Journal of Electronics and Information Science (2024)
Clausius Scientific Press, Canada

DOI: 10.23977/jeis.2024.090208
ISSN 2371-9524 Vol. 9 Num. 2

A GNN Approach for Turn-Level Traffic Prediction: Dynamic Relation Awareness and Hypergraph Modeling

Haoyang Duan[1,a,*], Feihu Jiang[1,b]

[1]School of Computer Science and Technology, USTC, Hefei, China
[a]duanhaoyang@mail.ustc.edu.cn, [b]jiangfeihu@mail.ustc.edu.cn
[*]Corresponding author

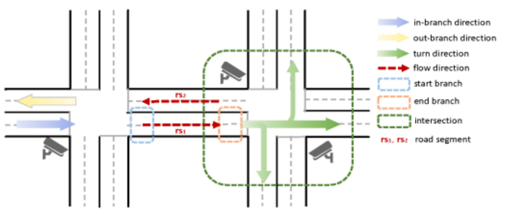
Keywords: Turn-level Traffic Flow Prediction, Relation-aware Diffusion Convolution, Hypergraph Learning

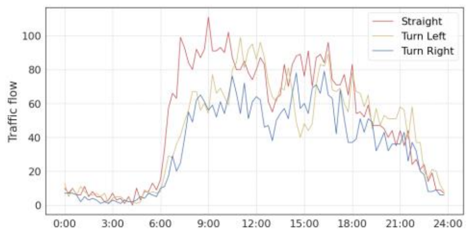
Abstract: It cannot be emphasized too much to predict traffic flow accurately in modern intelligent transportation systems. Though big progress has been made, few works focus on the turn-level traffic flow prediction, which is important to inspect fine-grained urban traffic dynamics closely. In this work, we develop a GNN (Graph Neural Network) approach built upon Dynamic Relation Awareness and Hypergraph modeling toward turn-level traffic flow prediction, namely DrahGNN. First, we construct a dynamic graph sequence where each snapshot denotes a turn-level traffic flow picture on top of a real-world road network. Second, we innovate a relation-aware spatiotemporal diffusion convolution network to capture road segments' differences and relatedness explicitly. Third, we construct a hypergraph in each time frame to capture high-order and manifold correlations between road segments and design an attentive two-stage message-passing mechanism for aggregating infor- mation from non-directly connected nodes. We conduct empirical studies on real-world data which demonstrate the effectiveness of our proposed framework.

1. Introduction

Traffic prediction, which encompasses the estimation of traffic flow volume and density, is pivotal in shaping intelligent urban environments. Within this realm, turn-level traffic prediction, which aims to forecast fine-grained traffic flow at distinct turn directions within the road network as shown in Figure 1(a), emerges as a critical endeavor and has been rarely studied. By providing valuable insights, it significantly contributes to the optimization of traffic signal control and the configuration of transportation networks, thereby facilitating the realization of intelligent cities [1, 2].

Recent advancements in graph neural networks (GNN) have demonstrated their superiority on the traffic flow prediction problem over traditional time series methods [3, 4], as well as other deep learning techniques such as CNN and RNN [5, 6, 7, 8, 9]. Though there are extensive studies ontraffic flow prediction, few focus on turn-level, which is more fine-grained and practical in real-world scenarios, like smart traffic signal control. Due to the dynamic complex spatiotemporal correlations between roads and the inherent difficulty of turn-level traffic prediction, existing methods still have some limitations.





(a) An illustrative example of the road network where the turn-level traffic is examined. (b) Traffic flow of different turn directions on a road segment during a day.

Figure 1: The diversity in turn-level traffic flow.

First, the correlations between roads change dynamically over time, but most existing GNN-based works [10, 11, 12] employ a static adjacent matrix to represent spatial dependencies between roads, which cannot reflect this dynamicity. Though some works attempt to construct a new graph structure that contains both spatial and temporal information [13, 14, 15], these manually predefined adjacency matrices may introduce bias, further reducing the generalizability of the model. Second, as shown in Figure 1(b), traffic flow distribution between different road segment pairs can have significant discrepancies in different turn directions and time segments in a day. However, few existing works take such inherent diversity into consideration. Third, the traffic interdependence on the road network is not limited to two adjacent road segments. For example, there are “hot routes” which attract more vehicles to the road network. A series of road segments that form these “hot routes” experience a larger traffic flow and may exhibit considerable correlations. Besides, some geographically distant road segments may also have dependencies because of similar urban functional areas. However, most existing works overlook these high-order and non-pairwise correlations.

Towards addressing the turn-level traffic flow prediction problem and the aforementioned challenges, we propose a novel GNN-based framework named DrahGNN. To accurately reflect dynamic and complex spatial dependencies between road segments, we construct a dynamic graph sequence based on the real-world road network and turn-level traffic flow. Further, we define a set of spatiotemporal aware relations to explicitly represent the diversity in turn-level traffic flow and based on them devise a novel multi-relational diffusion convolution. Moreover, to capture high-order and non-pairwise correlations, we construct a road hypergraph in each time slice and innovate an attentive hypergraph message passing mechanism. Our main contributions can be summarized as follows:

– We introduce a novel graph construction method that models road network as a dynamic graph sequence changing over time. Compared to previous works, our method requires much less prior knowledge and, therefore, enjoys promising generalizability.

– We design a new multi-relational diffusion convolution to capture complex correlations between road segments by explicitly exploiting spatiotemporal aware relations.

– To capture high-order and non-pairwise dependencies between road segments, we adaptively construct a road hypergraph in each time slice. Furthermore, an attentive two-stage hyperedge message passing mechanism incorporating relation-injection is proposed.

2. Preliminaries

2.1. Data Description

The collected data consists of the following two parts: (i) road network data; (ii) traffic flow data. As shown in Figure 1(a), the road network is organized hierarchically, where top-down components are intersections, branches, and lanes. Each intersection contains multiple branches which can be divided into two types according to their direction: in-branch entering the intersection and out-branch leaving the intersection. Each branch is composed of several lanes. Intersections are connected by road segments, and each road segment corresponds to a specific starting branch and an ending branch. In addition to the containing relations, the road network data also includes connectivity and turn directions which are straight, left turn, right turn and U turn between different intersections and branches.

Traffic flow data is collected by sensors deployed at end-branches of road segments. These sensors record the traffic flow count for each turn of that branch within a specific time interval(for example, five minutes). Combining this data with road network data, we can obtain the traffic flow number and turn direction between any two connected road segments for any given time period.

2.2. Problem Definition

We organize turn-level traffic flow data as a time-evolving traffic flow graph sequence $\mathcal{G}=\left\{\mathcal{G}^1,\mathcal{G}^2,\cdots,\mathcal{G}^T\right\}$, where T is the number of time slices. Each snapshot $\mathcal{G}^t=(\mathcal{V},\mathcal{E}^t,\mathcal{R},\mathcal{A}^t)$ is a multi-relational weighted directed graph. $\mathcal{V}$ denotes the shared node set with $|\mathcal{V}|=N$, which corresponds to the set of road segments in our scenario. $\mathcal{E}^t\subset\mathcal{E}$ denotes a set of weighted directed edges at time t, with which $\mathcal{E}$ denotes all possible links in road network. $\mathcal{R}$ denotes a set of spatial-temporal aware relations with $|\mathcal{R}|=P$. Each edge $(v_i,v_j)\in\mathcal{E}^t$ is labeled with a relation $r\in\mathcal{R}$, and represents that there exists traffic flow from node v_i to v_j in time slice t with its weight indicates the count of traffic flow. Note that $\left|\mathcal{E}^t\right|$ is dynamically changing over time. $\mathcal{A}^t\in\mathbb{R}^{N\times N}$ is adjacency matrix of $\mathcal{G}^t$. We denote initial node feature of $\mathcal{G}^t$ as $\mathbf{X}^t\in\mathbb{R}^{N\times F}$ where F is the dimension of node feature. We define our turn-level traffic flow prediction as a regression problem as follows:

Give a historical observed turn-level traffic flow graph sequence $\mathcal{G}=\left\{\mathcal{G}^1,\mathcal{G}^2,\cdots,\mathcal{G}^T\right\}$ with their corresponding node feature $\mathbf{X}^1,\mathbf{X}^2,\cdots,\mathbf{X}^T$, and road network data $\mathcal{D}$, we aim to learn a function $\mathcal{F}$ to forecast the weight on edges of graph $\mathcal{G}^{T+1}$ in next time slice(i.e., adjacency matrix $\mathcal{A}^{T+1}$):

$$\left\{\mathcal{G},\mathcal{D},\mathbf{X}^{1:T}\right\}\xrightarrow{\mathcal{F}}\mathcal{A}^{T+1}. \tag{1}$$

3. Methdology

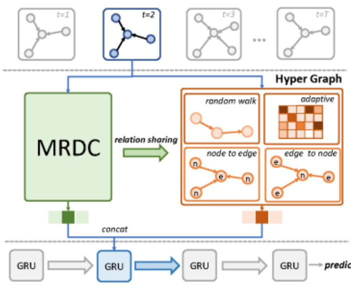


Figure 2: Framework overview of DrahGNN.

3.1. Multi-relational Diffusion Convolution

Figure 2 shows an overall architecture of our proposed DrahGNN framework. We first introduce multi-relational diffusion convolution mudole. This component aims to model the complex spatiotemporal correlations between road segments. As shown in Figure 1(b), the traffic flow presents diverse patterns under different turn directions and time segments. To capture such diversity, we propose to first define multiple turn-level spatiotemporal relations between segments, and then perform relation-aware aggregation on the road network.

The multiple relations between road segments are constructed from a joint spatiotemporal perspective. At the spatial dimension, we distinguish four turn directions of two connected road segments, which are straight, left turn, right turn and U turn. It can model the diversity of traffic patterns under different turns. Furthermore, for the temporal dimension, we evenly divided a day into n time segments, where different relations are built to model varying dependencies between road segments in different time segments. By integrating these two dimensions, we can define $4 \times n$ relations in $\mathcal{R}$ to capture diverse spatiotemporal traffic patterns.

Next, we design a multi-relational diffusion convolution (MRDC) to aggregate the traffic information, with the awareness of multiple spatiotemporal relations. Following [10], it collects messages from both upstream and downstream road segments in different orders based on diffusion convolution. However, rather than making no distinction among adjacent segments, our model processes their messages in a relation-specific way.

We initialize a representation $\mathbf{h}_r$ for each relation in $\mathcal{R}$: $\mathbf{h}_r \in \mathbb{R}^F, \forall r \in \mathcal{R}$. To explicitly incorporate multiple relations into diffusion convolution, we employ the multiplication operation to compose nodes and relations. Specifically,

$$\alpha(\mathbf{h}_v, \mathbf{h}_r) = \mathbf{h}_v \otimes \mathbf{h}_r, \tag{2}$$

where $\alpha : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R}^F$ is a composition function, $\otimes$ means element-wise production, $\mathbf{h}_v$ and $\mathbf{h}_r$ are embedding of node v and relation r, respectively.

We next give the recursive and accumulation procedures of multi-relational diffusion convolution. The update equation of the $k+1$ step of recursion can be given as:

$$\mathbf{h}_{v_{in}}^{k+1} = \sum_{(u,r)\in\mathcal{N}_{in}(v)} \frac{e_{u,v}}{deg_v^{in}} \alpha(\mathbf{h}_{u_{in}}^{k}, \mathbf{h}_r), \tag{3}$$

where $\mathbf{h}_{v_{in}}^{k+1}$ is in-embedding of node v in step $k+1$ which aggregates its incoming neighbors, $\mathcal{N}_{in}(v)$ is a set of direct neighbors of v for its incoming edges, edge $(u,v) \in \mathcal{E}^t$, $e_{u,v}$ is the edge weight from source node u to target node v, $deg_v^{in} = \sum_{(u,r)\in\mathcal{N}_{in}(v)} e_{u,v}$ is the in-degree of node v, $\mathbf{h}_{u_{in}}^{k}$ and $\mathbf{h}_r$ are in-embedding of node u in step k and embedding of relation r . Similarly, we can obtain the out-embedding of node v in step $k+1$ through the aggregation of its outgoing neighbors:

$$\mathbf{h}_{v_{out}}^{k+1} = \sum_{(w,r)\in\mathcal{N}_{out}(v)} \frac{e_{v,w}}{deg_v^{out}} \alpha(\mathbf{h}_{w_{out}}^{k}, \mathbf{h}_r). \tag{4}$$

With $\mathbf{h}_{v_{in}}^{0} = \mathbf{h}_{v_{out}}^{0} = \mathbf{x}_v$, where $\mathbf{x}_v$ is the initial representation of node v, the output embedding in this module of node v is accumulation of all recursion steps:

$$\mathbf{h}_v^{mrdc} = \sigma(\sum_{k=0}^{K}(\boldsymbol{\theta}_{0,k}\mathbf{h}_{v_{in}}^{k} + \boldsymbol{\theta}_{1,k}\mathbf{h}_{v_{out}}^{k})), v \in \mathcal{V}, \tag{5}$$

where σ is the activation function, K is the diffusion step, and $\boldsymbol{\theta}_{0,k}$ and $\boldsymbol{\theta}_{1,k} \in \mathbb{R}^{F\times F_1}$ are two trainable weight matrix. These parameters are shared across each time slice. Further, we output the relation embedding as:

$$\mathbf{z}_r = \mathbf{W}_{rel}\mathbf{h}_r, r \in \mathcal{R}, \tag{6}$$

where $\mathbf{W}_{rel} \in \mathbb{R}^{F\times F_1}$ is a trainable weight matrix. The embedding of relations is transformed into the same space as the node embedding, and is to be used in the subsequent modules.

3.2. Road Hypergraph Learning

In the scenario of turn-level traffic flow prediction, the long-range interactions on the road network can have effects on the traffic flow from a global perspective. However, GNNs may fall short in capturing non-pairwise relationships between road segments, while the over-smoothing problem [16] poses a significant challenge in representing high-order correlations. To overcome this limitation, we propose the integration of a hypergraph attention network at the road level. We first provide a definition of the road hypergraph for a road network. Then, we devise the adaptive hyperedge generation scheme in a hybrid manner to comprehensively represent the non-pairwise structure. Furthermore, the attentive hypergraph message passing process is proposed for high-order context learning.

Definition (Road Hypergraph.) A road hypergraph can be defined as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{S})$, where $\mathcal{V}$ is the set of road segment nodes with $|\mathcal{V}| = N$ and $\mathcal{S}$ represents the set of hyperedges with $|\mathcal{S}| = M$. A hyperedge is referred to as multi road segments and can be viewed as a subset of $\mathcal{V}$, which can contain any number of nodes. The connectivity relationship between nodes and hyperedges can be represented as an incident matrix $\mathbf{T} \in \{0,1\}^{N\times M}$. $\mathbf{T}_{ij} = 1$ means that hyperedge $s_j \in \mathcal{S}$ contains node $v_i \in \mathcal{V}$.

3.2.1. Road Hyperedge Generation

To model "hot routes" explained in Section 1, we incorporate random walks on a weighted directed graph to construct hyperedges. A random walk serves as a dynamic process initiated from an arbitrary node within the graph, progressively moving towards downstream neighbors at each step until a predefined number of steps is reached or no further downstream neighbors can be found. The ordered sequence of nodes encountered forms a random walk sequence. In each step, the transition probability is calculated based on the edge weights. We sample a fixed number of c distinct random walk sequences for each node on graph $\mathcal{G}^t$. Consider each sequence as a hyperedge, we get $M_1 = N \times c$ hyperedges and a incident matrix $\mathbf{T}_1 \in \mathbb{R}^{N\times M_1}$.

As explained in Section 1, road segments that are not geographically close to each other may still have correlations. To capture these implicit dependencies in the latent space, we adaptively learn a set of hyperedges inspired by [11,17]. Specifically, we initialize a node embedding $\mathbf{E}_v \in \mathbb{R}^{N\times D}$ and a hyperedge embedding $\mathbf{E}_h \in \mathbb{R}^{M_2\times D}$. Then the adaptive hyperedges can be learned by:

$$\mathbf{T}_2 = \mathrm{ReLU}(\tanh(\mathbf{E}_v\mathbf{E}_h^{\mathrm{T}})), \tag{7}$$

where $\mathbf{E}_h^{\mathrm{T}}$ means the transpose of $\mathbf{E}_h$. Then we set the first p largest elements in each column of matrix $\mathbf{T}_2$ to 1, and the remaining elements to 0. $\mathbf{E}_v$ and $\mathbf{E}_h$ will be trained during model training. Thus we get another incident matrix $\mathbf{T}_2 \in \mathbb{R}^{N \times M_2}$. The final incident matrix of $\mathcal{G}^t$ is the concatenation of $\mathbf{T}_1$ and $\mathbf{T}_2$:

$$\mathbf{T} = [\mathbf{T}_1 \parallel \mathbf{T}_2], \tag{8}$$

where $\parallel$ means the concatenation operation, $\mathbf{T} \in \mathbb{R}^{N \times M}$ where $M = M_1 + M_2$.

3.2.2. Attentive Hypergraph Message Passing

After obtaining the incident matrix, we further perform the attentive message passing mechanism in a node-hyperedge interactive manner.

Node $\rightarrow$ Hyperedge Aggregation. The connections from road segment nodes to the same hyperedge can be regarded as implicit distant-range relations. Specifically, we initialize hyperedge representation $\mathbf{h}_s^{init}$ using the relation embedding $\mathbf{Z}_r$ and a learnable weight $\boldsymbol{\beta} \in \mathbb{R}^{1 \times P}$: $\mathbf{h}_s^{init} = \boldsymbol{\beta}\mathbf{Z}_r$, which incorporates the relational context for hypergraph learning. Considering the different "hotness" of each node in a hyperedge, the relation-injected aggregation from road segment nodes to the hyperedge can be expressed as:

$$\begin{aligned}\mathbf{h}_s &= \sum_{v \in s} \gamma_v \mathbf{W}_1 \cdot (\mathbf{x}_v \otimes \boldsymbol{\beta}\mathbf{Z}_r),\\ \gamma_v &= \frac{deg_v^{in} + deg_v^{out}}{\sum_{e \in \mathcal{E}} w_e},\end{aligned} \tag{9}$$

where $\mathbf{h}_s$ and $\mathbf{x}_v$ are the embeddings of hyperedge s and node v, respectively, $v \in s$ means that node v is connected to hyperedge s, w_e is the weight of the edge e, $\otimes$ is the multiplication operation and $\mathbf{W}_1$ is a trainable weight matrix.

Hyperedge$\rightarrow$Node Attention. After getting all hyperedge embeddings, we conduct a multi-head hyperedge-to-node attention mechanism, which considers the importance of road substructures and propagates hyperedges back to nodes with different weights:

$$\mathbf{h}_v^b = \sigma(\sum_{s \in \mathcal{E}_v} \lambda_{vs} \mathbf{W}_2 \mathbf{h}_s), \tag{10}$$

where $\mathbf{h}_v^b$ is the output embedding of node v in the b-th attention head, σ is the activation function, $\mathcal{E}_v$ is the set of hyperedges that connected to node v, λ_{vs} is the attention coefficient between node v and hyperedge s, $\mathbf{h}_s$ is the embedding of hyperedge s obtained from the node-to-hyperedge stage, $\mathbf{W}_2$ is another trainable weight matrix. λ_{vs} can be computed as follows:

$$\begin{aligned}\lambda_{vs} &= \frac{\exp(\mathbf{a}^{\mathrm{T}}\mathbf{l}_s)}{\sum_{t \in \mathcal{E}_v} \exp(\mathbf{a}^{\mathrm{T}}\mathbf{l}_t)},\\ \mathbf{l}_s &= \mathrm{LeakyReLU}([\mathbf{W}_2\mathbf{h}_s \parallel \mathbf{W}_1\mathbf{x}_v])\end{aligned} \tag{11}$$

where $\mathbf{a}^{\mathrm{T}}$ is a trainable attention weight vector. We concatenate node embeddings from different attention heads to form the final node representation in this module:

$$\mathbf{h}_v^{hyper} = \left[\mathbf{h}_v^1 \parallel \mathbf{h}_v^2 \parallel \cdots \parallel \mathbf{h}_v^B\right], v \in \mathcal{V}, \tag{12}$$

where B is the number of attention heads.

3.3. Prediction Module

After getting node embeddings from the MRDC module and hypergraph learning module, we concatenate these two embeddings in each time slice to get the final spatial embedding of each node:

$$\mathbf{h}_{v_t}^{sp} = \left[\mathbf{h}_{v_t}^{mrdc} \parallel \mathbf{h}_{v_t}^{hyper}\right], v \in \mathcal{V} \tag{13}$$

where $\mathbf{h}_{v_t}^{sp}$, $\mathbf{h}_{v_t}^{mrdc}$ and $\mathbf{h}_{v_t}^{hyper}$ are final spatial embedding, multi-relational diffusion convolution embedding and hypergraph embedding of node v in time slice t, respectively. Thus, for each node in $\mathcal{V}$, we get a sequence of embedding vector $\left\{\mathbf{h}_t^{sp}\right\}_{t=1}^{T}$. We take these sequences as the input of a gated recurrent unit (GRU)[18], which is a simple yet powerful variant of RNN, and take the hidden state at last time slice T as the final embedding of each node which contains both spatial and temporal information:

$$\mathbf{h}_v^{final} = \mathrm{GRU}(\left\{\mathbf{h}_t^{sp}\right\}_{t=1}^{T}), \tag{14}$$

where $\mathbf{h}_v^{final}$ is the final embedding of node v . We use this final embedding and relation embedding to predict adjacent matrix $\mathcal{A}^{T+1}$ of the next time slice:

$$\mathcal{A}_{uv}^{T+1} = \mathrm{sigmoid}(\mathrm{Linear}(\mathbf{h}_u^{final} \otimes \mathbf{h}_r \otimes \mathbf{h}_v^{final})), \tag{15}$$

where $\mathbf{h}_u^{final}$ and $\mathbf{h}_v^{final}$ are the final embeddings of node u and node v, respectively. The relation embedding between node u and node v is denoted as $\mathbf{h}_r$. $\mathcal{A}_{uv}^{T+1}$ is the corresponding value in $\mathcal{A}^{T+1}$, $\otimes$ is the element-wise production, $\mathrm{Linear}(\cdot)$ is a linear transformation which transforms the vector to a scalar score.

Finally, we adopt Mean Absolute Error (MAE) as loss function.

4. Experiments

4.1. Experimental Setup

Table 1: Statistics of Datasets.

Description	DatasetA	DatasetB
# road segments	724	1043
# intersections	218	317
# time slices	1152	2976
# average links	1018	973
# possible existing links	1654	2182
time range	12/14/2021 - 12/25/2021	1/1/2023 - 1/31/2023
average traffic flow	32.77	14.89
max traffic flow	130	80

Dataset. We evaluate our DrahGNN framework on two real-world datasets. The first dataset, which we called datasetA, contains 12 days of traffic flow (from December 14, 2021, to December 25, 2021) in a city in China. The second dataset called datasetB, contains 31 days of traffic flow (from January 1st, 2023, to January 31, 2023) in another city in China. Both datasets are recorded with a 15-minute interval. The detailed statistics are listed in Table 1.

In our experiments, we set the duration of the time slice and input length to 15 minutes and 6, respectively. Min-max normalization is used to scale the traffic flow values into [0, 1].

Baselines. We compared DrahGNN with following baselines:

– **HA**[19] uses the average value of historical observations as the prediction value of future time.

– **ARIMA**[3] is a classical time series model for predicting future values.

– **LSTM** [20] is powerful variant of RNN.

– **DCRNN**[10] defines diffusion convolution and incorporates it into a GRU.

– **STGCN**[12] combines GCN with a 1D gated temporal convolution.

– **Graph WaveNet**[11] learns an adaptive adjacent matrix and combines GCN with temporal convolution.

– **ASTGCN**[21] incorporates spatial-temporal attention mechanism and fuse attention matrix with GCN.

– **STSGCN**[15] constructs a localized spatial-temporal graph using spatial graphs at consecutive time steps to capture localized spatial-temporal correlations synchronously.

– **STFGNN** [14] incorporates DTW [3] technique and then constructs a spatial-temporal fusion graph, where this static adjacent matrix involves temporal information.

– **DSTAGNN**[13] utilizes Wasserstein distance [18] and spatial-temporal at tention mechanism to capture dynamic spatial-temporal correlations.

Implementation Details. We split the dataset into training set, validation set and test set as 7:1:2, and the hyper-parameters for all models are set according to the performance on the validation set. In DrahGNN we set the diffusion step $K = 5$ in MRDC. The number of time segments divided in each day is $n = 12$, i.e., each time segment equals 2 hours and contains 8 time slices. In hypergraph construction, the number of distinct random walk and the walk length sampled per node is 20 and 15, respectively. Besides, we build 200 adaptive hyperedges, each of which contains 100 nodes. And the number of attention heads in the hyperedge-to-node stage is 4. We adopt the AdamW optimizer to train our model, with an initial learning rate set to 0.001, and 0.1 decay factor for every 18 epochs. Additionally, an early stopping strategy is employed.

Evaluation Protocol. We adopt three widely used evaluation metrics to evaluate the performance of the model: mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean squared error (RMSE). Note that for MAE and RMSE, only possibly existing edges are computed, and for MAPE, only edges with non-zero ground truth values are computed.

4.2. Overall Performance

Table 2 and Table 3 show the overall performance comparison between our DrahGNN and different baselines on two datasets. The results indicate that DrahGNN surpasses other baselines in terms of overall performance for turn-level traffic flow prediction. Furthermore, we have the following findings: (i) Deep learning-based methods perform better than statistical methods, i.e., HA and ARIMA. This is because statistical methods only forecast future traffic flow using a simple statistical approximation of historical observations, which cannot capture complex correlations between road segments. (ii) GNN-based methods are better than other methods (i.e., HA, ARIMA and LSTM). This implies that utilizing graph structure to capture spatial dependencies is critical in traffic flow prediction problems. (iii) Among GNN-based models, those using a static, geometric distance-based adjacent matrix (i.e., DCRNN, STGCN, ASTGCN and Graph Wavenet) perform worse than those incorporating a computed adjacent matrix that contains spatial-temporal information (except STFGNN on datasetB). This reveals the importance of an accurate and dynamic adjacent matrix in traffic flow prediction. The relatively poor performance of Graph Wavenet reveals its drawbacks: it cannot simultaneously stack spatial-temporal layers and expand the receptive field of one-dimensional convolutions. (iv) Due to the Chinese New Year which included in time span of datasetB, the traffic flow in this dataset is sparser and the values of traffic flow are smaller. These pose a greater challenge and difficulty for the performance of the prediction models. Under this situation, the performance of STFGNN significantly decreases, indicating a drawback of STFGNN: the DTW[22] technique it uses struggles to capture correlations between high-dimensional time series in sparse data scenarios. (v) Though DSTAGNN slightly outperforms DrahGNN in terms of the MAPE metric on datasetA, it performs considerably poorly in terms of MAE and RMSE, which indicates its ability to predict non-zero values is better than that of zero values. This is because the Wasserstein distance [18] it employs cannot capture the diversity of traffic flow in different turn directions. In terms of overall performance, DrahGNN is more suitable for turn-level traffic flow prediction.

Table 2: Performance comparison of turn-level traffic flow prediction of our DrahGNN and baselines on datasetA.

Method	**HA**	**ARIMA**	**LSTM**	**DCRNN**	**STGCN**	**GWN**	**ASTGCN**	**DSTAGNN**	**STSGCN**	**STFGNN**	**DrahGNN**
Category	Statistical	Statistical	RNN	GNN	GNN	GNN	GNN	GNN	GNN	GNN	GNN
MAE	4.2209	3.5742	3.3498	3.2653	3.1255	3.4969	3.0073	3.3301	2.9281	2.9691	**2.4652**
RMSE	9.5449	7.3675	6.3755	6.5509	5.9065	7.3063	6.0469	7.1137	5.9994	5.9431	**4.7154**
MAPE(%)	45.82	44.56	39.97	43.81	38.81	45.57	35.69	**32.17**	34.62	38.01	33.42

Table 3: Performance comparison of turn-level traffic flow prediction of our DrahGNN and baselines on datasetB.

Method	**HA**	**ARIMA**	**LSTM**	**DCRNN**	**STGCN**	**GWN**	**ASTGCN**	**DSTAGNN**	**STSGCN**	**STFGNN**	**DrahGNN**
Category	Statistical	Statistical	RNN	GNN	GNN	GNN	GNN	GNN	GNN	GNN	GNN
MAE	2.3005	2.0172	1.8995	1.8701	1.8493	1.9895	1.7977	1.8824	1.7721	3.6867	**1.6682**
RMSE	5.6091	4.2309	4.4546	4.2087	4.0628	4.7537	4.0392	4.5847	4.1037	8.0579	**3.4039**
MAPE(%)	61.69	58.12	98.07	56.05	56.01	58.23	56.01	58.19	56.13	73.35	**55.62**

4.3. Ablation Study

To further validate the effectiveness of each design of our model, we further compare DrahGNN with its two variants on dataset A:

− **w/o hyper:** It removes the entire hypergraph module.

− **w/o rel & hyper:** It removes the entire hypergraph module and relation mechanism, where the model only contains the vanilla diffusion convolution module.

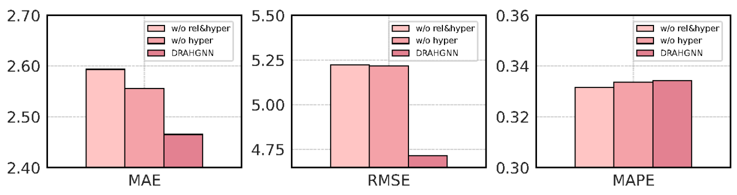


Figure 3: Ablation experimental results.

As we can see from the comparing results in Figure 3, DrahGNN performs better than its variants in terms of MAE and RMSE, which demonstrates the effectiveness of each component of DrahGNN. However, in terms of the MAPE metric, the performance of DrahGNN did not show a significant improvement and even slightly decreased. This is because when calculating MAPE, we only consider the edges where the ground truth values are non-zero. This implies that the model has improved its prediction accuracy for zero values without sacrificing overall performance.

4.4. Parameter Sensitivity

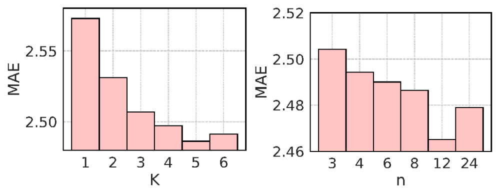


Figure 4: Impact of different diffusion steps K and number of time segments n.

In order to analyze the impact of different diffusion steps K in MRDC and the number of time segments n in a day, we conducted experiments over these two parameters on datasetA. The results are shown in Figure 4. Specifically, in terms of K, we set it varying from 1 to 6, and observe the performance of model. We observed that as K increases, the MAE on the test set gradually decreases, indicating an improvement in model performance. This is reasonable because K can be viewed as the receptive field of diffusion convolution. A larger K allows the model to aggregate information from neighbors at higher order, thereby capturing more complex spatial correlations. However, with K continues to increase, we observed a decline in model performance. Similar to the over-smoothing issue caused by stacking too many layers in GCN, this suggests that we cannot increase K without limitations. In terms of n, we vary it in [3, 4, 6, 8, 12, 24], as can be seen the performance of model improves with n increasing. But when $n = 24$, the model performance decreases which implies that it is not advisable to divide time segments into excessively fine granularity.

5. Related Work

5.1. Graph Based Traffic Forecasting

Recently, GNN has been widely used in traffic forecasting due to its ability of representing non-euclidean data. Specifically, DCRNN[10] models traffic flow as a diffusion process and fuses this process into a RNN-based sequence to sequence framework. STGCN[12] exploits GCN[23] and temporal convolution to represent spatial-temporal simultaneously. However, these methods capture spatial dependencies only using a static adjacent matrix defined based on geographical distance, which may be imprecise and cannot represent dynamic dependencies. In order to explore spatial dependencies beyond geographical distance, Graph Wavenet[11] and AGCRN[24] propose to learn hidden graph structure using trainable node embedding.

There are also some methods of studying a new graph structure that reflects complex spatial-temporal dependencies between roads more precisely. STFGNN [14] exploits DTW technique to compute the similarity between time sequences of nodes to construct a spatial-temporal fusion graph. DSTAGNN[13] emerges Wasserstein distance[25] to measure differences of traffic flow distributions in different roads and construct a new graph to incorporate the spatial-temporal attention mechanism. Nevertheless, none of these methods focus on turn-level traffic forecasting, and neglect the global, non-pairwise correlation between road segments.

5.2. Multi-Relational Graph Convolution

In spite of great success in modeling graph-structured data, GCN[23] cannot be trivially applied in directed multi-relational graphs, which are in a more general form and each edge has a label indicating the relation between source and target node[26]. RGCN[27] extends GCN by using distinct weights for different relations, and addresses the over-fitting issue by regularizing weights with basis and block-diagonal decomposition. CompGCN [26] learns representations for both nodes and relations and utilizes a variety of entity-relation composition operators to distinguish specific relations. However, these methods are all extensions of GCN, which cannot be trivially used for weighted directed graphs.

5.3. Hypergraph Neural Networks

Graphs can solely model pairwise relationships between nodes, and most GNNs suffer from the problem of over-smoothing. To address these issues, Zhou et al.[28] proposed a hypergraph where a hyperedge can contain an arbitrary number of nodes, which can represent high-order and non-pairwise relationships. Feng et al.[29] propose the hypergraph neural networks (HGNN), which performs node-edge-node transformations and can be seen as a generalization of spectral convolution to hypergraph. Ding et al.[30] construct a hypergraph using sequential and semantic hyperedges, then propose a dual attention mechanism to perform inductive text classification. These works have demonstrated the superiority of hypergraphs in capturing higher-order and non-pairwise relationships.

6. Conclusion

In this paper, we investigated the turn-level traffic flow prediction problem and proposed a novel GNN approach built upon Dynamic Relation Awareness and Hypergraph modeling (DrahGNN) to solve it. We constructed a dynamic graph sequence that can represent precise and dynamic correlations between road segments. In order to capture the diversity in turn-level traffic flow, we defined a set of spatiotemporal aware relations, based on which we designed a multi-relational diffusion convolution. Furthermore, to extract high-order and non-pairwise correlations between road segments, we constructed a hypergraph in each time slice and then designed an attentive two-stage hypergraph message-passing mechanism that incorporates relation injection. Extensive experiments on real-world datasets demonstrate DrahGNN's effectiveness and performance superiority over state-of-the-art baselines.

References

[1] Fang, M., Tang, L., Yang, X., Chen, Y., Li, C., Li, Q.: Ftpg: A fine-grained traffic prediction method with graph attention network using big trace data. IEEE Transactions on Intelligent Transportation Systems 23(6), 5163–5175 (2021).

[2] Kan, Z., Tang, L., Kwan, M.P., Ren, C., Liu, D., Li, Q.: Traffic congestion analysis at the turn level using taxis' gps trajectory data. Computers, Environment and Urban Systems 74, 229–243 (2019).

[3] Liu, W., Zheng, Y., Chawla, S., Yuan, J., Xing, X.: Discovering spatio-temporal causal interactions in traffic data streams. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 1010–1018 (2011).

[4] Lippi, M., Bertini, M., Frasconi, P.: Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. IEEE Transactions on Intelligent Transportation Systems 14(2), 871–882 (2013).

[5] Li, Y., Shahabi, C.: A brief overview of machine learning methods for short-term traffic forecasting and future directions. Sigspatial Special 10(1), 3–9 (2018).

[6] Yao, H., Tang, X., Wei, H., Zheng, G., Li, Z.: Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 5668–5675 (2019).

[7] Yao, H., Wu, F., Ke, J., Tang, X., Jia, Y., Lu, S., Gong, P., Ye, J., Li, Z.: Deep multi-view spatial-temporal network for taxi demand prediction. In: Proceedings of the AAAI conference on artificial intelligence. vol. 32 (2018).

[8] Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for citywide crowd flows prediction. In:

Proceedings of the AAAI conference on artificial intelligence. vol. 31 (2017).
[9] Zhang, J., Zheng, Y., Qi, D., Li, R., Yi, X.: Dnn-based prediction model for spatiotemporal data. In: Proceedings of the 24th ACM SIGSPATIAL international conference on advances in geographic information systems. pp. 1–4 (2016).
[10] Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint arXiv:1707.01926 (2017).
[11] Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph wavenet for deep spatialtemporal graph modeling. arXiv preprint arXiv:1906.00121 (2019).
[12] Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875 (2017).
[13] Lan, S., Ma, Y., Huang, W., Wang, W., Yang, H., Li, P.: Dstagnn: Dynamic spatialtemporal aware graph neural network for traffic flow forecasting. In: International Conference on Machine Learning. pp. 11906–11917. PMLR (2022).
[14] Li, M., Zhu, Z.: Spatial-temporal fusion graph neural networks for traffic flow forecasting. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 4189–4196 (2021).
[15] 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. vol. 34, pp. 914– 921 (2020).
[16] Rusch, T.K., Bronstein, M.M., Mishra, S.: A survey on oversmoothing in graph neural networks. arXiv preprint arXiv:2303.10993 (2023).
[17] Wu, J., Qi, Q., Wang, J., Sun, H., Wu, Z., Zhuang, Z., Liao, J.: Not only pairwise relationships: Fine-grained relational modeling for multivariate time series forecasting. In: IJCAI (2023).
[18] Cho, K., Van Merri¨enboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014).
[19] Hamilton, J.D.: Time series analysis. Princeton university press (2020).
[20] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997).
[21] 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. vol. 33, pp. 922–929 (2019).
[22] Deng, J.J., Leung, C.H.: Dynamic time warping for music retrieval using time series modeling of musical emotions. IEEE transactions on affective computing 6(2), 137–151 (2015).
[23] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016).
[24] Maganioti, A.E., Chrissanthi, H.D., Charalabos, P.C., Andreas, R.D., George, P.N. and Christos, C.N. (2010) Cointegration of Event-Related Potential (ERP) Signals in Experiments with Different Electromagnetic Field (EMF) Conditions. Health, 2, 400-406.
[25] Panaretos, V.M., Zemel, Y.: Statistical aspects of wasserstein distances. Annual review of statistics and its application 6, 405–431 (2019).
[26] Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.: Composition-based multirelational graph convolutional networks. arXiv preprint arXiv:1911.03082 (2019).
[27] Schlichtkrull, M., Kipf, T.N., Bloem, P., Van Den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15. pp. 593–607. Springer (2018).
[28] Zhou, D., Huang, J., Sch¨olkopf, B.: Learning with hypergraphs: Clustering, classification, and embedding. Advances in neural information processing systems 19 (2006).
[29] Feng, Y., You, H., Zhang, Z., Ji, R., Gao, Y.: Hypergraph neural networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 3558– 3565 (2019).
[30] Ding, K., Wang, J., Li, J., Li, D., Liu, H.: Be more with less: Hypergraph attention networks for inductive text classification. arXiv preprint arXiv:2011.00387 (2020).