Multi-ATGCN: A Multi-View Graph Neural Network-based Framework for Citywide Crowd Inflow Forecasting

Songhua Hu*
hsonghua@umd.edu
University of Maryland

Yiqun Xie
xie@umd.edu
University of Maryland

Jiawei Jiang
jwjiang@buaa.edu.cn
Beihang University

Chenfeng Xiong
chenfeng.xiong@villanova.edu
Villanova University

Paul Schonfeld
pschon@umd.edu
University of Maryland

ABSTRACT

Citywide crowd inflow forecasting plays a key role in travel demand modeling and urban planning. However, this task is challenging due to complex spatial dependency, diverse temporal patterns, and heterogeneous external effects. To this end, we propose a Multi-graph Multi-head Adaptive Temporal Graph Convolutional Network (Multi-ATGCN), to fuse multi-view spatial structures, multi-head temporal patterns, and various external effects, for multi-step crowd inflow forecasting. Specifically, Multi-ATGCN has a temporal fusion layer that fuses multiple temporal patterns including closeness, period, and trend. It also has a graph learning module that can adaptively learn graph structures given prior knowledge of inter-zonal distance closeness, origin-destination volume, and functional similarity. In addition, external static variables and time-varying variables are incorporated via parameter initialization and sequence concatenation. Finally, all information are fused and fed into an integration of zone-specific mix-hop GCN and recurrent neural network to explicitly model spatiotemporal dependency. Experiments on two real-world citywide crowd inflow forecasting tasks demonstrate a steady performance improvement and comparable computational efficiency of Multi-ATGCN over state-of-the-art baselines, with even greater improvement exhibited in data-sparse zones and long-horizon prediction.

CCS CONCEPTS

• **Information systems** → **Geographic information systems**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Graph neural network, spatiotemporal graphs, multivariate time series forecasting, human mobility

*Corresponding Author.

KDD '23, August 06–10, 2023, Long Beach, CA

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Songhua Hu, Yiqun Xie, Jiawei Jiang, Chenfeng Xiong, and Paul Schonfeld. 2023. Multi-ATGCN: A Multi-View Graph Neural Network-based Framework for Citywide Crowd Inflow Forecasting. In *Proceedings of Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23).* ACM, New York, NY, USA, 13 pages. https://doi.org/XXXXXXX.XXXXXXX

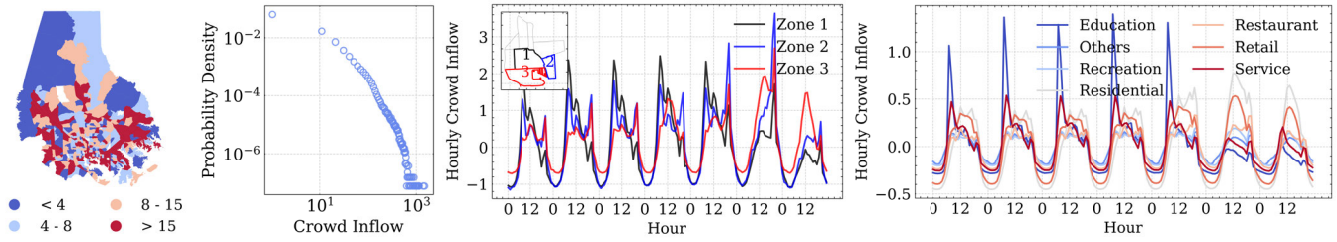
1 INTRODUCTION

Crowd inflow forecasting aims to predict the future volume of population flowing into a specific zone given its historical observations, including both self information (e.g., historical crowd inflow) and external factors (e.g., weather, holidays, and land use) [39]. An accurate and timely prediction of crowd inflow plays a critical role in a wide range of real-world scenarios. For example, it informs operators of on-demand mobility services to better allocate their resources ahead of time to reduce supply-demand imbalance and mitigate shocks [29, 36]. Local merchants can also effectively adjust staffing and capacity to respond to dynamic changes. Broadly in transportation, crowd inflow, a.k.a. trip attraction tables, is also the key component of travel demand modeling [12]. Accurate forecasting of the inflow can largely benefit the whole life cycle of travel demand modeling by providing insights in a time-varying and continuous manner instead of traditional snapshots [12].

The problem of crowd inflow forecasting is challenging as they involve complex spatiotemporal dependency, diverse temporal dynamics, and high nonlinearity triggered by external factors [31]. Compared to traditional statistical and machine learning models that have limited expressiveness and flexibility, deep learning methods have become prevalent because of their strong capability in handling nonlinear relationships, unstructured data, and knowledge fusion [31]. However, there are several challenges to be addressed:

1) **Diverse temporal dynamics**. Temporal patterns of hourly crowd flow time series are highly diverse since they are a mixture of different seasonality (e.g., daily and weekly), trends (e.g., short-term and long-term), and white noise. Meanwhile, they also exhibit high locality since travel behaviors are a function of numerous local factors [12, 22]. Another visible issue is the over-dispersion nature of crowd inflow across different zones, and such a dispersion is further intensified by irregular zone systems (see the near power-law pattern in Figure 1 (b)). The high diversity in crowd inflow time series requires the model can learn both global and local knowledge from heterogeneous zone-specific patterns [2].

2) **Multi-view graph structures**. Unlike microscopic traffic flow mostly constrained by road connectivity, spatial dependency



(a) Spatial plot (b) Log-log plot (c) Time series of nearby zones (d) Time series across land use types

Figure 1: Illustration of crowd inflow in Baltimore.

in crowd inflow is not only constrained by distance but also highly correlated to zonal functionality, accessibility, mobility connectedness, and other unobserved factors [10, 22, 26, 35]. As shown in Figure 1 (a) and (c), the spatial distribution of crowd inflow does not strictly follow a distance decay rule. Nearby census tracts may also present different crowd inflow temporal patterns. Hence, a single predefined adjacency matrix can not well describe the real structure of crowd inflow graphs.

3) **Effects of external factors**. Crowd inflow time series is associated with a variety of external factors with diverse dimensions[14, 20, 36]. Time-varying features like weather and holidays would trigger abnormal fluctuations in crowd inflow. In addition, temporal cycle patterns of the crowd inflow are conditional on the zonal static features such as socioeconomics, demographics, and land use (Figure 1 (d)). Such a diverse set of external information should be carefully handled to enable the model to learn useful knowledge from multi-dimensional variables.

To address these challenges, we propose a Multi-graph Multi-head Adaptive Temporal Graph Convolutional Network (Multi-ATGCN) for multivariable crowd inflow forecasting. Specifically, the Multi-ATGCN contains several main modules to address the challenges accordingly: 1) A *Multi-head temporal fusion* module to fuse multiple temporal patterns including closeness, period, and trend. 2) A *Multi-view adaptive graph construction* module to learn an adaptive graph structure given prior knowledge from different adjacency matrices measured by distance closeness, OD volume, and functional similarity. 3) An integration of *Recurrent neural network* (RNN) and *Zone-specific mix-hop GCN* (ZMGCN) for jointly handling complex spatiotemporal dependency. 4) An *Auxiliary variable enrichment* decorator scattering across the framework to handle external static variables and temporal variables via parameter initialization and sequence concatenation. In summary, the contributions of this study can be highlighted as follows:

1) We propose a general framework for citywide spatiotemporal crowd flow forecasting, which is more comprehensive than previous studies as it integrates multi-head temporal patterns, multi-view spatial structures, and multi-dimensional auxiliary effects.

2) We combine adaptive learning with mix-hop convolution to process graphs given (or without) prior knowledge of their structures. We consider the diversity in temporal patterns by involving zone-specific parameters. Our method provides a guide to handling data with complex temporal dynamics and unclear graph structures.

3) We handle external variables separated by static or time-varying dimensions, which can decrease the risk of polluting the target time series while remaining the model's capability in learning heterogeneous external information.

4) The proposed model is evaluated on two real-world citywide datasets and exhibits steady performance improvement and comparable efficiency over extensive state-of-the-art baselines. Such an improvement is even more salient in data-sparse zones and long-horizon scenarios that are more difficult to predict.

2 RELATED WORK

Crowd inflow forecasting: Crowd inflow time series forecasting has been extensively studied in past decades. Statistical methods such as autoregressive integrated moving average and traditional machine learning models such as support vector machine and tree-based models [29, 37] are mainly used in early studies. However, similar to other traffic forecasting tasks (e.g., traffic speed and volume) [38], one main challenge of crowd inflow forecasting is the complex spatiotemporal dependency, which cannot be well solved by traditional methods [31]. Hence, deep learning methods become prevalent by fusing different deep learning techniques into one framework. Generally, in a hybrid setting, the convolutional neural network (CNN) or GCN is used to capture spatial dependency [10, 30], the CNN or RNN is used to learn temporal dynamics [9, 19, 32, 35], and the fully-connected neural network (FNN) is used to integrate auxiliary information [14, 20, 36, 39]. Although deep learning methods have achieved promising results in crowd flow forecasting, previous studies separately focus on addressing parts of the challenges, while a dearth of studies comprehensively integrates these advances into a holistic entity to test its performance. In addition, external effects are always neglected in previous studies or simply integrated by FNN without considering the diversity in data dimensions. A well-designed module that can handle different types of external variables is absent.

Deep learning in spatiotemporal modeling: RNN has been widely used for time series forecasting, among which Gated Recurrent Unit (GRU) [5] and Long Short-Term Memory (LSTM) [11] are most popular. In recent years, CNN has also been employed in time series forecasting by tackling temporal sequences in a non-recursive manner [32, 33]. As for spatial modeling, early studies directly applied CNN to learn spatial dependency among crowd inflow by treating the city as an image, for example, partitioning a city into grids and applying a convolutional kernel to detect spatial relationships [35, 39]. However, a major limitation of CNN is that it is not compatible with non-Euclidean space. When the analytical unit is irregular, such as administrative zones [26], the adjacency of two pixels in the CNN filter cannot reflect the real closeness.

GCN, on the other hand, is more appropriate because of its ability to capture non-Euclidean relations [30, 31]. One main limitation of current spatiotemporal models is their parameters are globally shareable. The locality is yet to be carefully addressed, which, however, is important for crowd inflow forecasting considering their highly over-dispersion and zone-specific patterns.

Graph structure modeling: One key component of GCN is the adjacency matrix, which is used to describe the graph structure via pairwise zone connectivity. Previous studies have proposed various forms of adjacency matrices, including predefined matrices, adaptive matrices, and multi-view matrices [31]. Predefined matrices are based on prior knowledge of the graph structure, such as zonal functional similarity [8], distance [19], and temporal similarity [39]. These matrices, however, may not precisely reflect the real spatial structure due to unobserved knowledge [10]. Therefore, the adaptive matrix is proposed by setting the adjacency matrix as the learnable parameter. Experiments have proved that the adaptive mechanism helps discover hidden spatial structures and enhance the model performance [2, 10, 32, 33]. Recently, due to the increasingly complex spatial structure, some studies [8, 26, 35] adopt multiple adjacency matrices to describe spatial dependency, i.e., the multi-view GCN. However, few studies have combined multi-view GCN with adaptive graph learning to validate its performance.

3 PROBLEM STATEMENT

This study is intended to forecast the future crowd inflow of each geographic zone across the city. The crowd inflow is defined as the number of people entering a specific zone on an hourly basis. Instead of splitting the city into grids, this study adopts the irregular administrative zone, i.e., the census tract, as the analytical unit. The crowd flow is viewed as a directed graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{A})$, where $\boldsymbol{V}$ is the set of $|\boldsymbol{V}| = N$ zones (i.e., census tracts), $\boldsymbol{E}$ is the set of edges indicating the connectivity between zones, and $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix. Note that one graph may have multiple adjacency matrices to describe multi-view connectivity. Hence, a generalization of $\boldsymbol{A}$ is $\hat{\boldsymbol{A}} \in \mathbb{R}^{M \times N \times N}$, where M is the number of adjacency matrices of the graph $\mathcal{G}$.

In the crowd inflow forecasting problem, the graph structure is assumed to be static over time, while variables attached to each zone may be either static or time-varying [12]. Assumed the whole time series is split into multiple time fragments by a rolling window with several d_E-step historical windows and a d_D-step prediction window, then, for a time fragment whose current time is t_0, each zone i is associated with a feature set including:

1) Time-varying auxiliary features $\boldsymbol{Z}_i^{(t_0)} = [\boldsymbol{z}_{i,t_0-d_E}, \boldsymbol{z}_{i,t_0-d_E+1}, \ldots, \boldsymbol{z}_{i,t_0-1}]^T \in \mathbb{R}^{d_E \times d_F}$, such as holidays, weekends, and weather, where d_F is the number of time-varying auxiliary features.

2) Static features $\boldsymbol{S}_i \in \mathbb{R}^{1 \times d_S}$, such as socioeconomics, demographics, and land use, where d_S is the number of static features.

3) Historical crowd inflow $\widetilde{\boldsymbol{Y}}_i^{(t_0)} = f_T(\boldsymbol{Y}_C, \boldsymbol{Y}_P, \boldsymbol{Y}_T) \in \mathbb{R}^{d_E \times 1}$, where $\boldsymbol{Y}_C, \boldsymbol{Y}_P, \boldsymbol{Y}_T$ are different temporal heads (Eqs. 3-5) and $f_T(.)$ is a weighting function to fuse all temporal heads (Eq. 6).

Let $\boldsymbol{X}^{(t_0)} = [\widetilde{\boldsymbol{Y}}^{(t_0)}, \boldsymbol{Z}^{(t_0)}]$, our goal is to learn a nonlinear map $\mathcal{F}(.)$ from all features to future crowd inflow, given the graph $\mathcal{G}$:

$$\hat{\boldsymbol{Y}}_{:,t_0:(t_0+d_D)} = \mathcal{F}(\boldsymbol{X}^{(t_0)}, \boldsymbol{S}; \mathcal{G}), \ \forall\, t_0 \in [d_E, d_E+1, \ldots, T-d_D] \quad (1)$$

4 PROPOSED APPROACH: MULTI-ATGCN

This study introduces a Multi-ATGCN for multivariable crowd inflow forecasting. Figure 2 shows the high-level architecture, with each module described in detail in the following section.

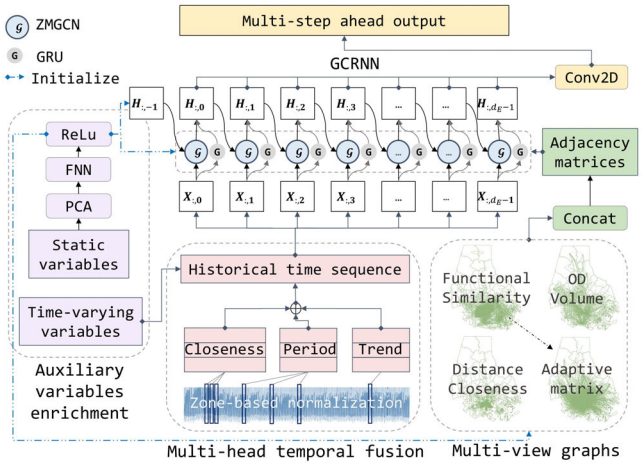


Figure 2: The Multi-ATGCN architecture.

4.1 Multi-head Temporal Fusion

As mentioned before, the distribution of crowd inflow is strongly skewed. Some zones, such as those located in downtown, have a much larger crowd inflow compared with those located in suburban. This may cause the model to focus on a portion of more attractive zones while ignoring those less vibrant zones [12, 24]. Hence, a census tract-based normalization is applied to each time series, followed by a global normalization, to assign a zone-specific bias to each census tract [24]:

$$y'_{i,t} = \left(\frac{y_{i,t} - \mu_i}{\sigma_i} - \mu'\right)/\sigma' \quad (2)$$

where μ_i and σ_i are the mean and standard deviance (st.d.) of the crowd inflow in zone i across the training set; $y_{i,t}$ is the crowd inflow of zone i at time t and $y'_{i,t}$ is its normalization; μ' and σ' are the global mean and st.d. of the normalized crowd inflow.

After normalization, three temporal heads are extracted to represent multi-dimensional temporal patterns [9, 39], including the closeness (daily patterns, green part of Figure 3), the period (weekly patterns, blue part of Figure 3), and the trend (monthly patterns, red part of Figure 3). Let t_0 be the current time, the set of closeness, period, and trend heads are expressed as:

$$\boldsymbol{Y}_C^{(n)} = \boldsymbol{Y}'_{:,(t_0-d_C*n-d_E):(t_0-d_C*n)}, \ n = [0, 1, 2, \ldots, n_C - 1] \quad (3)$$

$$\boldsymbol{Y}_P^{(n)} = \boldsymbol{Y}'_{:,(t_0-d_P*n-d_E):(t_0-d_P*n)}, \ n = [1, 2, \ldots, n_P] \quad (4)$$

$$\boldsymbol{Y}_T^{(n)} = \boldsymbol{Y}'_{:,(t_0-d_T*n-d_E):(t_0-d_T*n)}, \ n = [1, 2, \ldots, n_T] \quad (5)$$

where $\boldsymbol{Y}_C^{(n)}$, $\boldsymbol{Y}_P^{(n)}$, and $\boldsymbol{Y}_T^{(n)} \in \mathbb{R}^{N \times d_E}$ are the set of the n^{th} closeness, period, and trend heads, respectively; d_C, d_P, and d_T are the interval between each two closeness, period, and trend heads, respectively, which are typically defined as the corresponding cycle length (in this study, $d_C = 24h, d_P = 7 * 24h, d_T = 28 * 24h$); n_C, n_P, and n_T

are the number of closeness, period, and trend heads, respectively; $\boldsymbol{Y}'$ is the normalized crowd inflow from Eq. (2).

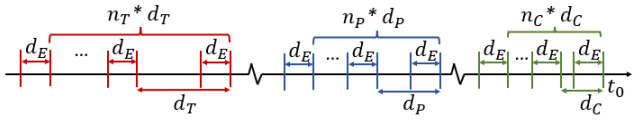


Figure 3: An example of multiple temporal heads.

Considering temporal patterns may vary across different census tracts, a parametric-matrix-based weighting function $f_T(.)$ is designed to fuse multiple temporal heads, which allows the model to adaptively adjust the weight of each temporal head for each zone [39]. In addition, instead of separately feeding the three temporal heads into the network and fusing their outputs in the last layer, this study fuses them before passing through the whole network, which can substantially mitigate memory load and accelerate the training process while remaining comparable accuracy. The output of the multi-head temporal fusion is:

$$\widetilde{\boldsymbol{Y}}^{(t_0)} = \sum_{n=1}^{n_C} \boldsymbol{Y}_C^{(n)} \odot \boldsymbol{W}_C^{(n)} + \sum_{n=1}^{n_P} \boldsymbol{Y}_P^{(n)} \odot \boldsymbol{W}_P^{(n)} + \sum_{n=1}^{n_T} \boldsymbol{Y}_T^{(n)} \odot \boldsymbol{W}_T^{(n)} \quad (6)$$

where $\widetilde{\boldsymbol{Y}}^{(t_0)} \in \mathbb{R}^{N \times d_E}$ is the fused crowd inflow; $\boldsymbol{W}_C^{(n)}, \boldsymbol{W}_P^{(n)}, \boldsymbol{W}_T^{(n)} \in \mathbb{R}^{N \times d_E}$ are zone-specific learnable weights; $\odot$ is Hadamard product.

Note that the multi-head temporal fusion is only applied to historical crowd inflow since other auxiliary time-varying features do not affect the future crowd inflow in such a multi-cycle manner. Other time-varying auxiliary features are directly *concatenated* with $\widetilde{\boldsymbol{Y}}$ to serve as the input of the next step.

4.2 Multi-view Adaptive Graph Learning

Human travel behavior is a function of numerous factors [35]. Hence, the spatial connection of crowd flow cannot be simply described by a single adjacency matrix. In this study, a multi-view graph is proposed to incorporate different types of inter-zonal connectivity. Specifically, three predefined adjacency matrices are first computed based on different measures. Then, a self-adaptive adjacency matrix is designed, initialized by predefined knowledge, and learned end-to-end through stochastic gradient descent. Last, all matrices are stacked together as the final set of adjacency matrices to jointly describe the spatial dependency.

Distance closeness ($\boldsymbol{A_D}$): The distance closeness is measured by the pairwise great circle distance between centroids of two census tracts. The thresholded Gaussian kernel [19] is employed to transfer the distance to the distance-based adjacency matrix:

$$\boldsymbol{A}_{\boldsymbol{D}i,j} = \begin{cases} \exp(-\frac{\text{dist}(i,j)^2}{\sigma^2}), & \text{if } \exp(-\frac{\text{dist}(i,j)^2}{\sigma^2}) \geq \varepsilon \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

where $\boldsymbol{A}_{\boldsymbol{D}i,j}$ is the distance-based edge weight between census tracts i and j; dist(.) is the great circle distance function; σ is the st.d. of distances; ε is the threshold (set as 0.1 here).

Functional (semantic) similarity ($\boldsymbol{A_F}$): The underlying assumption of functional similarity is that zones with similar functionality are more likely to present similar travel patterns. The static variables $\boldsymbol{S}_i$ are used to measure functional similarity, including zonal demographics, socioeconomics, and point-of-interests (POIs). Z-score normalization is applied to each variable across all census tracts, followed by a reciprocal Euclidean distance function:

$$\boldsymbol{A}_{\boldsymbol{F}i,j} = \begin{cases} \frac{1}{\sqrt{\sum_{r=1}^{d_S}(S'^{(r)}_i - S'^{(r)}_j)}}, & \text{if } i \neq j \\ 1, & \text{otherwise} \end{cases} \quad (8)$$

where $\boldsymbol{A}_{\boldsymbol{F}i,j}$ represents the functional similarity between census tracts i and j; $S'^{(r)}_i$ and $S'^{(r)}_j$ denote the r^{th} normalized static variable of the two census tracts i and j.

Origin-destination (OD) volume ($\boldsymbol{A_{OD}}$): OD volume directly measures the connectivity between two census tracts by their travel density. However, limited studies have used it to construct the adjacency matrix perhaps due to the data inaccessibility. To keep the diagonal elements as 1, this study defines the OD-based edge weight as the ratio of average OD volume to the self-loop volume truncated by a maximum of 1:

$$\boldsymbol{A}_{\boldsymbol{OD}i,j} = \min(\frac{OD_{i,j}}{OD_{j,j}}, 1) \quad (9)$$

where $\boldsymbol{A}_{\boldsymbol{OD}i,j}$ represents the OD-based edge weight between census tracts i and j, and $OD_{i,j}$ is the average OD volume between the two census tracts i and j across the training set.

Self-adaptive adjacency matrix ($\boldsymbol{A}_{\check{\boldsymbol{A}}}$): The bidirectional self-adaptive adjacency matrix is defined as the multiplication of two learnable zone embedding matrices $\vec{\boldsymbol{E}}_1 \in \mathbb{R}^{N \times d_{EB}}$ and $\overleftarrow{\boldsymbol{E}}_2 \in \mathbb{R}^{d_{EB} \times N}$:

$$\boldsymbol{A}_{\check{\boldsymbol{A}}i,j} = \text{SoftMax}(\text{ReLU}(\vec{\boldsymbol{E}}_1 \overleftarrow{\boldsymbol{E}}_2)) \quad (10)$$

where $\boldsymbol{A}_{\check{\boldsymbol{A}}i,j}$ denotes the self-adaptive edge weight between census tracts i and j, $\vec{\boldsymbol{E}}_1$ and $\overleftarrow{\boldsymbol{E}}_2$ are the source and target zone embedding, respectively. The ReLU function is used to eliminate weak connections, and the SoftMax function is applied to normalize $\boldsymbol{A}_{\check{\boldsymbol{A}}}$.

The prior knowledge of the adjacency matrix is incorporated into $\boldsymbol{A}_{\check{\boldsymbol{A}}}$ by using them to initialize $\vec{\boldsymbol{E}}_1$ and $\overleftarrow{\boldsymbol{E}}_2$ [32]. Assume the functional similarity matrix is used as the prior knowledge (Functional similarity is selected here since it outperforms others (Table 5)), initialized states of $\vec{\boldsymbol{E}}_1$ and $\overleftarrow{\boldsymbol{E}}_2$ are computed as:

$$\boldsymbol{A_F} = \boldsymbol{P}\text{Diag}(\boldsymbol{H})\boldsymbol{Q}^T \quad (11)$$

$$\vec{\boldsymbol{E}}_{I_1} = \boldsymbol{P}_{:,0:d_{EB}}\text{Diag}(\sqrt{\boldsymbol{H}_{0:d_{EB},:}}) \quad (12)$$

$$\overleftarrow{\boldsymbol{E}}_{I_2} = \text{Diag}(\sqrt{\boldsymbol{H}_{0:d_{EB},:}})\boldsymbol{Q}_{:,0:d_{EB}} \quad (13)$$

where $\boldsymbol{P} \in \mathbb{R}^{N \times N}, \boldsymbol{H} \in \mathbb{R}^{N}, \boldsymbol{Q} \in \mathbb{R}^{N \times N}$ are singular value decomposition (SVD) [27] of $\boldsymbol{A_F}$; $\vec{\boldsymbol{E}}_{I_1} \in \mathbb{R}^{N \times d_{EB}}$, $\overleftarrow{\boldsymbol{E}}_{I_2} \in \mathbb{R}^{d_{EB} \times N}$ are initialized states of $\vec{\boldsymbol{E}}_1$ and $\overleftarrow{\boldsymbol{E}}_2$; Diag(.) is the diagonal function.

All adjacency matrices are stacked vertically along a new dimension to construct the final adjacency matrices $\hat{\boldsymbol{A}} \in \mathbb{R}^{4 \times N \times N}$. $\hat{\boldsymbol{A}}$ is then fed into GCN for multi-graph convolution. Note that adjacency matrices can be easily removed or added depending on the data accessibility. For instance, if all prior knowledge is unavailable, $\hat{\boldsymbol{A}}$ is equal to $\boldsymbol{A}_{\check{\boldsymbol{A}}}$, and $\vec{\boldsymbol{E}}_1$ and $\overleftarrow{\boldsymbol{E}}_2$ can be initialized randomly.

4.3 Zone-specific Mix-hop GCN (ZMGCN)

At each time t, a spectral-based GCN [3] is applied to the crowd inflow to exploit signal correlations in the spatial dimension. Let "$\star_{\mathcal{G}}$" be the graph convolution operator on a graph $\mathcal{G}$, the spectral convolution is defined as the multiplication of a signal $\boldsymbol{X}_{:,t}$ with a kernel f_Θ [16]. Since the computational load of computing the kernel is expensive when the graph is large (time complexity = $O(|V|^3)$), this study adopts the Chebyshev polynomials (time complexity = $O(|E|)$) [25] to approximate the kernel f_Θ. The spectral graph convolution can then be written as [16]:

$$f_\Theta \star_{\mathcal{G}} \boldsymbol{X}_{:,t} = f_\Theta(\boldsymbol{L})\boldsymbol{X}_{:,t} \approx \sum_{k=0}^{K-1} T_k(\widetilde{\boldsymbol{L}})\boldsymbol{X}_{:,t}\boldsymbol{W}_k \tag{14}$$

where $\boldsymbol{W} \in \mathbb{R}^{K\times d_I\times d_O}$ is the polynomial weights and $\boldsymbol{W}_k \in \mathbb{R}^{d_I\times d_O}$ is its k^{th}-hop weight matrix; d_I and d_O are the dimension of input and output, respectively; $\widetilde{\boldsymbol{L}}$ is the scaled Laplacian matrix [16]; $T_k(\widetilde{\boldsymbol{L}}) \in \mathbb{R}^{N\times N}$ is the k^{th} Chebyshev polynomial approximation, which can be recursively computed as: $T_k(\widetilde{\boldsymbol{L}}) = 2\widetilde{\boldsymbol{L}}T_{k-1}(\widetilde{\boldsymbol{L}}) - T_{k-2}(\widetilde{\boldsymbol{L}})$, $T_0(\widetilde{\boldsymbol{L}}) = \boldsymbol{I_N}$, $T_1(\widetilde{\boldsymbol{L}}) = \widetilde{\boldsymbol{L}}$, where $\boldsymbol{I_N}$ is an identity matrix.

One issue of Eq. (14) is that for each hop, all zones share the same weight matrix $\boldsymbol{W}_k$, which is not optimal for crowd inflow forecasting since mobility patterns among census tracts are diverse. To increase the local diversity, this study introduces *zone-specific* parameterization in graph convolution. Specifically, $\boldsymbol{W}_k$ is modified as the multiplication of a zone embedding matrix $\boldsymbol{E}_{\mathcal{G}} \in \mathbb{R}^{N\times d_{EB}}$ and a zone-aware weight matrix $\boldsymbol{\Psi}_k \in \mathbb{R}^{d_{EB}\times d_I\times d_O}$:

$$f_\Theta \star_{\mathcal{G}} \boldsymbol{X}_{:,t} \approx \sum_{k=0}^{K-1} T_k(\widetilde{\boldsymbol{L}})\boldsymbol{X}_{:,t}\boldsymbol{E}_{\mathcal{G}}\boldsymbol{\Psi}_k \tag{15}$$

The reason to introduce a $\boldsymbol{E}_{\mathcal{G}}$ rather than directly expanding the size of $\boldsymbol{W}_k$ to $\mathbb{R}^{N\times d_I\times d_O}$ is to reduce parameter numbers, which can enhance computational efficiency [17], particularly for large graphs. Also, $\boldsymbol{E}_{\mathcal{G}}$ is allowed to share among hops and adjacency matrices to mitigate the memory burden. To include external variables, the $\boldsymbol{E}_{\mathcal{G}}$ is initialized by the principal components of their static features $\boldsymbol{S}$:

$$\boldsymbol{P}_S Diag(\boldsymbol{H}_S)\boldsymbol{Q}_S{}^T = \boldsymbol{S} \tag{16}$$

$$\boldsymbol{E}_{I_{\mathcal{G}}} = ReLU(\boldsymbol{S}\boldsymbol{Q}_S\boldsymbol{W}_S + \boldsymbol{b}_S) \tag{17}$$

where $\boldsymbol{E}_{I_{\mathcal{G}}}$ is the initialized state of $\boldsymbol{E}_{\mathcal{G}}$; $\boldsymbol{P}_S \in \mathbb{R}^{N\times d_{EB}}, \boldsymbol{H}_S \in \mathbb{R}^{d_{EB}}, \boldsymbol{Q}_S \in \mathbb{R}^{d_S\times d_{EB}}$ are outputs of principal component analysis (PCA) [27] of $\boldsymbol{S}$; $\boldsymbol{S}\boldsymbol{Q}_S \in \mathbb{R}^{N\times d_{EB}}$ is the projection of the first k principal components of $\boldsymbol{S}$; $\boldsymbol{W}_S$ and $\boldsymbol{b}_S$ are the weight and bias.

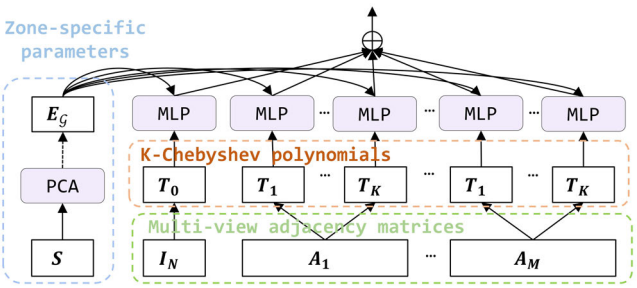


Figure 4: The ZMGCN architecture.

The aforementioned GCN can be easily generalized to multi-view graphs by computing Chebyshev polynomials for each adjacency matrix and fusing them together (Figure 4). This study employs a *mix-hop* manner for hop fusion with hop-wise learnable weights. Moreover, to avoid repeatedly counting the self-loop matrix, $\boldsymbol{I_N}$ is separately computed. The multi-view GCN can be written as:

$$f_\Theta \star_{\mathcal{G}} \boldsymbol{X}_{:,t} \approx \gamma_0 \boldsymbol{I_N}\boldsymbol{X}_{:,t}\boldsymbol{E}_{\mathcal{G}}\boldsymbol{\Psi}_0 + \sum_{m=1}^{M}\sum_{k=1}^{K-1} \gamma_k^{(m)} T_k(\widetilde{\boldsymbol{L}}^{(m)})\boldsymbol{X}_{:,t}\boldsymbol{E}_{\mathcal{G}}\boldsymbol{\Psi}_k^{(m)} \tag{18}$$

where $\boldsymbol{\Psi}_k^{(m)}$ is the zone-aware weight matrix of the k^{th} Chebyshev polynomial approximation for the m^{th} adjacency matrix; M is the number of adjacency matrices; $\boldsymbol{\Psi}_0$ is the zone-aware weight matrix for the $\boldsymbol{I_N}$; $\widetilde{\boldsymbol{L}}^{(m)}$ is the scaled Laplacian matrix derived from the m^{th} adjacency matrix; γ_0 and $\gamma_k^{(m)}$ are the Softmax-transformed fusion weights for each hop. In sum, Eq. (18) can be viewed as a high-level representation of crowd inflow exploiting the mixed information from $(K-1)$-hop neighborhoods, where neighborhoods are determined by different types of adjacency matrices.

4.4 Graph Convolutional RNN (GCRNN)

This study integrates the ZMGCN into the GRU-based RNN, named the graph convolutional recurrent neural network (GCRNN), to jointly capture spatiotemporal dependency of crowd inflow. The form of the GCRNN is as follows:

$$\boldsymbol{\Gamma}_{u,t} = \sigma(f_{\Theta_u} \star_{\mathcal{G}} [\boldsymbol{X}_{:,t}, \boldsymbol{H}_{:,t-1}] + \boldsymbol{E}_{\mathcal{G}}\boldsymbol{b}_u) \tag{19}$$

$$\boldsymbol{\Gamma}_{r,t} = \sigma(f_{\Theta_r} \star_{\mathcal{G}} [\boldsymbol{X}_{:,t}, \boldsymbol{H}_{:,t-1}] + \boldsymbol{E}_{\mathcal{G}}\boldsymbol{b}_r) \tag{20}$$

$$\widetilde{\boldsymbol{H}}_{:,t} = \tanh(f_{\Theta_H} \star_{\mathcal{G}} [\boldsymbol{X}_{:,t}, \boldsymbol{\Gamma}_{r,t} \odot \boldsymbol{H}_{:,t-1}] + \boldsymbol{E}_{\mathcal{G}}\boldsymbol{b}_c) \tag{21}$$

$$\boldsymbol{H}_{:,t} = (1 - \boldsymbol{\Gamma}_{u,t}) \odot \boldsymbol{H}_{:,t-1} + \boldsymbol{\Gamma}_{u,t} \odot \widetilde{\boldsymbol{H}}_{:,t} \tag{22}$$

where $t = [t_0 - d_E, t_0 - d_E + 1, \ldots, t_0 - 1]$ is the time index of each step in the historical window and t_0 is the current time; $\star_{\mathcal{G}}$ denotes the ZMGCN defined in Eq. (18) and $f_{\Theta_u}, f_{\Theta_r}, f_{\Theta_H}$ are corresponding kernels; $\boldsymbol{X}_{:,t} \in \mathbb{R}^{N\times d_I}$ is the input signal at time t; $\boldsymbol{H}_{:,t} \in \mathbb{R}^{N\times d_H}$ is the hidden state at time t, which is a linear combination of the previous state $\boldsymbol{H}_{:,t-1}$ and the candidate's state $\widetilde{\boldsymbol{H}}_{:,t}$; $\boldsymbol{\Gamma}_{u,t}$ and $\boldsymbol{\Gamma}_{r,t} \in \mathbb{R}^{N\times d_H}$ are the update gate and reset gate, respectively; $\boldsymbol{E}_{\mathcal{G}} \in \mathbb{R}^{N\times d_{EB}}$ is the zone embedding matrix and $\boldsymbol{b}_u, \boldsymbol{b}_r, \boldsymbol{b}_c \in \mathbb{R}^{d_{EB}\times d_H}$ are zone-specific biases.

Some additional techniques are added to GCRNN to enhance model robustness and accuracy. First, temporal patterns of crowd inflow are conditional on inherent features (i.e., $\boldsymbol{S}$) of census tracts. However, directly appending static variables to $\boldsymbol{X}_t$ may pollute the sequence data with non-sequential information [15]. To avoid it, this study includes the effects of static variables by using their principal components to initialize the hidden state of GCRNN, i.e., $\boldsymbol{H}_{:,t_0-d_E-1}$ (Similar to Eqs. (16, 17)). In addition, the adjacency matrix may not accurately reflect the real zone connectivity, which would induce irrelevant noise into the graph convolution. Hence, a skip connection is included along the GCRNN with the GRU as a bypass path (Figure 2). Assume $\ddot{\boldsymbol{H}}_{:,t} \in \mathbb{R}^{N\times d_H}$ is the output of the bypass GRU, the final output of the GCRNN can be expressed as:

$$\check{\boldsymbol{H}}_{:,t} = \sigma(\boldsymbol{\alpha}_t)\boldsymbol{H}_{:,t} + (1 - \sigma(\boldsymbol{\alpha}_t))\ddot{\boldsymbol{H}}_{:,t} \tag{23}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{d_E}$ is the learnable fusion weight and $\boldsymbol{\alpha}_t$ is the weight for the t^{th} step in the historical window. Note that the GCRNN can be easily extended to multiple layers by stacking the modules vertically and passing the output of the last layer into the next layer.

4.5 Multi-step Output

Traditional RNN generates the prediction based on its last hidden state (i.e., $\boldsymbol{H}_{:,t_0-1}$), which may lose accuracy when making multi-step predictions due to the memory vanishing in long sequences. The attention-based mechanism has been proposed by including information from all hidden states [1]. This study takes inspiration from it by including all hidden states (i.e., $\boldsymbol{H}_{:,(t_0-d_E):t_0}$) to generate multi-step predictions. In addition, instead of employing an RNN-based decoder to recursively generate multi-step outputs, this study directly places a 2D CNN (kernel size = $(1, d_H)$, # input channels = d_E, # output channels = d_D) to transform hidden states into normalized d_D-step crowd inflow, which can substantially increase the computational efficiency without losing accuracy. Finally, the mean absolute error (MAE) is used as the model loss:

$$\mathcal{L}(\mathcal{F}_\theta) = \sum_{i=1}^{N} \sum_{t=t_0}^{t_0+d_D-1} |\hat{y}_{i,t} - y_{i,t}| \tag{24}$$

where $y_{i,t}$ is the crowd inflow of census tract i at time t and $\hat{y}_{i,t}$ is its prediction; $\mathcal{L}(.)$ is the loss function; $\mathcal{F}_\theta$ represents all parameters in the nonlinear mapping $\mathcal{F}(.)$.

5 EXPERIMENTS

5.1 Data Description

Current public spatiotemporal datasets are mainly related to traffic flow, while a comprehensive citywide crowd flow dataset including various external variables is absent. In this study, we collected and prepared such datasets and make them public on GitHub[1]. The crowd inflow is calculated using data from SafeGraph [23], a data company that aggregates anonymized mobile device location data (MDLD) in the US. All MDLD are de-identified and contain no private personal information. Specifically, the Core Places US dataset is used to obtain the geographical coordinates of each POI. Then, the Weekly Places Patterns (v2) dataset is used to extract the POI-level hourly visit. Last, the hourly visit is aggregated at a census tract level. The weekly OD volume is extracted as well for graph adjacency matrix building. Finally, two cities are selected as case studies: Washington, D.C., and Baltimore, and their data statistics are reported in Table 1. Visualization of crowd flow time series is shown in Appendix A.

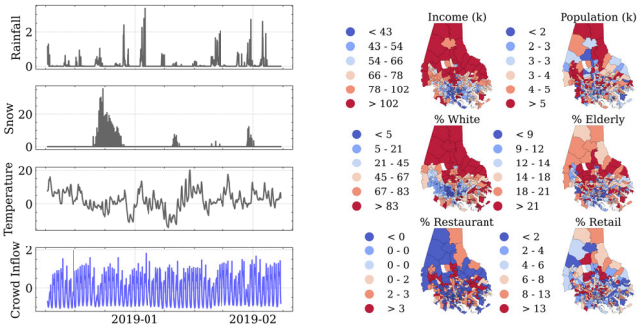
A set of auxiliary variables are collected, including time-varying variables (holiday, weekend, precipitation, temperature, snowfall) and static variables (demographics, socioeconomics, land use). An illustration of part of the collected variables is depicted in Figure 5. Among them, socioeconomics and demographics are from the 2015–2019 American Community Survey of the US Census Bureau. POI features are from SafeGraph. Partisanship is from the 2016 presidential election result provided by the MIT election lab. Weather conditions are from NOAA's National Centers for Environmental Information. The detailed static variables include:

[1]https://github.com/SonghuaHu-UMD/MultiSTGraph/tree/master/raw_data

Table 1: Data statistics

	Washington, D.C.	Baltimore
Date Range	01/01/2019 – 05/31/2019	
Sample Rate	1 hour	
Input length (d_E)	24 hours	
Output length (d_D)	3 hours, 6 hours, 12 hours, 24 hours	
# Zones	237	403
# Samples	858,888	1, 460, 472
Mean	30.169	14.410
St.d.	84.023	29.300

- **Demographics**: % non-Hispanic Whites, % African Americans, % Asians, % Hispanics, % males, % residents 18-44, % residents 45-64, % residents >65.
- **Socioeconomics**: Total population, % urbanized populations, median household income, % Democrats, % Republicans, % highly-educated residents.
- **Land use**: Area, # residential POIs, # retail trade, # personal and public services, # educational institutions, # recreation, # restaurants, # other POIs.



(a) Time-varying **(b) Static**

Figure 5: Illustration of external variables in Baltimore.

5.2 Baselines for Comparison

The performance of Multi-ATGCN is extensively compared with a variety of baselines and state-of-the-art models, including:

1) Baselines without GCN:

- **FNN**: A basic 2-layer FNN with ReLU as the activation.
- **LSTM** [11]: A sequence-to-sequence (S2S) LSTM-based RNN.
- **GRU** [4]: A S2S GRU-based RNN.

2) Baselines with pre-defined GCN:

- **STGCN** [37]: A integration of GCN and temporal 1D CNN;
- **DCRNN** [19]: A diffusion convolutional RNN that models temporal dynamics using GRU and captures spatial dependency via diffusion GCN.

3) Baselines with attention-based GCN:

- **ASTGCN** [9]: An attention-based spatiotemporal GCN, which considers multiple temporal heads and integrates spatiotemporal attention mechanisms into GCN.
- **GMAN** [40]: A multi-attention GCN that leverages the node2vec algorithm to learn zone structural information while performing spatiotemporal attention mechanisms.

4) Baselines with adaptive GCN:

- **GWNET** [33]: A spatiotemporal GCN that integrates self-adaptive diffusion GCN for spatial modeling with stacked dilated 1D CNNs for temporal modeling.
- **AGCRN** [2]: An adaptive graph convolutional RNN that enhances graph convolutions by zone-specific parameters and self-adaptive graph learning.
- **MTGNN** [32]: A GCN framework for multivariate time series forecasting, which includes a graph learning module, a mix-hop propagation layer for spatial modeling, and a dilated inception layer for temporal modeling.

5) Baselines with differential equation-based GCN:

- **STGODE** [7]: A spatiotemporal graph ordinary differential equation network which captures spatiotemporal dynamics through a tensor-based ordinary differential equation.
- **STG-NCDE** [6]: A spatiotemporal graph neural controlled differential equation (NCDE) which connects two NCDEs for spatial and temporal processing.

5.3 Experiment Settings

All baselines, including the Multi-ATGCN, are implemented in PyTorch 1.10.2 and executed on servers with NVIDIA Tesla T4 GPU. The implementation details of all models can be found in Appendix B. The Adam optimizer is employed to minimize the model loss with the learning rate decaying from 0.003, with early stop strategy, gradient clipping, and dropout applied. Datasets are split into training sets, validation sets, and test sets according to chronological order with a split ratio of 7:1.5:1.5. The best hyperparameters are chosen using the asynchronous successive halving algorithm (ASHA) [18], with detailed parameter study results reported in Appendix C. Multi-ATGCN achieves the best performance when zone embedding dimension (d_{EB}) = 20, orders of Chebyshev polynomials (K) = 2, # RNN layers = 2, # RNN hidden units (d_H) = 64, # closeness heads (n_C) = 2, # period heads (n_P) = 1, and # trend heads (n_T) = 1. For easy comparison among different models, the code and data formats follow the framework proposed by [28]. Codes and datasets are available on GitHub[2].

This study deploys three widely used metrics to evaluate model performances on the testing set, including mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean square error (RMSE). Each experiment is repeated 10 times with different random seeds and average metrics are reported. Similar to [35], this study set the lower bound of hourly crowd inflow as 10. Low-demand scenarios are less important for real-world applications. Sensitivity analysis of the effects of lower bound on model performance is reported in Appendix D.

5.4 Baseline Comparison

Table 2 shows the average performances of all models on the two testing datasets, with the forecasting horizon varying from 3 to 24 hours. Percentages in brackets are the performance increase of Multi-ATGCN versus the baseline. T-test is conducted between the outcomes of each baseline and Multi-ATGCN. Overall, Multi-ATGCN achieves state-of-the-art results on all tasks, outperforming all baselines over different horizons. Compared with the best baseline (underlined), Multi-ATGCN yields a 1.9-2.8%, 2.8-3.1%, 3.8-7.4%, and 5.1-6.4% reduction in MAE for 3, 6, 12, and 24 horizons prediction respectively (> 95% confidence). One noteworthy finding is that the performance improvement of Multi-ATGCN increases with the prediction horizon. The reasons are several folds. First, long-horizon prediction relies more on period and trend temporal patterns; hence, models that include multiple temporal heads such as Multi-ATGCN (ASTGCN as well) become superior. Second, long-horizon dependency is more complex. Hence, models that can ingest more information such as multi-view spatial structures and external effects may gain greater benefits.

The performance of the same model varies significantly across datasets and forecasting horizons among the baselines. However, it is consistently observed that baselines without GCN exhibit the poorest performance, while those with adaptive GCN, such as MTGNN and GWNET, consistently rank in the first tier. Attention-based baselines, such as GMAN, follow closely behind. In addition, baselines with well-designed graphs, such as AGCRN and GMAN, perform better when the prediction horizon is short, while those relying more on self-time series, such as STGCN and ASTGCN, show higher performance when the prediction horizon is long.

Figure 6 shows the 24-hour forecasting results of Multi-ATGCN in census tracts where the model shows the best and worst performance. As shown, crowd inflow in zones with the best performances presented more rhythmic patterns and thus is easier to predict. On the other hand, crowd inflow in zones with poor prediction accuracy is more randomly fluctuating. For those zones, although the Multi-ATGCN cannot well fit observations, it still shows the ability to generate a stable output following the average trend. Another noteworthy finding is that the well-performed census tracts have a much larger crowd inflow ($> 10^2$) compared to those poorly-performed zones (< 10), indicating the over-dispersion distribution of crowd flow may be an underlying factor leading to unfair global models. Hence, considering zone-specific biases and parameters are particularly important to mitigate model unfairness [34].

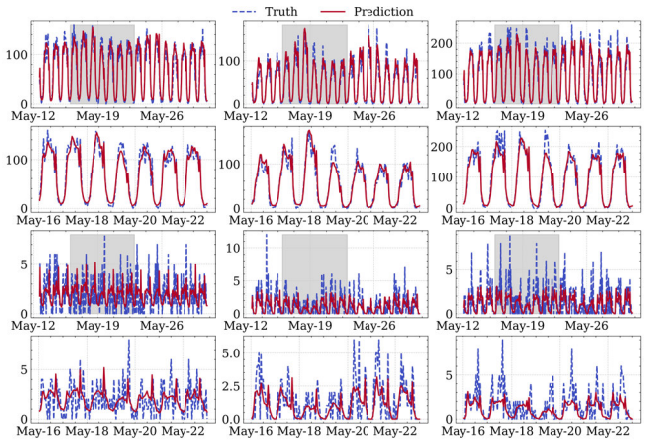


Figure 6: Forecasting results of the top and last three census tracts in Baltimore. The first (third) row shows the forecasting results of the top (last) three census tracts, with the gray areas zooming in and showing in the next row.

[2]https://github.com/SonghuaHu-UMD/MultiSTGraph

Table 2: Model performances comparison

Baltimore												
	Horizon = 3			Horizon = 6			Horizon = 12			Horizon = 24		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
FNN	9.82 (26.7%)***	19.02 (29.9%)***	0.29 (19.0%)***	10.44 (28.3%)***	20.46 (27.7%)***	0.31 (21.0%)***	11.29 (31.4%)***	22.22 (31.4%)***	0.32 (19.9%)***	11.91 (34.0%)***	23.41 (32.4%)***	0.34 (23.8%)***
LSTM	8.34 (13.7%)***	17.56 (24.0%)***	0.27 (11.3%)***	8.77 (14.6%)***	18.40 (19.6%)***	0.28 (12.1%)***	9.12 (15.1%)***	19.14 (20.3%)***	0.28 (8.2%)***	9.81 (19.8%)***	20.58 (23.1%)***	0.30 (14.1%)***
GRU	8.21 (12.3%)***	17.29 (22.8%)***	0.26 (9.9%)***	8.58 (12.7%)***	18.00 (17.8%)***	0.28 (10.1%)***	9.10 (14.9%)***	19.10 (20.2%)***	0.28 (8.0%)***	9.49 (17.2%)***	19.92 (20.5%)***	0.29 (11.2%)***
DCRNN	7.98 (9.8%)***	15.96 (16.4%)***	0.26 (8.2%)***	8.27 (9.5%)***	16.62 (11.0%)***	0.27 (7.1%)***	9.04 (14.3%)***	18.26 (16.5%)***	0.28 (7.3%)***	9.26 (15.0%)***	19.03 (16.8%)***	0.28 (9.1%)***
STGCN	7.42 (2.9%)***	13.74 (2.9%)***	0.24 (2.2%)**	7.87 (4.8%)***	15.09 (1.9%)**	0.26 (4.5%)***	8.10 (4.3%)***	16.24 (6.1%)***	0.26 (1.6%)**	8.28 (5.1%)***	16.53 (4.2%)***	0.27 (3.1%)***
STG-NCDE	7.96 (9.5%)***	14.06 (5.1%)***	0.25 (5.7%)***	8.76 (14.5%)***	15.84 (6.6%)***	0.27 (8.1%)***	8.95 (13.4%)***	16.87 (9.6%)***	0.28 (6.5%)***	9.19 (14.4%)***	17.95 (11.8%)***	0.29 (11.7%)***
STGODE	7.47 (3.6%)***	13.79 (3.3%)***	0.25 (3.4%)***	8.03 (6.8%)***	15.77 (6.2%)***	0.27 (7.2%)***	8.53 (9.2%)***	16.67 (8.5%)***	0.27 (4.9%)***	9.04 (12.9%)***	17.46 (9.3%)***	0.29 (9.3%)***
ASTGCN	8.14 (11.5%)***	16.09 (17.1%)***	0.26 (10.7%)***	8.25 (9.2%)***	16.42 (9.9%)***	0.27 (8.8%)***	8.67 (10.7%)***	17.24 (11.6%)***	0.28 (6.5%)***	8.54 (8.0%)***	16.83 (5.9%)***	0.27 (5.4%)***
GMAN	7.53 (4.4%)***	14.09 (5.3%)***	0.24 (2.3%)***	7.82 (4.2%)***	15.37 (3.7%)***	0.26 (5.4%)***	8.41 (7.9%)***	16.64 (8.3%)***	0.27 (4.5%)***	8.98 (12.4%)***	18.45 (14.2%)***	0.29 (11.1%)***
GWNET	7.76 (7.3%)***	14.28 (6.5%)***	0.25 (5.9%)***	8.25 (9.2%)***	15.96 (7.3%)***	0.27 (6.8%)***	8.39 (7.7%)***	16.49 (7.5%)***	0.26 (1.7%)*	8.95 (12.2%)***	17.75 (10.8%)***	0.28 (7.9%)***
AGCRN	7.40 (2.7%)***	13.94 (4.2%)***	0.24 (2.1%)***	8.23 (9.0%)***	15.79 (6.2%)***	0.27 (7.0%)***	9.01 (14.0%)***	17.52 (13.0%)***	0.28 (7.0%)***	9.31 (15.5%)***	18.74 (15.5%)***	0.29 (9.8%)***
MTGNN	7.40 (2.8%)***	13.63 (2.1%)**	0.24 (1.7%)**	7.73 (3.1%)***	14.92 (0.8%)*	0.25 (2.3%)**	8.05 (3.8%)***	15.99 (4.6%)***	0.26 (1.3%)*	8.60 (8.6%)***	17.33 (8.6%)***	0.28 (6.6%)***
Multi-ATGCN	**7.2**	**13.34**	**0.24**	**7.49**	**14.8**	**0.25**	**7.75**	**15.25**	**0.26**	**7.87**	**15.83**	**0.26**
Washington, D.C.												
	Horizon = 3			Horizon = 6			Horizon = 12			Horizon = 24		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
FNN	14.73 (30.3%)***	44.53 (36.6%)***	0.29 (18.6%)***	16.38 (31.7%)***	51.63 (39.3%)***	0.32 (18.9%)***	18.24 (36.6%)***	58.45 (44.2%)***	0.33 (20.5%)***	19.50 (37.2%)***	63.93 (47.0%)***	0.35 (20.5%)***
LSTM	14.38 (28.6%)***	48.06 (41.3%)***	0.29 (18.4%)***	14.41 (22.3%)***	51.12 (38.7%)***	0.29 (11.4%)***	13.74 (15.8%)***	44.73 (27.1%)***	0.28 (6.8%)***	16.01 (23.5%)***	52.38 (35.3%)***	0.32 (12.5%)***
GRU	14.11 (27.2%)***	47.05 (40.0%)***	0.28 (17.9%)***	13.84 (19.1%)***	46.24 (32.2%)***	0.28 (9.4%)***	13.76 (15.9%)***	42.44 (23.2%)***	0.29 (8.6%)***	15.31 (20.0%)***	47.52 (28.7%)***	0.31 (11.5%)***
DCRNN	12.50 (17.9%)***	38.41 (26.5%)***	0.27 (12.4%)***	12.65 (11.5%)***	38.05 (17.7%)***	0.27 (5.8%)***	13.34 (13.3%)***	38.31 (14.9%)***	0.28 (7.3%)***	14.78 (17.2%)***	42.22 (19.8%)***	0.31 (10.6%)***
STGCN	11.46 (10.3%)***	32.41 (12.9%)***	0.25 (6.5%)***	11.93 (6.1%)***	33.29 (5.9%)***	0.27 (3.5%)***	12.49 (7.4%)***	36.40 (10.5%)	0.27 (2.9%)**	13.11 (6.6%)***	38.11 (11.1%)***	0.29 (4.6%)***
STG-NCDE	11.89 (13.6%)***	33.35 (15.4%)***	0.26 (12.7%)***	12.82 (12.6%)***	34.31 (8.7%)***	0.29 (9.1%)***	13.10 (11.7%)***	37.63 (13.4%)***	0.29 (9.8%)***	14.58 (16.0%)***	39.40 (14.0%)***	0.32 (11.7%)***
STGODE	11.10 (13.4%)***	32.58 (13.4%)***	0.26 (10.8%)***	11.83 (5.4%)***	33.19 (5.6%)***	0.27 (4.2%)***	12.51 (7.5%)***	36.62 (11.0%)***	0.28 (5.7%)***	13.37 (8.5%)***	38.75 (12.6%)***	0.30 (5.6%)***
ASTGCN	12.16 (15.5%)***	33.71 (16.2%)***	0.27 (12.9%)***	11.87 (5.7%)***	32.86 (4.7%)***	0.27 (4.9%)***	13.13 (11.9%)***	35.85 (9.1%)***	0.28 (5.9%)***	13.08 (6.4%)***	35.76 (5.3%)***	0.29 (5.2%)***
GMAN	10.79 (4.9%)***	30.16 (6.4%)***	0.24 (1.1%)*	12.25 (8.6%)***	36.50 (14.2%)***	0.26 (1.3%)*	13.01 (11.1%)***	37.22 (12.4%)***	0.28 (5.4%)***	13.53 (9.5%)***	38.31 (11.6%)***	0.29 (6.2%)***
GWNET	11.51 (10.8%)***	31.77 (11.1%)***	0.25 (6.1%)***	12.07 (7.2%)***	33.28 (5.9%)***	0.26 (1.3%)*	12.58 (8.1%)***	35.08 (7.1%)***	0.27 (2.8%)**	14.04 (12.8%)***	41.47 (18.3%)***	0.30 (7.2%)***
AGCRN	10.47 (1.9%)**	28.54 (1.1%)*	0.24 (1.0%)*	11.95 (6.3%)***	34.37 (8.9%)***	0.27 (3.7%)***	12.62 (8.3%)***	35.74 (8.8%)***	0.27 (3.1%)***	14.70 (16.7%)***	41.09 (17.6%)***	0.30 (9.1%)***
MTGNN	10.49 (2.1%)**	28.64 (1.4%)*	0.24 (1.3%)*	11.52 (2.8%)**	33.80 (7.3%)***	0.26 (1.0%)*	12.83 (9.9%)***	35.81 (9.0%)***	0.28 (5.7%)***	13.85 (11.6%)***	38.07 (11.0%)***	0.30 (7.1%)***
Multi-ATGCN	**10.27**	**28.23**	**0.23**	**11.2**	**31.33**	**0.26**	**11.57**	**32.59**	**0.26**	**12.24**	**33.87**	**0.28**

Significance codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1.

5.5 Model Analysis

Performance across census tracts: Checking how the model performs across different zones allows us to compare and detect model weaknesses. Figure 7 shows how model performances vary across census tracts with different POI counts based on the 24-hour forecasting for Baltimore. As shown, MAPE broadly decreases with the increase of POI count, while MAE and RMSE follow an increasing pattern. It is plausible since MAPE is a relative metric while MAE and RMSE are absolute metrics. On the one hand, zones with fewer POIs are less likely to attract higher population flow, resulting in lower absolute metrics. On the other hand, zones with fewer POIs are more difficult to predict due to their higher randomness (Figure 6) and higher sensitivity toward external interventions [12], leading to higher relative metrics. It is also worth mentioning that Multi-ATGCN presents a predominantly superior performance in data-sparse zones. Compared with the best baselines, i.e., STGCN and ASTCGN, the best baseline on 24-hour forecasting, Multi-ATGCN leads to an 8.7%, 19.1%, and 13.3% reduction in MAPE, RMSE, and MAE, respectively, in census tracts with the fewest POIs. Such an improvement demonstrates that efforts in involving zone-specific parameters, capturing external effects, and learning complex spatial structures can successfully improve the model capability in handling more intractable tasks.

Ablation Study: An ablation study is conducted on the Baltimore data to validate the effectiveness of key components that contribute to the model performance when making the 24-hour prediction. The Multi-ATGCNs without different components are outlined as follows:

- **w/o Auxiliary**: Multi-ATGCN w/o all auxiliary variables.
- **w/o Closeness**: Multi-ATGCN w/o closeness temporal heads.
- **w/o Period**: Multi-ATGCN w/o period temporal heads.
- **w/o Trend**: Multi-ATGCN w/o trend temporal heads.

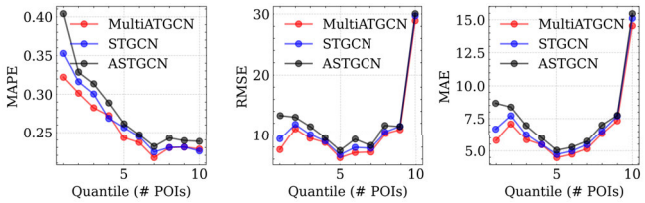


Figure 7: (Top 3) Model performance across zones with different POI counts. POI counts are categorized by deciles.

- **w/o ZBN**: Multi-ATGCN w/o zone-based normalization. Specifically, a global z-score normalization is applied to all crow inflow.
- **w/o ZSP**: Multi-ATGCN w/o zone-specific parameters. Specifically, the zone embedding dimension d_{EB} is set as 1, and $\boldsymbol{E}_{\mathcal{G}}$ is set as a fixed tensor filled with 1.
- **w/o GCN**: Multi-ATGCN w/o the graph convolution. Specifically, the GCRNN is replaced by a two-layer GRU-based RNN.
- **w/o FNNO**: Multi-ATGCN w/o fully-connected output layer that uses all hidden states. Specifically, only the last hidden state from GCRNN is fed into the output layer.

The mean and st.d. of all metrics on the testing set over 10 repetitions for each version of Multi-ATGCN are reported in Table 3. The fully-connected output layer that uses all hidden states exhibits the greatest contribution to model performance, validating that including all hidden states of the GCRNN is helpful for long-horizon prediction. This also explains why RNN-based models such as AGCRN deteriorate significantly with the increase of prediction horizon. The effect of GCN is evident as well, indicating the importance of enabling the information flow among interdependent zones. The effects of closeness and period temporal head are equivalently great, while the benefit brought by trend temporal head is minor. The auxiliary information also significantly increases the

Table 3: Ablation study

	MAE	RMSE	MAPE
	Mean±st.d.(Δ%)	Mean±st.d.(Δ%)	Mean±st.d.(Δ%)
w/o Auxiliary	8.113 ± 0.125(3.04%)	16.370 ± 0.711(3.38%)	0.265 ± 0.004(2.09%)
w/o Closeness	8.144 ± 0.114(3.43%)	16.747 ± 0.249(5.76%)	0.269 ± 0.004(3.83%)
w/o Trend	7.954 ± 0.117(1.02%)	16.384 ± 0.197(3.47%)	0.264 ± 0.004(1.82%)
w/o Period	8.144 ± 0.100(3.44%)	16.774 ± 0.357(5.94%)	0.267 ± 0.003(2.81%)
w/o ZSP	7.976 ± 0.076(1.30%)	16.085 ± 0.255(1.59%)	0.261 ± 0.002(0.57%)
w/o ZBN	8.036 ± 0.110(2.06%)	16.319 ± 0.289(3.06%)	0.265 ± 0.003(1.92%)
w/o GCN	8.151 ± 0.086(3.53%)	16.520 ± 0.651(4.33%)	0.266 ± 0.002(2.41%)
w/o FNNO	8.314 ± 0.204(5.60%)	16.743 ± 0.703(5.74%)	0.270 ± 0.006(3.93%)
Multi-ATGCN	**7.874 ± 0.078**	**15.834 ± 0.606**	**0.260 ± 0.002**

Table 4: The comparison of computational cost

Model	# Parameters	Training/epoch	Eval.	MAE
LSTM	186925	9.30 s	0.92 s	16.01
GRU	144045	8.93 s	0.88 s	15.31
DCRNN	372353	102.45 s	12.15 s	14.78
STGCN	732577	58.61 s	6.57 s	13.11
STG-NCDE	376284	263.58 s	20.24 s	14.58
STGODE	1613964	89.74 s	5.85 s	13.37
ASTGCN	988260	85.40 s	11.49 s	13.08
GMAN	380033	218.88 s	16.95 s	13.53
GWNET	350716	47.37 s	3.90 s	14.04
AGCRN	752730	35.53 s	4.44 s	14.70
MTGNN	718840	31.31 s	2.95 s	13.85
Multi-ATGCN (Spectral)	1987764	79.31 s	8.45 s	12.81
Multi-ATGCN	1603463	46.30 s	5.78 s	12.24

model performance, which confirms the importance of including contextual information. Last, the two types of zone-specific processing, i.e., zone-based normalization and zone-specific parameters, both moderately enhance the model performance, indicating the necessity of involving local individual details.

Complexity Analysis: To evaluate the computational cost, the number of parameters and training time of Multi-ATGCN are compared with other baselines running on the D.C. data for 24-hour forecasting. As shown in Table 4, Multi-ATGCN has the second most parameters as a sacrifice for extracting multiple temporal heads, integrating various auxiliary information, constructing multi-view graphs, and learning zone-specific patterns. However, the training and evaluation speed of Multi-ATGCN is comparable to many state-of-the-art models, since it generates all predictions at once using the 2D CNN instead of iteratively using the S2S framework (e.g., DCRNN). Considering the salient performance improvement and the relatively fast computation speed, the overall computational cost of Multi-ATGCN is moderate. We also consider replacing Chebyshev polynomials with classical Spectral GCN to test the effectiveness of Chebyshev polynomials. The results show that both the number of parameters and training and evaluation time increase when using Spectral GCN.

Graph Construction: To validate the effectiveness of the proposed graph construction module, this study constructs a set of different adjacency matrices and reports their performance in Table 5. Experiments are run on D.C. data for 3-hour prediction (See Appendix E for Baltimore data). Seven types of single adjacency matrices are compared. The identity matrix ($\boldsymbol{I_N}$) is used as the baseline. Functional similarity ($\boldsymbol{A_F}$), OD volume ($\boldsymbol{A_{OD}}$), and distance closeness ($\boldsymbol{A_D}$) are predefined matrices. In addition, two types of adaptive matrices are compared. The unidirectional method is from Eq. (10) (SoftMax(ReLU($\vec{E}_1\vec{E}_2$))), while the bidirectional method simplifies it by assuming a symmetric matrix (SoftMax(ReLU($\vec{E}_1\vec{E}_1^T$))). Last, the multi-view method ($\widehat{\boldsymbol{A}}$) is the fusion of all adjacency matrices.

Table 5: Comparison of different adjacency matrices

	MAE	RMSE	MAPE
	Mean ± st.d.(Δ%)	Mean ± st.d.(Δ%)	Mean ± st.d.(Δ%)
Identity	10.849 ± 0.221(5.62%)	30.110 ± 0.606(6.66%)	0.250 ± 0.004(6.84%)
Distance	10.845 ± 0.202(5.58%)	30.005 ± 0.596(6.28%)	0.248 ± 0.003(5.86%)
Bi-Adaptive	10.702 ± 0.174(4.19%)	29.421 ± 0.592(4.22%)	0.245 ± 0.002(4.49%)
Uni-Adaptive	10.691 ± 0.132(4.08%)	29.451 ± 0.534(4.32%)	0.244 ± 0.002(4.27%)
Distance+Uni-Adaptive	10.655 ± 0.130(3.73%)	29.311 ± 0.447(3.83%)	0.242 ± 0.002(3.47%)
OD	10.572 ± 0.118(2.92%)	29.034 ± 0.459(2.84%)	0.241 ± 0.002(2.99%)
OD+Uni-Adaptive	10.569 ± 0.122(2.89%)	29.011 ± 0.481(2.76%)	0.241 ± 0.001(3.08%)
Functional	10.527 ± 0.115(2.48%)	28.900 ± 0.462(2.37%)	0.240 ± 0.002(2.56%)
Functional+Uni-Adaptive	10.502 ± 0.117(2.24%)	28.771 ± 0.470(1.91%)	0.242 ± 0.001(3.42%)
Multi-view	**10.272 ± 0.102**	**28.231 ± 0.359**	**0.234 ± 0.001**

Besides the single adjacency matrices, this study also considers the combination between adaptive matrices and pre-defined matrices. As shown, the multi-view matrix achieves the lowest loss. The combination of the functional similarity matrix and the unidirectional adaptive matrix is the second-best, followed by the single functional similarity matrix and OD volume-related matrices. Although the two self-adaptive methods perform slightly worse than predefined methods, their performances are still remarkable even without any given prior knowledge. Also, incorporating the adaptive matrix into the pre-defined matrix can increase the overall performance, suggesting that the adaptive matrix can effectively capture hidden patterns that are not adequately represented by predefined measures alone. Last, the distance closeness matrix is only slightly better than the identity matrix, which implies that directly using distance to measure the zone connectivity may cannot explicitly capture the real graph structure.

To further compare different adjacency matrices, the spatial distributions of four adjacency matrices are depicted in Figure 8. Distance closeness is not plausible when geographic units are irregular since the distance between two zones is highly influenced by their area. Large zones, for example, the suburban areas, are more likely to be tagged as "isolated" since the distance from their centroids to others is inevitably long. However, the crowd inflow in these areas is not unrelated to other flows. Adjacency matrices measured by functional similarity and OD volume do not neglect these suburban areas, but the two measures also present highly different distributions. Last, the self-adaptive learning adjacency matrix is different from the other three predefined matrices. It presents more like a fusion of all of them, indicating that the real zone connectivity is complex and cannot be simply interpreted by one measure.

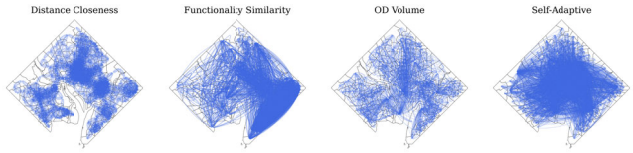


Figure 8: Adjacency matrices measured by (a) distance closeness, (b) functional similarity, (c) OD volume, and (d) self-adaptive in D.C. Only top 5% links are shown.

6 DISCUSSION ON APPLICATION SCENARIO

Our framework provides a solution to predict hourly crowd flow using MDLD and other related data, which can bring new directions to urban and traffic planning. One potential application is to improve travel demand models [21] (Appendix F). The high forecasting

accuracy and efficiency of Multi-ATGCN, along with continuously-collected MDLD, enable traffic engineers to model travel demand in a continuous manner. Meanwhile, data requirements are typically not more onerous for our models than for traditional travel demand models since the required data are quite similar.

When deploying the model to public systems, one main potential challenge is to meet the hourly update and execution frequency. The cloud computing services, such as Amazon Web Services (AWS) EMR and AWS Lambda, are feasible solutions to guarantee computational efficiency [13]. For example, all data storage, processing, and modeling can be finished on cloud services. The system will be scheduled hourly to fetch all related data and pass through the pre-trained Multi-ATGCN for forecasting. Multi-ATGCN will be trained and updated in an online manner by continuously including new information. In the future, we also consider the possibility of incorporating this framework with our current travel app[3] to accurately predict crowd flow to help travelers avoid congestion.

7 CONCLUSION

This study proposes a comprehensive GCN-based framework, the Multi-ATGCN, for citywide crowd inflow forecasting considering complex spatiotemporal dependency and heterogeneous external effects. By integrating a variety of deep learning techniques and spatiotemporal information, Multi-ATGCN performs strong flexibility, comparable efficiency, and superior performance in multi-step time series forecasting. Specifically, multiple temporal components are extracted to represent complex temporal dynamics. Multi-view self-adaptive adjacency matrices are constructed to comprehensively describe spatial structures. Parameter initialization and time sequence concatenation are further employed to learn from auxiliary variables. Last, all information is fused and passed through a zone-specific mix-hop GCRNN for jointly handling spatiotemporal dependency. Experiments on two real-world datasets show that Multi-ATGCN achieves state-of-the-art performance, and the advantages are more evident in data-sparse zones and long-horizon prediction. The ablation study further demonstrates the importance of different components in improving the model performance. In the future, a verification of more types of datasets is warranted to further test the model's generalizability.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems* 33 (2020), 17804–17815.

[3] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203* (2013).

[4] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).

[5] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[6] Jeongwhan Choi, Hwangyong Choi, Jeehyun Hwang, and Noseong Park. 2022. Graph neural controlled differential equations for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 6367–6374.

[7] Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. 2021. Spatial-temporal graph ode networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*. 364–373.

[8] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. 2019. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 3656–3663.

[9] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 922–929.

[10] Liangzhe Han, Bowen Du, Leilei Sun, Yanjie Fu, Yisheng Lv, and Hui Xiong. 2021. Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 547–555.

[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[12] Songhua Hu and Chenfeng Xiong. 2023. High-dimensional population inflow time series forecasting via an interpretable hierarchical transformer. *Transportation research part C: emerging technologies* 146 (2023), 103962.

[13] Songhua Hu, Chenfeng Xiong, Mofeng Yang, Hannah Younes, Weiyu Luo, and Lei Zhang. 2021. A big-data driven approach to analyzing and modeling human mobility trend under non-pharmaceutical interventions during COVID-19 pandemic. *Transportation Research Part C: Emerging Technologies* 124 (2021), 102955.

[14] Renhe Jiang, Xuan Song, Dou Huang, Xiaoya Song, Tianqi Xia, Zekun Cai, Zhaonan Wang, Kyoung-Sook Kim, and Ryosuke Shibasaki. 2019. Deepurbanevent: A system for predicting citywide crowd dynamics at big events. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2114–2122.

[15] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3128–3137.

[16] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[17] Duo Li, Jie Hu, Changhu Wang, Xiangtai Li, Qi She, Lei Zhu, Tong Zhang, and Qifeng Chen. 2021. Involution: Inverting the inherence of convolution for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12321–12330.

[18] Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-Tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. 2020. A system for massively parallel hyperparameter tuning. *Proceedings of Machine Learning and Systems* 2 (2020), 230–246.

[19] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2017. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*.

[20] Binbing Liao, Jingqing Zhang, Chao Wu, Douglas McIlwraith, Tong Chen, Shengwen Yang, Yike Guo, and Fei Wu. 2018. Deep sequence learning with auxiliary information for traffic prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 537–546.

[21] Michael G McNally. 2007. The four-step model. In *Handbook of transport modelling*. Emerald Group Publishing Limited.

[22] Zheyi Pan, Yuxuan Liang, Weifeng Wang, Yong Yu, Yu Zheng, and Junbo Zhang. 2019. Urban traffic prediction from spatio-temporal data using deep meta learning. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1720–1730.

[23] SafeGraph. 2020. SafeGraph Data for Academics. "https://www.safegraph.com/"

[24] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

[25] Martin Simonovsky and Nikos Komodakis. 2017. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3693–3702.

[26] Junkai Sun, Junbo Zhang, Qiaofei Li, Xiuwen Yi, Yuxuan Liang, and Yu Zheng. 2020. Predicting citywide crowd flows in irregular regions using multi-view graph convolutional networks. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[27] Michael E Wall, Andreas Rechtsteiner, and Luis M Rocha. 2003. Singular value decomposition and principal component analysis. *A practical approach to microarray data analysis* (2003), 91–109.

[28] Jingyuan Wang, Jiawei Jiang, Wenjun Jiang, Chao Li, and Wayne Xin Zhao. 2021. Libcity: An open library for traffic prediction. In *Proceedings of the 29th International Conference on Advances in Geographic Information Systems*. 145–148.

[29] Tao Wang, Songhua Hu, and Yuan Jiang. 2021. Predicting shared-car use and examining nonlinear effects using gradient boosting regression trees. *International Journal of Sustainable Transportation* 15, 12 (2021), 893–907.

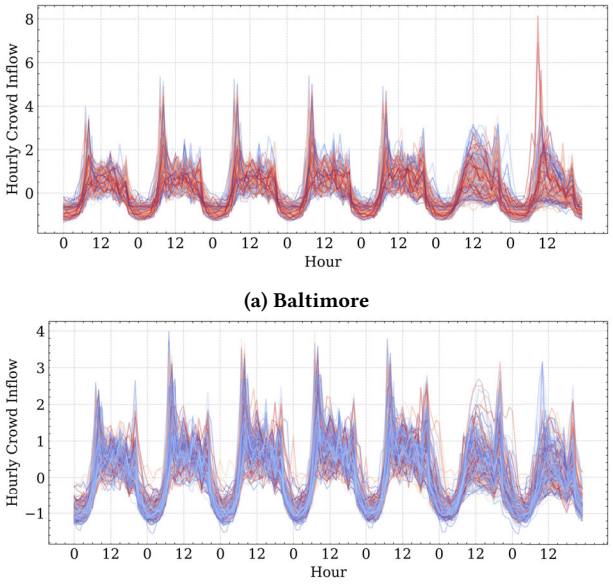
[3] A smartphone app operated in the DC-Maryland-Virginia area: https://incentrip.org/

[30] Yuandong Wang, Hongzhi Yin, Hongxu Chen, Tianyu Wo, Jie Xu, and Kai Zheng. 2019. Origin-destination matrix prediction via graph convolution: a new perspective of passenger demand modeling. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1227–1235.
[31] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.
[32] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 753–763.
[33] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121* (2019).
[34] Yiqun Xie, Erhu He, Xiaowei Jia, Weiye Chen, Sergii Skakun, Han Bao, Zhe Jiang, Rahul Ghosh, and Praveen Ravirathinam. 2022. Fairness by "Where": A Statistically-Robust and Model-Agnostic Bi-Level Learning Framework. (2022).
[35] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.
[36] Junchen Ye, Leilei Sun, Bowen Du, Yanjie Fu, Xinran Tong, and Hui Xiong. 2019. Co-prediction of multiple transportation demands based on deep spatio-temporal neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 305–313.
[37] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875* (2017).
[38] Zhuoning Yuan, Xun Zhou, and Tianbao Yang. 2018. Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 984–992.
[39] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Thirty-first AAAI conference on artificial intelligence*.
[40] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. 2020. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 1234–1241.

8 APPENDIX

A CROWD FLOW TIME SEIRES

The time series of weekly-average hourly crowd inflow in the two study areas are depicted in Figure 9.



(a) Baltimore

(b) D.C.

Figure 9: Normalized time series of census tract-level weekly-average hourly crowd inflow.

B IMPLEMENTATION DETAILS

The implementation details of all models are listed as follows:

1) FNN: A two-layer FNN with a hidden size of 128 and using the ReLU function between the two layers as activation.

2) LSTM/GRU: The LSTM and GRU are implemented in a sequence-to-sequence manner to recursively generate multi-step-ahead output. The encoder and decoder follow the same structure, each contains 2 layers of LSTM (GRU) with 64 hidden units. An FNN is applied to the output of the RNN at each time step to convert it to the final prediction. Models are trained using the teacher-forcing strategy with a ratio of 0.5. The learning rate is set to 0.01 with a decaying ratio of 0.1 in 5, 20, and 40 epochs.

3) STGCN: Two spatiotemporal convolutional (ST-Conv) blocks are stacked, followed by an output layer containing two temporal CNN and one FNN. The channels of three layers in ST-Conv blocks are (64, 32, 1) and (64, 32, 128), respectively. Both the graph convolution kernel size and temporal convolution kernel size are set to 3. Similar to our study, the Chebyshev polynomials approximation is used for graph convolution. The learning rate is set to 0.001 with a decaying ratio of 0.7 in every 5 epochs.

4) DCRNN: DCRNN is implemented in a S2S manner to recursively generate multi-step-ahead output. The encoder and decoder follow the same structure, each contains 2 layers of diffusion convolutional GRUs with 64 hidden units. The dual random walk approach is adopted for the diffusion process. The learning rate is set to 0.01 with a decaying ratio of 0.1 in 5, 20, and 40 epochs.

5) ASTGCN: ASTGCN builds three attention-based STGCN for three types of temporal heads (i.e., close, period, and trend) respectively, and fuses their outputs using a parametric-matrix-based weighting function. The hidden units for graph convolution and temporal convolution are set to 64 with a kernel size of 3. The learning rate is set to 0.0001.

6) GWNET: GWNET stacks 2 spatial-temporal layers. Each spatial-temporal layer is constructed by a graph convolution layer and a gated temporal convolution layer. The hidden units for all convolution networks are set to 32 with a kernel size of 2. 1×1 convolution with output channels of 256 is set for skip connection. Two 2D CNNs are stacked as the final output layers by first projecting the channels to 512 and then downsampling to the output dimension.

7) AGCRN: The AGCRN consists of an encoder and a 2D CNN which is used to replace the decoder. The encoder is constructed by 2 layers of adaptive zone-specific graph convolution GRU with 64 hidden units. The dimension of zone embedding is set as 10, and the order of Chebyshev polynomials is set as 2. The learning rate is set to 0.003 with a decaying ratio of 0.75 in 5, 15, 30, and 40 epochs.

8) GMAN: First, the node2vec algorithm is used to learn a zone embedding vector with a dimension of 64 and combined it with temporal embedding vectors. For node2vec, the number of random walks is set as 100 with 50 iterations. Then, an encoder and a decoder both with 2 ST-Attention blocks, one transform attention layer (number of attention heads is set as 2), and 2 FNNs are constructed to generate the final output. The learning rate is set to 0.001 with a decaying ratio of 0.7 when the loss does not decrease for 5 epochs.

9) MTGNN: In MTGNN, temporal convolution and graph convolution are interleaved with each other to capture temporal and spatial dependency respectively. The channel of convolutional layers is set as 32, and the number of layers is set as 3. The temporal inception layer consists of four filter sizes, viz. 1×2, 1×3, 1×6, and 1×7. The output module consists of two 1×1 convolution layers. MTGNN also includes a graph self-learning module with a zone embedding dimension of 40.

10) STGODE: The hidden dimensions of temporal dilation convolution blocks are set to 64, 32, 64, and 3 Spatial-Temporal Graph ODE blocks are contained in each layer. The regularized hyperparameter is set to 0.8. The thresholds of the spatial adjacency matrix are set to 10 and 0.5 respectively, and the threshold of the semantic adjacency matrix is set to 0.6. The model is trained using Adam optimizer with a learning rate of 0.01.

11) STG-NCDE: The order of Chebyshev polynomials is set to 2 and the zone embedding size is set to 8. The dimensionality of the hidden vector is set to 64. The learning rate is set to 0.001 and the weight decay is 0.001.

12) Multi-ATGCN: The Adam optimizer is employed to minimize the model loss with the learning rate decaying (Starting from 0.003, decaying by 75% once the number of epochs reaches 5, 10, 20, and 30 epochs). Each model is run 50 epochs and an early stop strategy with a patience of 10 is used by monitoring the loss in the validation set. The batch size is set as 16. Gradient clipping (maximum norm = 5) is performed during the training process to mitigate exploding gradients. Dropout (ratio = 0.1) is applied before the output layer.

C PARAMETER STUDY

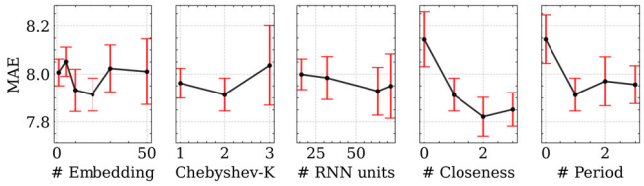
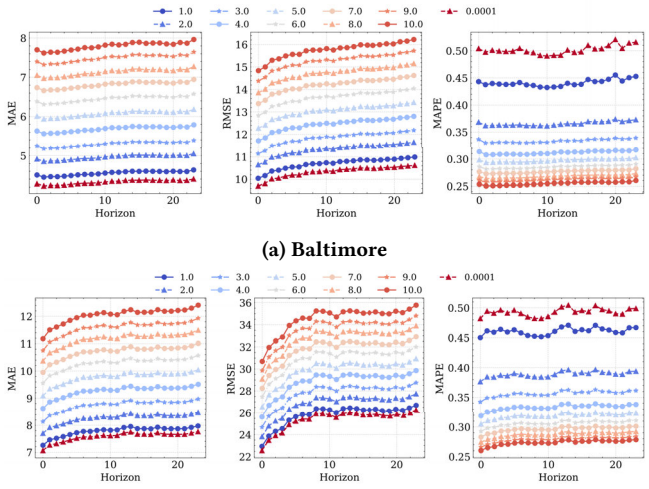


Figure 10: Influence of different core parameters on MAE.

A parameter study is conducted on five core hyperparameters of Multi-ATGCN, including the zone embedding dimension, orders of Chebyshev polynomials, # RNN hidden units, # closeness temporal heads, and # period temporal heads. Experiments are run on Baltimore data for 24-hour prediction and results are depicted in Figure 10. As shown, with the increase of each hyperparameter, the model loss decreases at first and then slightly rebounds. All these hyperparameters would increase the model complexity with the increase of their value. Hence, an excessively small hyperparameter would simplify the model and thus lead to underfitting. On the other hand, a large hyperparameter will significantly increase the parameter numbers, making the model harder to optimize and causing overfitting. Overall, it would be a good practice to find the most appropriate hyperparameter for each scenario for achieving the best performance.



(a) Baltimore

(b) Washington D.C.

Figure 11: Model performance under different lower bounds across prediction horizons.

D EFFECTS OF LOWER BOUND

We use three evaluation metrics to measure the performance of all models on the testing set:

$$MAPE = \frac{1}{|\Omega|}\sum_{i=1}^{N}\sum_{t=t_0}^{t_0+d_D-1}\frac{|\hat{y}_{i,t}-y_{i,t}|}{|y_{i,t}|}I(y_{i,t}\geq\varepsilon) \tag{25}$$

$$MAE = \frac{1}{|\Omega|}\sum_{i=1}^{N}\sum_{t=t_0}^{t_0+d_D-1}|\hat{y}_{i,t}-y_{i,t}|I(y_{i,t}\geq\varepsilon) \tag{26}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}\sum_{t=t_0}^{t_0+d_D-1}(\hat{y}_{i,t}-y_{i,t})^2}{|\Omega|}I(y_{i,t}\geq\varepsilon)} \tag{27}$$

where $y_{i,t}$ is the crowd inflow of census tract i at time t and $\hat{y}_{i,t}$ is its prediction; N is the number of census tracts; d_D is the length of the prediction window; Ω is the set of observations that meets $y_{i,t} \geq \varepsilon$; ε is the lower bound; $I(.)$ is the indicator, which is 1 when $y_{i,t} \geq \varepsilon$ and 0 otherwise.

The lower bound ε is to exclude extremely small values in the testing set, which would significantly affect the three evaluation metrics. The model performance varying across different ε at each horizon is shown in Figure 11. Overall, with the increase of the lower bound, the MAE and RMSE greatly increase while the MAPE significantly decreases. For example, changing the lower bound from 0.0001 to 10, the MAE for the Baltimore dataset increases from 4.326 to 7.874, the RMSE increases from 10.327 to 15.835, while the MAPE decreases from 0.502 to 0.255. Another takeaway message here is that the decreasing rate of MAPE is gradually flattening. Increasing the lower bound from 0.0001 to 2 would lead to a 13.6% drop in MAPE while increasing the lower bound from 2 to 4 only leads to another 5.4% drop.

E GRAPH CONSTRUCTION IN BALTIMORE

The section records the graph construction analysis using the Baltimore data for 24-hour prediction. Results are similar to D.C. data reporting in the main text.

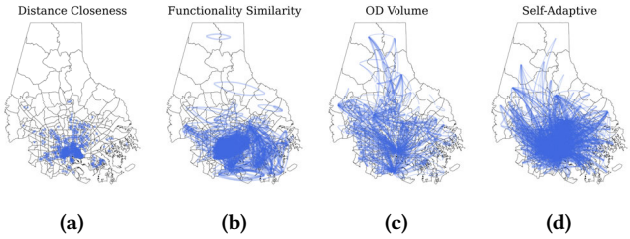


Figure 12: Adjacency matrices measured by (a) distance closeness, (b) functional similarity, (c) OD volume, and (d) self-adaptive in Baltimore. Only top 5% links are shown.

F EXAMPLE OF APPLICATION SCENARIO

Figure 13 shows an example of Multi-ATGCN-embedded travel demand modeling framework. Specifically, fine-grained crowd flow will be estimated and hourly updated via MDLD. Collected data will be aggregated to analysis zones to obtain travel Production-Attraction (PA) matrices. Combining with other external variables, Multi-ATGCN will be trained to forecast future PA matrices. Finally, outcomes will be fed into dynamic traffic assignment to generate citywide, road-level, time-dependent traffic volume and speed in the future.

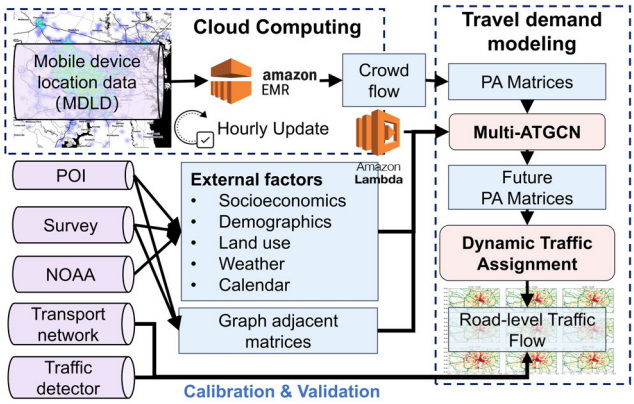


Figure 13: Example roles of Multi-ATGCN in travel demand model.