nodes will likely change over time. The idea is to use a dynamic spatial attention mechanism to adaptively capture the dynamic spatial correlation strengths

among nodes in the spatial dimension. The dynamic global attention is calculated as follows:

$$S = V_s \cdot \sigma\left(\left(X^{(0)} W_1\right) \cdot W_2 \cdot \left(W_3 X^{(0)}\right)^T + b_s\right) \tag{3}$$

$$\mathrm{DGAN}\left(X^{(0)}\right) = \mathrm{MatMul}\left(\mathrm{softmax}(S), X^{(0)}\right) + X^{(0)} \tag{4}$$

where $X^{(0)} = (X_1, X_2, \cdots, X_T) \in \mathbb{R}^{T \times N \times C}$,$S \in \mathbb{R}^{T \times N \times N}$, T is the length of the temporal dimension, C is the number of features, $V_s, b_s \in \mathbb{R}^{N \times N}$, $W_1, W_2 \in \mathbb{R}^{C \times C}$, and $W_3 \in \mathbb{R}^{N \times N}$ are learnable parameters, and σ is the activation function. A softmax function is used to dynamically compute the attention matrix. In Equation (3), the attention matrix adjusting the correlation between nodes is to capture the global influence from other nodes over time, while the residual connection preserves the information of the centered nodes.

3.3.2. Incorporation of Structural Information

In spectral graph analysis, studying the structural information of the graph usually refers to analyzing the corresponding Laplacian matrix and its eigenvalues. D denotes the diagonal degree matrix, and the Laplacian matrix of a graph and its normalization are defined as $L = D - A$ and $L^{sys} = D^{1/2} L D^{1/2} = I_N - D^{1/2} A D^{1/2}$, where $D_{ii} = \sum_j^n D_{ij}$, and I_N is the identity matrix. The eigen decomposition of normalized Laplacian is $L^{sys} = U \Lambda U^T$, where U is the Fourier basis, and $\Lambda = diag[\lambda_1, \lambda_2, \cdots, \lambda_n]$ is the diagonal matrix of eigenvalues. The GCN model constructs a filter to implement graph convolution operations in the domain of the Laplacian spectrum. The polynomial filtering operation for graph convolution can be formulated as:

$$g_\theta * x = g_\theta(L)x = U g_\theta(\Lambda) U^T x \tag{5}$$

where $g * x$ denotes a graph convolution and $g_\theta(\Lambda)$ denotes the spectral filter. The θ denotes the polynomial filter coefficients, and $g_\theta(\lambda) = \sum_{i=0}^{K} \theta_i \lambda^k$, $\lambda \in [0, 2]$. However, direct eigenvalue decomposition on a larger-scale Laplacian matrix is computationally expensive. The ChebNetII [46] model, a type of graph convolution neural network, is adopted here to incorporate the prior structural information of the input graph. The graph convolution therefore can be formulated as:

$$\mathrm{GCN}\left(X^{(0)}, A\right) = \frac{2}{K+1} \sum_{k=0}^{K} \sum_{j=0}^{K} w_j T_k(\ddagger_j) T_k(\hat{L}) f_\theta\left(X^{(0)}\right) \tag{6}$$

where $\hat{L} = 2L^{sys}/\lambda_{max} - I_N$, λ_{max} is the largest eigenvalue of L^{sys}, K is a hyper-parameter, the Chebyshev nodes $\ddagger_j$ for $T_k(\cdot)$ is defined as $\cos((j + 1/2)\pi/(K+1))$, $w_j, j \in \{1, \cdots, K\}$ is the learnable parameters, and f_θ is an MLP. The recursive definition of Chebyshev polynomial can be expressed as $T_k(\ddagger) = 2x T_{k-1}(\ddagger) - T_{k-2}(\ddagger)$, with $T_0(\ddagger) = 1$ and $T_1(\ddagger) = x$.

3.4. Temporal Modeling

Metro passengers exhibit considerable stability in their travel behaviors, particularly concerning temporal characteristics, encompassing recent metro passenger flow trends, daily cyclic patterns, and weekly fluctuations. Building on time series decomposition methods and the strong performance of Transformer architectures in handling long-term dependencies, our temporal modeling framework integrates Transformer blocks with seasonal-trend decomposition blocks [47]. This integration aims to effectively address the intricate periodic patterns in metro passenger flow data by capturing both cyclic fluctuations and trend components concurrently. Given a series of length L and dimension D, denoted as $X \in \mathbb{R}^{L \times D}$, the process of series decomposition can be succinctly articulated as:

$$X_{trend} = \mathrm{AvgPool}(\mathrm{Padding}(X)) \tag{7}$$

$$X_s = X - X_{trend} \tag{8}$$

where $X_{trend}, X_s \in \mathbb{R}^{L \times D}$, AvgPool$(\cdot)$ represents the average pooling with padding. The input of the temporal encoder X_{en} consists of the recent term X_{en}^r and the long term X_{en}^l.

$$X_{en} = \left[X_{en}^l, X_{en}^r\right] \tag{9}$$

The input of the temporal decoder consists of the seasonal component X_{des} and the trend component X_{det}. Each of them comprises two segments: the component decomposed from the recent term X_{en}^r to convey recent information, and the other contains placeholders with length T_p filled by a scalar value of zero and a mean of X_{en}^r. They can be expressed as:

$$X_{ens}, X_{ent} = \text{SeriesDecomp}(X_{en}^r) \tag{10}$$

$$X_{des} = [X_{ens}, X_0], X_{det} = [X_{ent}, X_{Mean}] \tag{11}$$

3.4.1. Temporal Encoder

The encoder adopts a multilayer structure. In the l-th encoder, the overall equations can be summarized as: $X_{en}^l = \text{Encoder}\ (X_{en}^{l-1})$, where $l \in \{1, \cdots, L\}$. The details can be formalized as:

$$S_{en}^{l,1}, _ = \text{SeriesDecomp}\left(\text{LSA}\left(X_{en}^{l-1}\right) + X_{en}^{l-1}\right) \tag{12}$$

$$S_{en}^{l,2} = \text{FeedForward}\left(S_{en}^{l,1}\right) + S_{en}^{l,1}, X_{en}^l = S_{en}^{l,2} \tag{13}$$

where $S_{en}^{l,i}$, $i \in \{1,2\}$ denotes the seasonal component after the i-th decomposition block in the l-th layer, respectively. LSA is a locality-aware sparse attention operation. The feed-forward consists of two linear transformations with a ReLU activation in between, as in:

$$\text{FeedForward}\left(S_{en}^{l,1}\right) = \text{ReLU}(W_{l,1} S_{en}^{l,1} + b_{l,1}) W_{l,2} + b_{l,2} \tag{14}$$

where $W_{l,i}$, $b_{l,i}$, $i \in \{1,2\}$, respectively represent the projector and bias in the i-th layer.

3.4.2. Temporal Decoder

The decoder also adopts a multilayer structure. In the l-th decoder, the overall equations can be summarized as: $X_{de}^l, \mathcal{T}_{de}^l = \text{Decoder}\ (X_{de}^{l-1}, \mathcal{T}_{de}^{l-1}, K_{en}, V_{en})$, where $l \in \{1, \cdots, L'\}$. The details can be formalized as:

$$K_{en} = X_{en}^L, \quad V_{en} = X_{en}^L + \sum_{i=1}^{L} V_{en}^i \tag{15}$$

$$S_{de}^{l,1}, \mathcal{T}_{de}^{l,1} = \text{SeriesDecomp}\left(\text{LSA}\left(X_{de}^{l-1}\right) + X_{de}^{l-1}\right) \tag{16}$$

$$S_{de}^{l,2}, \mathcal{T}_{de}^{l,2} = \text{SeriesDecomp}\left(\text{CFA}\left(S_{en}^{l,1}, K_{en}, V_{en}\right) + S_{en}^{l,1}\right) \tag{17}$$

$$S_{de}^{l,3} = \text{FeedForward}\left(S_{en}^{l,2}\right) + S_{en}^{l,2}, \quad X_{de}^l = S_{de}^{l,3} \tag{18}$$

$$\mathcal{T}_{de}^l = \mathcal{T}_{de}^{l-1} + W_{l,1} \mathcal{T}_{de}^{l,1} + W_{l,2} \mathcal{T}_{de}^{l,2} \tag{19}$$

where $S_{de}^{l,i}$, $\mathcal{T}_{de}^{l,j}$, $i \in \{1,2,3\}$,$j \in \{1,2\}$ are the seasonal and trend part component after the i-th decomposition block in the l-th layer. LSA is locality-aware sparse attention operation, and CFA is cross-full attention operation. $W_{l,j}$, $i \in \{1,2\}$ is the projector for the j-th trend component $\mathcal{T}_{de}^{l,j}$. The feed-forward also consists of two linear transformations with a ReLU activation in between. The final prediction output is the sum of the two refined decompositions after a linear transformation and can be expressed as:

$$Y = \text{Linear}\left(X_{de}^{L'} + \mathcal{T}_{de}^{L'}\right) \tag{20}$$

where Linear denotes a linear transformation.

3.5. Locality-Aware Sparse Attention

The multi-head self-attention mechanism is a specific implementation of the attention mechanism and a commonly adopted form due to its effectiveness in capturing long-term dependencies [29]. The basic operation in multi-head attention is scaled dot-product attention, defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{D}} * M\right)V \tag{21}$$

where $Q = K = V$, Q, K, V, and D are queries, keys, values, and their dimensions, respectively. The mask matrix M is an optional term to avoid future information leakage.

Metro passenger flow dynamics are also susceptible to perturbations induced by unforeseen events, such as extreme weather occurrences or holidays. Consequently, the variations in flow patterns at the time of observation are closely linked with the local contextual cues. Nevertheless, conventional self-attention mechanisms, tailored for discrete tokens, compute similarities between queries and keys solely based on their point-wise values, disregarding the local contextual nuances. This limitation challenges the effective application of such mechanisms to sequence transformations in passenger flow analysis, potentially causing query-key mismatches and optimization difficulties. [48] endeavored to mitigate this discrepancy by leveraging causal convolution [49]. However, the adequacy of extracting local context information through fixed window convolutions remains questionable. Concurrently, the computational complexity of self-attention is $O(L^2)$, making it infeasible to directly model long-term series. To alleviate these two problems, we design a locality-aware sparse attention block (LSA), as illustrated in Figure 3. In the Mixture of Causal Conv, a set of causal convolution kernels of different sizes are employed to extract and fuse multiple local context information, transforming the input into queries and keys, and combining them using a set of data-dependent weights. The details can be expressed as:

$$Q = \text{Softmax}(\text{L}(X)) * F_Q(X) \tag{22}$$

$$K = \text{Softmax}(\text{L}(X)) * F_K(X) \tag{23}$$

where F_Q, F_K, are two different sets of causal convolution kernel, and L is a fully connected layer. $\text{Softmax}(\text{L}(X))$ is the weight for combing. Then, the max-mean measurement [30] method is used as a select function to select top u dominant queries. The details can be expressed as:

$$\overline{Q} = \text{Select}(Q) \tag{24}$$

$$\text{LSAttention}\left(\overline{Q}, K, XW\right) = \oplus(\text{LShead}_1, \cdots, \text{LShead}_\text{h})W^O \tag{25}$$

$$\text{LShead}_\text{i} = \text{Attention}\left(\overline{Q}_i, K_i, XW_i\right) \tag{26}$$

where Select () is the operation of max-mean measurement, and Attention is the operation of scaled dot-product attention, and W is the projector.

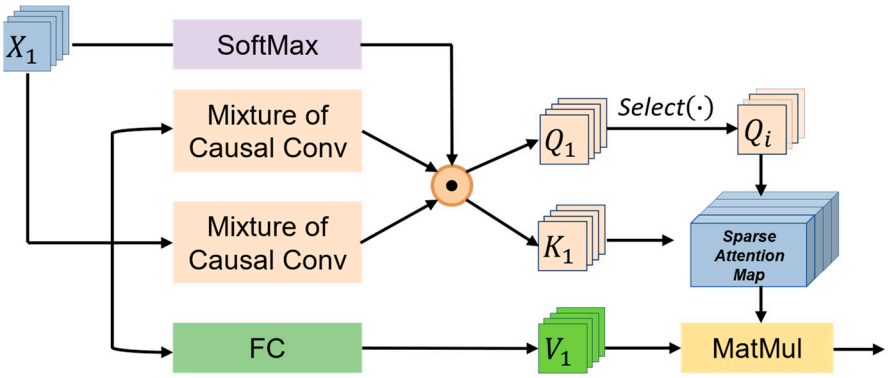


Figure 3. The architecture of locality-aware sparse attention. The input initially passes through a fully connected layer and Softmax to obtain values and a set of data-dependent weights. Subsequently, the input proceeds through two sets of convolutional layers with different kernel sizes (e.g., 1, 3). The results are then weighted by the obtained weights to generate queries and keys. Finally, the select function chooses the top u dominant queries to complete the attention operation with keys and values.

3.6. Embedding

The enduring constancy of static node features underscores the need to incorporate spatial heterogeneity while preserving the structural integrity of graphs. To address this, we introduce an augmented node embedding matrix, denoted as $\mathrm{NE} \in \mathbb{R}^{N \times C}$, where each node is associated with an additional embedding vector. We also leverage fixed position embeddings, denoted as $\mathrm{PE}(t, d)$, to provide the model with order-awareness, where t signifies the position of the input element and d denotes the vector dimension. Furthermore, to capture hierarchical temporal granularities (e.g., minutes, hours, and weeks), we integrate learnable temporal stamp embeddings (e.g., MLP), denoted as $\mathrm{SE}(t)$, corresponding to each hierarchical level. The amalgamation of these components yields the final temporal position embedding $\mathrm{TE}(t)$ for an input element positioned at t, encapsulated by the expression:

$$\mathrm{TE}(t) = \mathrm{PE}(t,) + \mathrm{SE}(t) \tag{27}$$

$$\mathrm{PE}(t, 2d) = \sin\left(t/1000^{2d/D}\right) \tag{28}$$

$$\mathrm{PE}(t, 2d+1) = \cos\left(t/1000^{2d/D}\right) \tag{29}$$

where t is the relative index of each symbol in the sequence and $1 \leq d \leq D$.

3.7. Loss Function

In the training stage, to minimize the error between the ground truth and the predicted value, the loss function is defined as:

$$\mathcal{L}(Y, \hat{Y}) = \frac{1}{|T| \times |\mathcal{V}|} \sum_{\tau \in T} \sum_{v \in \mathcal{V}} \left\| \mathcal{Y}^{(\tau, v)} - \hat{\mathcal{Y}}^{(\tau, v)} \right\| + \lambda L_{reg} \tag{30}$$

where the first term is used to calculate loss and the L_{reg} is a regularization term; λ is a hyperparameter; Y and $\hat{Y}$ are denoted as the ground truth and the predicted value. All model parameters are updated by minimizing the loss function through the mini-batch gradient descent algorithm in the training process.

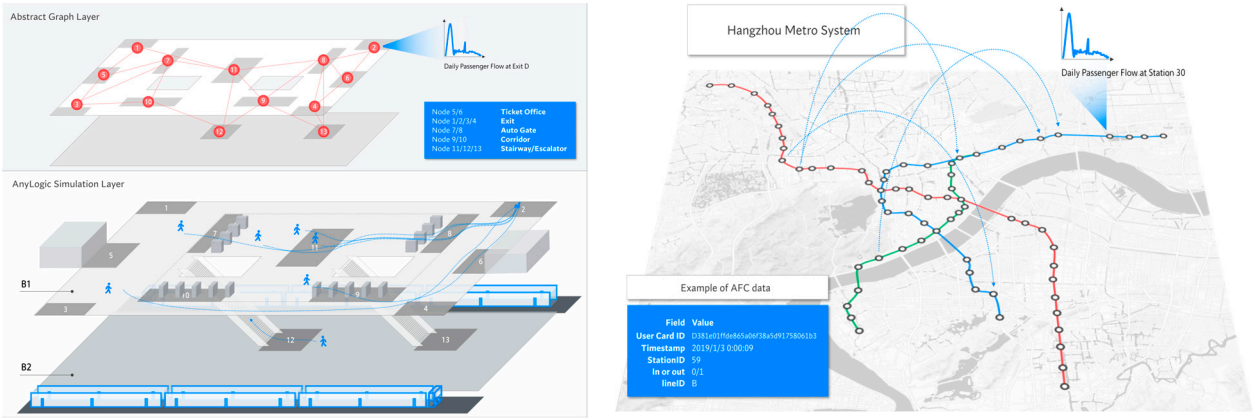
4. Experiments

4.1. Data Preparation

4.1.1. AnyLogic Simulation in Metro Station

In accordance with the architectural 3D CAD model, we identified 13 Origin-Destination (OD) regions within the metro station. As delineated by circular markers in Figure 4a, we

first identified 13 OD regions within the two-level metro station, each of which corresponds to pertinent functional facilities (such as escalators, exits, and security checkpoints) or commercial facilities (including ticket and vending machines), and these regions are areas where passenger flow converges and disperses. This deliberate selection serves to abstract the intricate movements of passengers within the metro station, conceptualizing their transit between these 13 OD pairs, thereby constituting a graph with 13 nodes. As shown in Figure 4, if there is an edge between any two nodes, it indicates a transition of passenger flow between these two nodes. In AnyLogic-professional-8.7.11 simulation, sensors positioned within the aforementioned 13 OD regions meticulously captured variations in passenger flows. These observations encompassed movements within functional facilities as well as within commercial facilities. Data collection was conducted comprehensively from the operational hours of 6:00 a.m. to 11:00 p.m., facilitating an extensive examination of passenger dynamics within the station.



(**a**) (**b**)

Figure 4. Simulation layout in AnyLogic for metro station and Hangzhou Metro System. The red lines represent the edges connecting the nodes, while the blue dashed lines indicate an example of passenger flow direction. (**a**) Anylogic for Metro Station; (**b**) Hangzhou Metro System.

4.1.2. Hangzhou Metro System

Each piece of the AFC record in Hangzhou data that will be used in the in-flow and out-flow data generation process is summarized in Figure 4b. The data is divided into specific periods according to the AFC field to aggregate inflow and outflow. Then, we set a time interval window to count the number of inbounds and outbounds of each metro station to obtain the MSP inflow and the MSP outflow.

In this study, two distinct passenger flow datasets are used, whose detailed characteristics are outlined in Table 1. The first dataset comprises simulated data obtained from 13 sensors deployed within an AnyLogic simulated metro station. This dataset encompasses real-time passenger flow recordings aggregated approximately every 5 min over 21 days, from 06:00 to 23:00, about eighty-seven thousand data points. The second dataset is from the Hangzhou metro system (https://tianchi.aliyun.com/competition/entrance/231708/information (accessed on 30 October 2023)), comprising 81 metro stations and an aggregate of 70 million data points collected from three metro lines in Hangzhou, China, spanning from 1 January to 26 January 2019. To facilitate passenger flow prediction, the raw data has been processed, aggregating inflow and outflow information into discrete 5-min intervals, thereby enhancing temporal granularity, from 06:00 to 23:30.

Table 1. Dataset description.

Datasets	Nodes	Time Span	Daily Range
Ours	13	1 January–21 January	6:00–23:00
Hangzhou	81	1 January 2019–26 January 2019	6:00–23:30

4.2. Setting

All datasets are split at a ratio of 7:1:2 into training sets, validation sets, and test sets by the time and normalized all data into the range $[-1, 1]$ with the Min-Max method. The STGNN model uses the PyTorch framework and NVIDIA GPU (GeForce RTX 3090). We train our model using Adam optimizer [50] with a learning rate decay strategy. We set an early stop of 10 to save the best model as the model checkpoint. The initial learning rate and batch size are set as 0.001 and 32, respectively. The model dimensions C and D are 512, and the number of attention heads h is 8. For our dataset, L and L' are 2 and 1, respectively. For the Hangzhou dataset, L and L' are 3 and 1, respectively. For evaluation, we re-transform the predicted values back to the ground truth and employ two widely adopted metrics, namely Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), to evaluate the prediction performance of our model and baseline models.

4.3. Baseline Methods

The proposed STGNN model is compared with two types of methods: non-graph methods and graph-based methods. For the non-graph methods, the chosen baseline models are (1) LSTM [51]: Long Short-Term Memory network, a special RNN model. (2) Reformer [31]: An effective transformer replaces dot-product attention with one that uses locality-sensitive hashing for long sequence modeling. (3) Autoformer [47]: A novel decomposition architecture uses an auto-correlation mechanism based on the series periodicity to conduct the dependencies discovery and representation aggregation at the sub-series level. (4) TimesNet [52]: A temporal 2D variation model extends the analysis of temporal variations into 2D space by transforming the 1D time series into a set of 2D tensors based on multiple periods.

As for the graph-based methods, the chosen baseline models are (5) GCN [53]: A graph convolution network is used to capture spatial dependence. (6) STG-NODE [54]: A spatial-temporal graph network based on a neural-controlled differential equation extends the concept of a neural-controlled differential equation and designs two neural-controlled differential equations for temporal processing and spatial processing. (7) T-GCN [25]: A temporal graph convolution network combines graph convolution and gated recurrent units to capture spatial dependence and temporal dependence. (8) ASTGCN [41]: a spatial-temporal convolutional network based on an attention mechanism uses temporal and spatial attention to capture spatial-temporal dependencies. (9) ASTGNN [55]: a spatial-temporal graph network is based on attention mechanisms, graph convolution networks, learning dynamics, and the heterogeneity of spatial-temporal graph data.

4.4. Performance Evaluation

4.4.1. Comparison with Baseline Methods

The proposed STGNN model is compared with various baseline methods for predicting 10, 20, 30, and 60 steps on a simulated metro flow dataset. The results in Table 2 substantiate the superior efficacy of the proposed STGNN model across all evaluated metrics.

Table 2. Average performance comparison for our dataset.

Models	Prediction Step = 10		Prediction Step = 20		Prediction Step = 30		Prediction Step = 60	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LSTM	6.40	3.68	6.65	3.81	7.43	4.10	9.62	5.39
Reformer	6.80	3.25	7.38	3.43	8.68	3.98	9.27	4.56
Autoformer	6.45	2.85	6.64	2.98	7.55	3.58	9.60	4.77
TimesNet	5.41	2.38	6.02	2.61	6.46	2.98	7.41	3.39
GCN	9.33	5.04	10.11	6.02	10.59	6.18	11.17	7.55
T-GCN	5.85	2.76	6.54	3.14	7.30	3.35	8.72	4.22
STG-NCDE	7.39	3.94	8.30	3.89	9.93	4.66	10.81	5.89
ASTGCN	5.14	2.19	5.81	2.61	6.92	2.69	7.71	3.42
ASTGNN	4.89	2.01	5.55	2.46	6.71	2.47	7.58	3.29
STGNN (ours)	4.58	1.91	5.07	2.23	5.67	2.39	7.17	3.08

Among the non-graph methods, LSTM exhibits the least favorable performance, primarily attributed to its inability to effectively capture the recurrent and daily patterns inherent in urban commuter behavior, as well as the broader fluctuations within metro flow dynamics. Autoformer, while slightly outperforming LSTM, struggles to capture the subtle long-term patterns in metro flow variations, indicating limitations in its autocorrelation mechanism. Conversely, the Reformer leveraging attention mechanism, demonstrates superior performance over other non-graph methods in shorter prediction intervals (i.e., 10 and 20 steps). Notably, TimesNet, which scrutinizes complex temporal metro flow patterns from a two-dimensional space, excels over non-graph methods in longer prediction horizons (i.e., 30 and 60 steps).

Turning to the graph-based methods, GCN fares relatively poorly owing to its exclusive focus on static spatial features, overlooking the temporal evolution of metro flow distribution. In contrast, T-GCN incorporates temporal dynamics and employs gated recurrent units (GRUs) to capture temporal dependencies, thereby exhibiting notable enhancements in short-term prediction accuracy. STG-NCDE, integrating spatiotemporal controlled differential equations, demonstrates commendable performance in analyzing traffic flow data, albeit its effectiveness wanes in capturing prolonged temporal dependencies within the metro flow. ASTGCN and ASTGNN excel in capturing spatial-temporal dependencies, with ASTGNN surpassing all methods in short-term predictions (10 and 20 steps). Notably, STGNN outperforms ASTGNN in 30 and 60 steps, underscoring its capability to enhance long-term prediction accuracy. Specifically, STGNN exhibits improvements over ASTGNN by 15.4% and 5.4% in terms of RMSE and by 0.6% and 0.3% in MAE for intervals of 30 and 60 steps, respectively.

Tables 3 and 4 illustrate the average prediction performance over 10, 20, 30, and 60 steps ahead on the Hangzhou metro flow dataset. The findings consistently affirm the superior performance of the STGNN model across all evaluated metrics. Notably, among the non-graph methods, Reformer outperforms TimesNet, suggesting that the efficacy of analyzing complex spatial-temporal patterns within metro flow data in a two-dimensional space diminishes with an expanded node count.

Table 3. Average performance comparison over 10 and 20 steps for the Hangzhou dataset.

Models	Prediction Step = 10				Prediction Step = 20			
	Inflow		Outflow		Inflow		Outflow	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LSTM	29.13	19.15	38.93	23.74	32.92	21.29	41.42	25.17
Reformer	22.47	13.27	30.83	18.01	24.43	14.01	31.29	18.47
Autoformer	24.59	15.41	32.65	20.68	32.44	19.95	39.44	24.30
TimesNet	23.94	14.03	32.10	17.95	25.36	14.91	33.81	18.91
GCN	67.82	38.51	74.44	41.10	68.60	39.80	75.22	42.91
T-GCN	27.96	17.08	35.94	22.79	30.53	18.45	38.62	24.04
STG-NCDE	24.20	13.57	31.61	18.35	30.93	16.86	42.21	23.37
ASTGCN	23.01	14.36	32.99	19.86	25.03	14.92	34.33	21.88
ASTGNN	20.61	12.30	28.63	17.12	22.36	13.44	29.54	17.86
STGNN (ours)	18.90	11.58	27.02	16.90	20.10	12.01	27.51	17.05

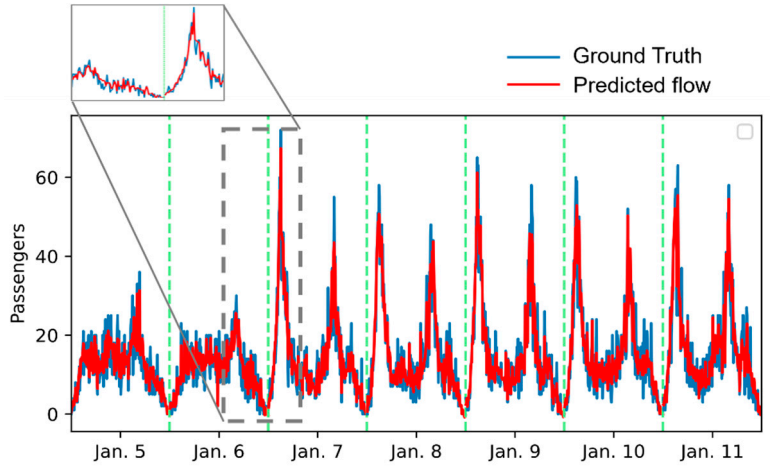
Table 4. Average performance comparison over 30 and 60 steps for the Hangzhou dataset.

Models	Prediction Step = 30				Prediction Step = 60			
	Inflow		Outflow		Inflow		Outflow	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LSTM	34.41	22.38	43.47	26.94	37.20	24.25	46.42	29.39
Reformer	25.41	14.40	31.83	18.83	27.26	15.86	34.42	20.55
Autoformer	38.75	22.79	41.56	25.59	54.38	29.14	57.25	32.14
TimesNet	26.47	15.18	34.95	19.34	28.62	16.73	36.33	20.66
GCN	69.20	40.50	76.46	43.34	73.83	44.84	80.11	46.76
T-GCN	32.96	20.08	40.79	25.99	35.33	23.45	44.32	28.34
STG-NCDE	38.95	20.45	51.39	27.40	41.81	22.03	54.53	30.10
ASTGCN	30.34	17.99	36.80	22.10	32.87	19.96	39.33	23.28
ASTGNN	23.97	13.92	30.43	18.32	25.59	14.95	32.92	19.96
STGNN (ours)	21.44	12.65	27.81	17.12	23.40	13.78	28.93	18.35

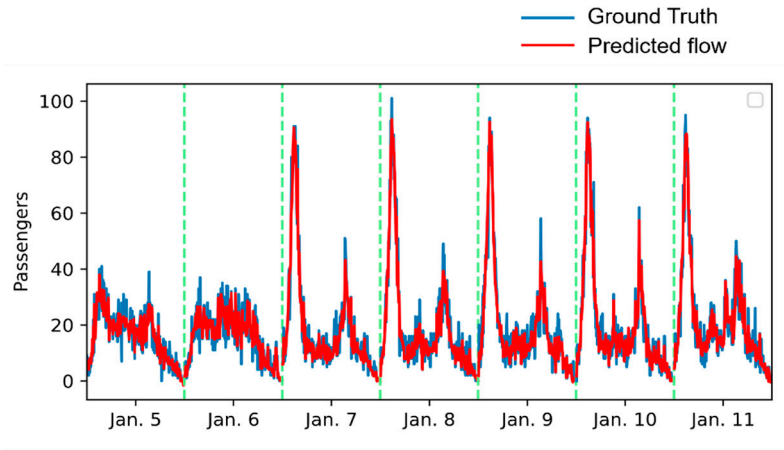
Regarding the graph-based methods, ASTGNN maintains its dominance over all other baseline methods. Specifically, concerning inflow and outflow predictions in the Hangzhou dataset, our STGNN model demonstrates notable performance over ASTGNN. Precisely, in the context of 30 step predictions, STGNN outperforms ASTGNN by 10.5% and 8.6% in terms of RMSE and by 9.1% and 6.5% in terms of MAE for inflow and outflow, respectively. Moreover, for 60-step predictions, STGNN exhibits enhancements over ASTGNN by 8.5% and 12.1% in terms of RMSE and by 7.8% and 8.0% in terms of MAE for inflow and outflow, respectively.

4.4.2. Comparison with Ground Truth

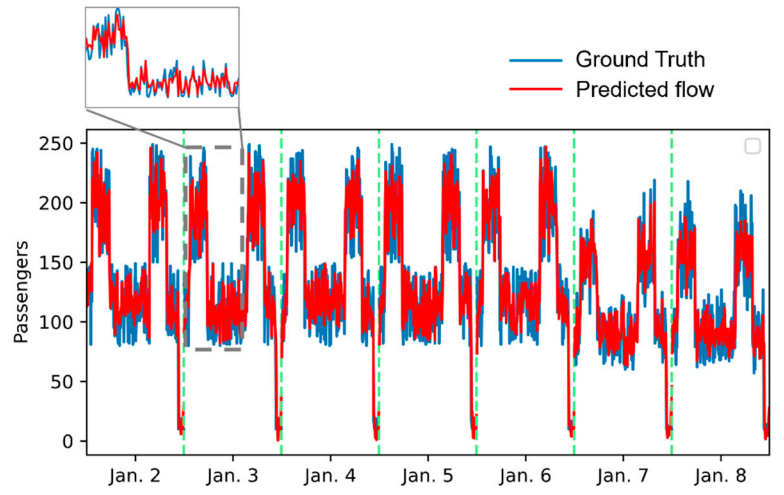
In Figure 5, we visualize the comparison in 10 steps predicted by STGNN and the ground truth values of two representative metro stations for the Hangzhou dataset and one node for the simulation dataset. From the figures, we can observe that the passenger flow predicted by STGNN is close to the ground truth values.



(a) Yongjiang Road Station



(b) Fuxing Road Station



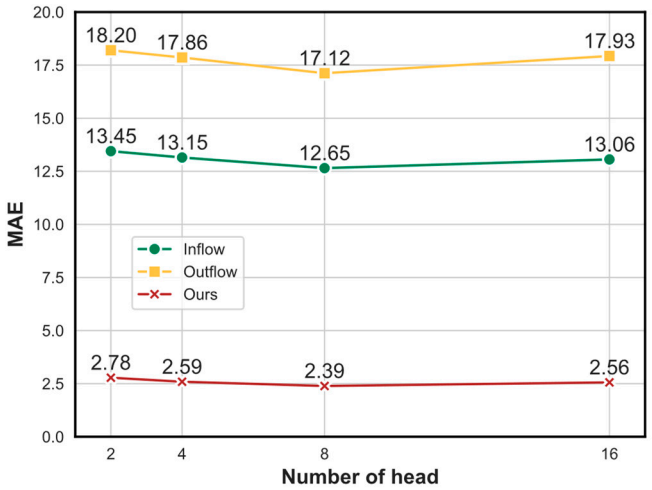
(c) Node 3 in Anylogic for Metro Station

Figure 5. Comparison with Ground Truth.

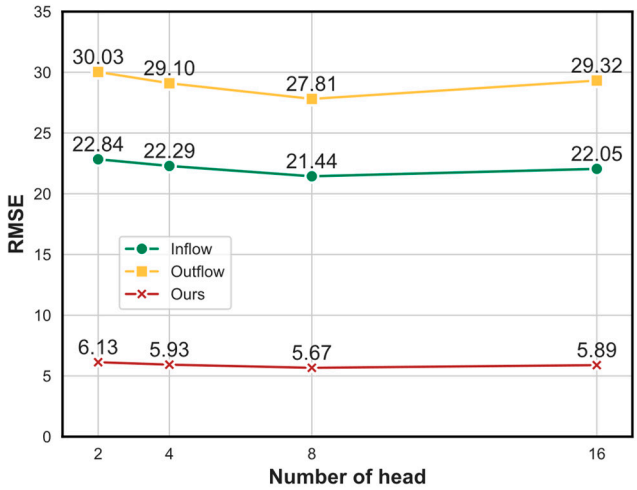
4.5. Effect of Different Network Configurations

A series of experiments are conducted to explore the impact of key hyper-parameter settings on model performance, employing different network configurations. All models adhere to uniform settings except for the studied variable. Figure 6 illustrates the results obtained with varying model dimensions and attention head numbers across all datasets,

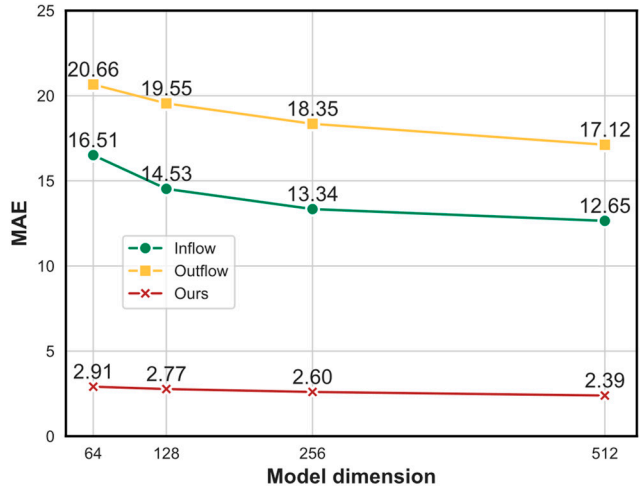
with the prediction step set to 30. In comparison to attention heads, the STGNN model is more sensitive to changes in dimensionality. Across all datasets, an augmentation in model dimensions correlates with a discernible reduction in RMSE and MAE. Notably, this trend is particularly pronounced in the Hangzhou dataset, where more substantial improvements are discernible. In contrast, it is noteworthy that an increase in the number of attention heads does not consistently yield improved prediction performance. Optimal results are attained when employing eight attention heads.



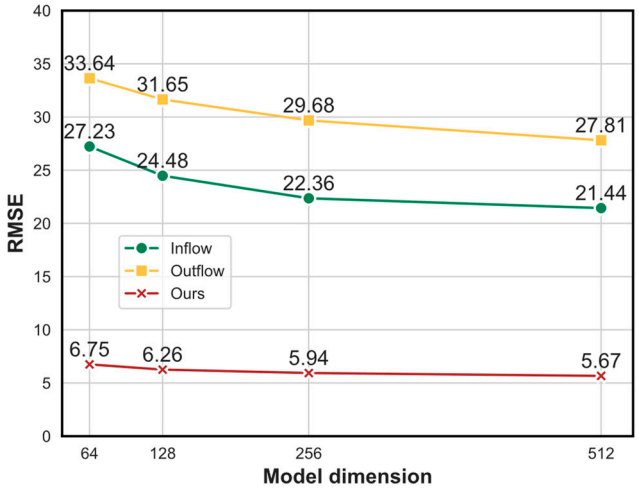
(a) The impact of number of head on MAE



(b) The impact of number of head on RMSE



(c) The impact of model dimension on MAE



(d) The impact of model dimension on RMSE

Figure 6. Network configuration analysis. These four figures demonstrate the impact of different model dimensions and attention head numbers on RMSE and MAE, respectively.

4.6. Ablation Experiments

To further analyze the effectiveness of the key components in STGNN, we conduct ablation experiments to evaluate three variants of our model on two datasets:

STGNN-noSM: It removes the total spatial modeling module to investigate the benefits of modeling the spatial dependencies of the passenger flow network.

STGNN-noDAG: It removes the dynamic global attention network to investigate the utility of dynamic spatial correlation strength impact between global node pairs rather than relying solely on static topological relationships.

STGNN-noLSA: It replaces the locality-aware sparse attention block with a traditional attention block to investigate the usefulness of considering various contextual information.

Except for the variances, all the variant models share identical settings. Figure 7 shows the performance of the three variants on our simulation dataset and the Hangzhou dataset. We can easily observe that the changes in RMSE and MAE are almost consistent across different datasets. By comparing STGNN-noSM and STGNN, no doubt that modeling the spatial dependences plays a positive role in multi-step metro station passenger flow prediction and verifies the effectiveness of the spatial modeling module. STGNN-noDAG, which only uses GCN for spatial modeling, yields inferior performance compared to STGNN but is slightly better than STGNN-noSM. This result indicates the significance of the dynamic global spatial relationship between global node pairs. Moreover, the superior performance of STGNN compared to STGNN-noLSA demonstrates the effectiveness of modeling local trends by employing multi-head self-attention for long-term passenger flow forecasting.

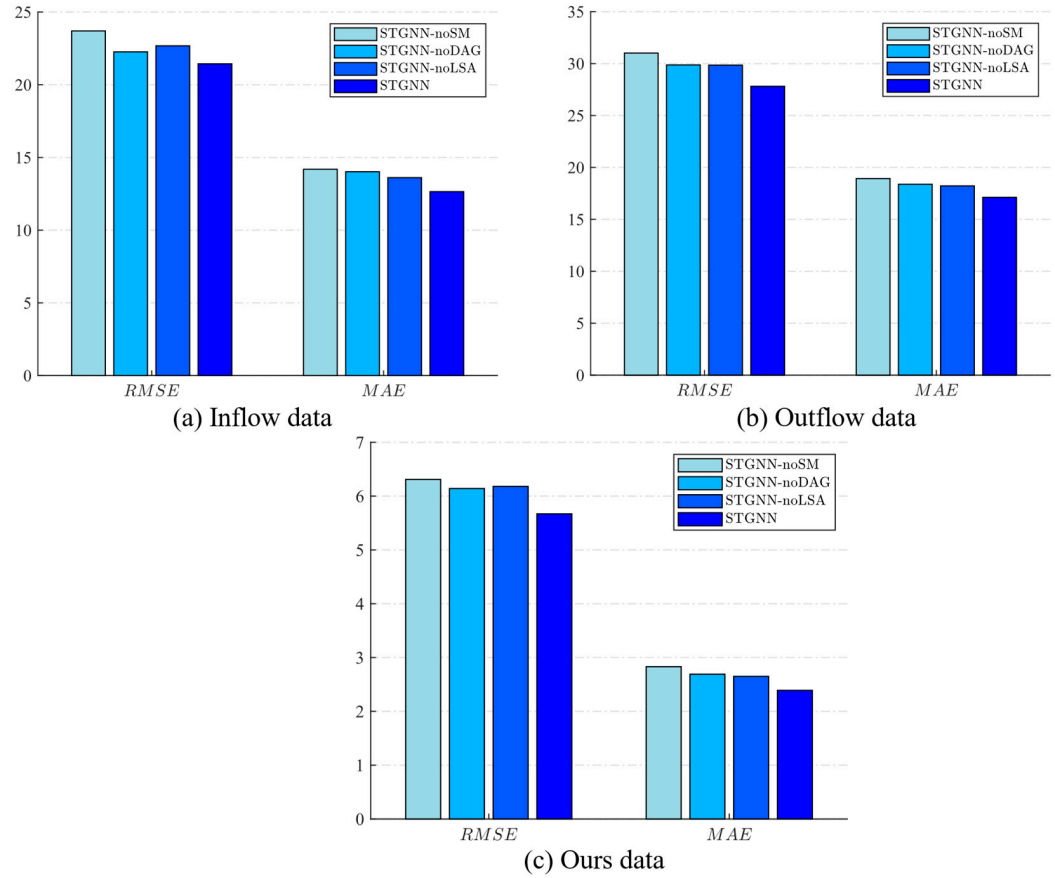


Figure 7. Components analysis. These three figures demonstrate the impact of different model components on RMSE and MAE, respectively.

4.7. Computation Cost Study

For the ASTGCN, ASTGNN, and STGNN models, the computational complexity of attention operation is O($N^2 \cdot C$) in the spatial dimension. In the temporal dimension, the attention operation complexity for ASTGCN and ASTGNN is O($L^2 \cdot D$), while for STGNN, it is O($L \log L \cdot D$). Table 5 shows the recorded inference time for one epoch during the training stage for the four spatial-temporal models in two different prediction steps. STG-NODE has a relatively long computation time because it uses ordinary differential equations for

spatial-temporal modeling. STGNN utilizes sparse attention and a generative decoder, resulting in significantly faster inference time than ASTGCN and ASTGNN.

Table 5. The computation time for the training epoch.

Models	Computation Time for Training Epoch (s)	
	Prediction Step = 30	Prediction Step = 60
STG-NODE	119	209
ASTGCN	11.49	27.81
ASTGNN	18.33	33.31
STGNN	8.59	13.28

5. Discussion and Conclusions

This study presents a finely resolved subway station model using Anylogic passenger simulation. Additionally, we propose a novel spatial-temporal graph neural network named STGNN tailored for passenger flow prediction tasks in metro stations. Based on this, new modules for spatial and temporal dimensions are redesigned to capture respective spatial and temporal dependencies. Specifically, we devise a dynamic global attention network to discern and adaptively capture the dynamic global influence along the spatial dimension, thereby facilitating the exploration of passenger flow fluctuation between global node pairs. The GCN integrated into the spatial modeling module incorporates pertinent prior information derived from the input graph structure. Furthermore, we introduce specialized modules such as the series decomposition block and locality-aware sparse attention block designed for time series prediction. The series decomposition block decomposes the passenger flow series into seasonal and trend-cyclical components, helping the model more accurately capture the seasonality and progression of the passenger flow series. The locality-aware sparse attention block aims to extract multiple local contexts of information and alleviate the computational complexity associated with long sequence modeling, contributing to more accurate passenger flow predictions and reduced inference time. Through comparative experiments and ablation experiments on the simulation dataset and Hangzhou dataset, we demonstrate the effectiveness and robustness of our proposed approach.

Nevertheless, the STGNN model has limitations: (1) The intricate patterns of fluctuation and the spatial-temporal relationships inherent in metro passenger flow remain subject to the influence of numerous unknown external events, such as weather conditions. Our model can only implicitly learn the impact of these unknown external events on passenger flow evolutions and cannot accurately assess the influence of any specific external factor. (2) Our model analyzes passenger flow variations by aggregating long-term and recent-term passenger flow segments, where the long-term segments only include daily-period passenger flow while ignoring weekly-period passenger flow variations. In future work, we will enhance the applicability and generalization capability of STGNN from the following aspects: (1) We plan to adjust the components of the STGNN to generalize it to other prediction tasks. (2) We plan to construct the learning of spatial correlations of passenger flow between metro stations from a multi-graph perspective. (3) Future research could incorporate multi-source data, such as weather forecasts, holidays, and metro schedules, to enhance the model's robustness. These will help optimize metro schedules, enhance passenger experience, and support data-driven decision-making in infrastructure development.

Author Contributions: Y.C.: Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Writing—original draft, Writing—review and editing. M.Z.: Formal analysis, Any logic Software, Writing—review and editing. Y.D.: Methodology, Writing—review and editing. K.W.: Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Writing—original draft, Funding acquisition, Supervision, Writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by grants from the National Key R&D Program of China (2022YFC3801300) and a cooperative project with CCTEG China Coal Mining Research Institute (TDKC-DZ-2022-016).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wei, Y.; Chen, M.C. Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks. *Transp. Res. Part C Emerg. Technol.* **2012**, *21*, 148–162. [CrossRef]
2. Wu, Y.; Hernández-Lobato, J.M.; Zoubin, G. Dynamic covariance models for multivariate financial time series. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 558–566.
3. Mudelsee, M. *Climate Time Series Analysis*; Atmospheric and Oceanographic Sciences Library; Springer: Cham, Switzerland, 2010; 397p.
4. Dai, X.; Fu, R.; Lin, Y.; Li, L.; Wang, F. Y Deeptrend: A deep hierarchical neural network for traffic flow prediction. *arXiv* **2017**, arXiv:1707.03213.
5. Van Lint, J.W.C.; Van Hinsbergen, C.P.I.J. Short-term traffic and travel time prediction models. *Artif. Intell. Appl. Crit. Transp. Issues* **2012**, *22*, 22–41.
6. Ahmed, M.S.; Cook, A. *R Analysis of Freeway Traffic Time-Series Data by Using Box-Jenkins Techniques*; Transportation Research Board, National Academies of Sciences, Engineering, and Medicine: Washington, DC, USA, 1979; No. 722.
7. Milenković, M.; Švadlenka, L.; Melichar, V.; Bojović, N.; Avramović, Z. SARIMA modelling approach for railway passenger flow forecasting. *Transport* **2018**, *33*, 1113–1120. [CrossRef]
8. Chen, E.; Ye, Z.; Wang, C.; Xu, M. Subway passenger flow prediction for special events using smart card data. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 1109–1120. [CrossRef]
9. Ariyo, A.A.; Adewumi, A.O.; Ayo, C.K. Stock price prediction using the ARIMA model. In Proceedings of the 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, Cambridge, UK, 26–28 March 2014; pp. 106–112.
10. Williams, B.M.; Hoel, L.A. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *J. Transp. Eng.* **2003**, *129*, 664–672. [CrossRef]
11. Bengio, Y.; LeCun, Y. Scaling learning algorithms towards AI. *Large Scale Kernel Mach.* **2007**, *34*, 1–41.
12. Ahmed, N.K.; Atiya, A.F.; Gayar, N.E.; El-Shishiny, H. An empirical comparison of machine learning models for time series forecasting. *Econom. Rev.* **2010**, *29*, 594–621. [CrossRef]
13. Martínez, F.; Frías, M.P.; Pérez, M.D.; Rivera, A.J. A methodology for applying k-nearest neighbor to time series forecasting. *Artif. Intell. Rev.* **2019**, *52*, 2019–2037. [CrossRef]
14. Wu, C.H.; Ho, J.M.; Lee, D.T. Travel-time prediction with support vector regression. *IEEE Trans. Intell. Transp. Syst.* **2004**, *5*, 276–281. [CrossRef]
15. Vlahogianni, E.I.; Golias, J.C.; Karlaftis, M.G. Short-term traffic forecasting: Overview of objectives and methods. *Transp. Rev.* **2004**, *24*, 533–557. [CrossRef]
16. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
17. Guo, J.; He, H.; He, T.; Lausen, L.; Li, M.; Lin, H.; Shi, X.; Wang, C.; Xie, J.; Sheng Zha, S.; et al. Gluoncv and gluonnlp: Deep learning in computer vision and natural language processing. *J. Mach. Learn. Res.* **2020**, *21*, 845–851.
18. Zheng, W.; Lee, D.H.; Shi, Q. Short-term freeway traffic flow prediction: Bayesian combined neural network approach. *J. Transp. Eng.* **2006**, *132*, 114–121. [CrossRef]
19. Xiao, H.; Sun, H.; Ran, B.; Oh, Y. Fuzzy-neural network traffic prediction framework with wavelet decomposition. *Transp. Res. Rec.* **2003**, *1836*, 16–20. [CrossRef]
20. Wang, L.; Zeng, Y.; Chen, T. Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. *Expert Syst. Appl.* **2015**, *42*, 855–863. [CrossRef]
21. Yu, J.; de Antonio, A.; Villalba-Mora, E. Deep learning (CNN, RNN) applications for smart homes: A systematic review. *Computers* **2022**, *11*, 26. [CrossRef]
22. Lim, B.; Zohren, S. Time-series forecasting with deep learning: A survey. *Philos. Trans. R. Soc. A* **2021**, *379*, 20200209. [CrossRef] [PubMed]
23. Yu, H.; Wu, Z.; Wang, S.; Wang, Y.; Ma, X. Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. *Sensors* **2017**, *17*, 1501. [CrossRef]
24. Khalil, S.; Amrit, C.; Koch, T.; Dugundji, E. Forecasting public transport ridership: Management of information systems using CNN and LSTM architectures. *Procedia Comput. Sci.* **2021**, *184*, 283–290. [CrossRef]

25. Zhao, L.; Song, Y.; Zhang, C.; Liu, Y.; Wang, P.; Lin, T.; Deng, M.; Li, H. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 3848–3858. [CrossRef]
26. Song, H.; Rajan, D.; Thiagarajan, J.; Spanias, A. Attend and diagnose: Clinical time series analysis using attention models. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32, No. 1.
27. Huang, S.; Wang, D.; Wu, X.; Tang, A. Dsanet: Dual self-attention network for multivariate time series forecasting. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; pp. 2129–2132.
28. Hao, S.; Lee, D.H.; Zhao, D. Sequence to sequence learning with attention mechanism for short-term passenger flow prediction in large-scale metro system. *Transp. Res. Part C Emerg. Technol.* **2019**, *107*, 287–300. [CrossRef]
29. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kasier, L.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
30. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 2–9 February 2021; Volume 35, No. 12. pp. 11106–11115.
31. Kitaev, N.; Kaiser, Ł.; Levskaya, A. Reformer: The efficient transformer. In Proceedings of the International Conference on Learning Representations, Virtual, 26 April–1 May 2020.
32. Wang, J.; Yang, C.; Jiang, X.; Wu, J. WHEN: A Wavelet-DTW hybrid attention network for heterogeneous time series analysis. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Long Beach, CA, USA, 6–10 August 2023; pp. 2361–2373.
33. Foumani, N.M.; Tan, C.W.; Webb, G.I.; Salehi, M. Improving position encoding of transformers for multivariate time series classification. *Data Min. Knowl. Discov.* **2024**, *38*, 22–48. [CrossRef]
34. Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; Long, M. iTransformer: Inverted transformers are effective for time series forecasting. In Proceedings of the Twelfth International Conference on Learning Representations, Vienna, Austria, 7–11 May 2023.
35. Yin, D.; Jiang, R.; Deng, J.; Li, Y.; Xie, Y.; Wang, Z.; Zhou, Y.; Song, X.; Shang, J.S. MTMGNN: Multi-time multi-graph neural network for metro passenger flow prediction. *GeoInformatica* **2023**, *27*, 77–105. [CrossRef]
36. Lu, Y.; Zheng, C.; Zheng, S.; Ma, J.; Wu, Z.; Wu, F.; Shen, Y. Multi-Spatio-Temporal Convolutional Neural Network for Short-Term Metro Passenger Flow Prediction. *Electronics* **2023**, *13*, 181. [CrossRef]
37. Ou, J.; Sun, J.; Zhu, Y.; Jin, H.; Liu, Y.; Zhang, F.; Huang, J.; Wang, X. STP-TrellisNets+: Spatial-temporal parallel TrellisNets for multi-step metro station passenger flow prediction. *IEEE Trans. Knowl. Data Eng.* **2022**, *35*, 7526–7540. [CrossRef]
38. Geng, X.; Li, Y.; Wang, L.; Zhang, L.; Yang, Q.; Ye, J.; Liu, Y. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, No. 1, pp. 3656–3663.
39. Song, C.; Lin, Y.; Guo, S.; Wan, H. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, No. 1, pp. 914–921.
40. Liang, Y.; Ke, S.; Zhang, J.; Yi, X.; Zheng, Y. Geoman: Multi-level attention networks for geo-sensory time series prediction. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; Volume 2018, pp. 3428–3434.
41. Guo, S.; Lin, Y.; Feng, N.; Song, C.; Wan, H. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, No. 1, pp. 922–929.
42. Feng, A.; Tassiulas, L. Adaptive Graph Spatial-Temporal Transformer Network for Traffic Forecasting. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, 17–21 October 2022; pp. 3933–3937.
43. Liu, Y.; Guo, B.; Meng, J.; Zhang, D.; Yu, Z. Spatio-Temporal Memory Augmented Multi-Level Attention Network for Traffic Prediction. *IEEE Trans. Knowl. Data Eng.* **2023**, *36*, 2643–2658. [CrossRef]
44. Wu, Z.; Jain, P.; Wright, M.; Mirhoseini, A.; Gonzalez, J.E.; Stoica, I. Representing long-range context for graph neural networks with global attention. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 13266–13279.
45. Topping, J.; Di Giovanni, F.; Chamberlain, B.P.; Dong, X.; Bronstein, M. Understanding over-squashing and bottlenecks on graphs via curvature. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022.
46. He, M.; Wei, Z.; Wen, J.R. Convolutional neural networks on graphs with chebyshev approximation, revisited. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 7264–7276.
47. Wu, H.; Xu, J.; Wang, J.; Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 22419–22430.
48. Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.X.; Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 5243–5253.
49. Bai, S.; Kolter, J.Z.; Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. Universal Language Model Fine-tuning for Text Classification. *arXiv* **2018**, arXiv:1803.01271.
50. Kinga, D.; Adam, J.B. A method for stochastic optimization. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015; Volume 5, p. 6.

51. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
52. Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; Long, M. Timesnet: Temporal 2D-variation modeling for general time series analysis. In Proceedings of the Eleventh International Conference on Learning Representations, Virtual, 25–29 April 2022.
53. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 3844–3852.
54. Choi, J.; Choi, H.; Hwang, J.; Park, N. Graph neural controlled differential equations for traffic forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; Volume 36, No. 6, pp. 6367–6374.
55. Guo, S.; Lin, Y.; Wan, H.; Li, X.; Cong, G. Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. *IEEE Trans. Knowl. Data Eng.* **2021**, *34*, 5415–5428. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.