Dynamic Graph Filters Networks: A Gray-box Model for Multistep Traffic Forecasting

Guopeng LI
Civil Engineering and Geosciences
Delft University of Technology
Delft, the Netherlands 2628 CN
Email: G.Li-5@tudelft.nl

Victor L. Knoop
Civil Engineering and Geosciences
Delft University of Technology
Email: V.L.Knoop@tudelft.nl

Hans van Lint
Civil Engineering and Geosciences
Delft University of Technology
Email: J.W.C.vanLint@tudelft.nl

Abstract— **Short-term traffic forecasting is one of the key functions in Intelligent Transportation System (ITS). Recently, deep learning is drawing more attention in this field. However, how to develop a deep learning based traffic forecasting model that can dynamically extract explainable spatial correlations from traffic data is still a challenging issue. The difficulty mainly comes from the inconsistency between static model structures and the dynamic evolution of traffic conditions. To overcome this difficulty, we proposed a novel multistep speed forecasting model, Dynamic Graph Filters Networks (DGFN). The major contribution is that the regular pixel-wise dynamic convolution is extended to graph topology. DGFN has a simple recurrent cell structure where local area-wide graph convolutional kernels are dynamically computed from varying inputs. Experiments on ring freeways show that DGFN is able to precisely predict short-term evolution of traffic speed. Furthermore, we theoretically explain why DGFN is not a pure "black-box", but a "gray-box" model that actually reduces entangled spatial and temporal features into one component representing dynamic spatial correlations. It permits tracking real-time interactions among adjacent links. DGFN has the potential to relate trained parameters in deep learning models with physical traffic variables.**

I. INTRODUCTION

With the development of urbanization, congestion has become a severe issue in many cities. To mitigate the incoming congestion in advance, timely short-term traffic forecasting has become one of the core functions in urban traffic control and guidance systems. For example, [1] and [2] showed that if travelers and traffic flows in urban areas are dynamically guided and rerouted using short-term traffic predictions, congestion and overall travel time can be effectively reduced during rush hours.

Network-level short-term traffic forecasting has been continuously studied for many years. This task is challenging because of the complex nonlinear spatiotemporal characteristics of traffic phenomena. Some key factors and variables required in traffic simulation systems, such as driving behaviors and origin-destination matrix, are difficult to estimate or costly to obtain. Therefore, data-driven approaches that focus on releasing the power of available and reliable data sources are becoming popular for practical applications. However, interpreting what these models learn from the limited observable data becomes a new issue. For traffic network dynamics, the interaction among adjacent links plays a critical role. The spatial correlation depends on not only the network topology, but also on real-time traffic conditions. How to define and extract these spatial correlations from data-driven models is pivotal for model interpretations. [3].

Many traffic forecasting models use predefined components to describe spatial correlations. For example, Space-Time Auto-Regressive Integrated Moving Average models (STARIMA) [4] induced weighting matrices calculated from the distances among various links to represent spatial correlations. But only upstream links are considered and all adjacent links share the same contribution coefficient. This simple assumption cannot reflect the complexity of traffic dynamics. Latent Space Model (LSM) [5] explicitly learns interactions between road segments based on their feature vectors. It permits modeling more complex but still static spatial dependencies. However, this state-independent assumption is not true according to traffic flow theories

Since 2014, although great effort has been conducted to apply deep learning techniques in short-term traffic forecasting domain [6]–[9], the same challenge still remains. Many deep learning models use stacked convolutional layers (regular convolution for route-level forecasting or graph convolution for network-level forecasting)[9]–[12] to capture spatial features. The complex inner structure and numerous trainable parameters lead to the "black-box" property. Additionally, convolutional kernels are fixed after training. The inconsistency between static convolutional structures and dynamic spatial correlations hinders us from seeking real-time explanations. Graph attention networks (GAT) [13] and its variants [14]–[16] are promising solutions. Dynamic convolutional kernels are computed from varying inputs to extract spatial features when applying the graph attention mechanism. If the specialities of traffic dynamics are considered properly when constructing model architectures, then extracting dynamic spatial correlations could be possible.

To this specific end, we proposed a novel graph attention variant, *Dynamic Graph Convolution* (DGC). The design of DGC considers one basic fact of traffic dynamics: Interactions among adjacent links depend on the network connectivity and area-wide traffic states. It means that traffic flow properties can only spread in a limited speed along the links on a road network. The core idea is to learn and mimic this information flow, such as stop-and-go waves. Next, DGC modules are implemented in a very simple RNN encoder-decoder to realize multistep speed forecasting, named as *Dynamic Graph Filters Networks* (DGFN). Validated on real-life datasets, DGFN yields satisfying short-term predictions up to $20\,\mathrm{min}$. We also theoretically explain that DGFN can convert spatial and temporal dependencies to dynamic spatial dependencies. This provides an alternative and simpler tool to study how deep learning models predict traffic conditions.

II. Methodology

Traffic dynamics on a road network can be written as a spatiotemporal graph:

$$\mathcal{G}(\mathcal{V}_N, \mathcal{E}, \mathbf{A}_{N\times N}; \mathbf{X}_{T\times N\times C}) \tag{1}$$

where $\mathcal{V}_N$ is the set of N nodes, $\mathcal{E}$ is the set of all edges, $\mathbf{A}$ is the adjacency matrix. For T time steps, traffic dynamics is represented by the tensor $\mathbf{X}$. The feature vector of node i at time t is $\mathbf{X}_{t,i} \in \mathbb{R}^C$, which contains C traffic variables.

Short-term traffic forecasting can be formulated as a sequence-to-sequence regression task: input is the observed feature tensor in the past m time steps, output is the predicted feature tensor in the next p time steps that maximizes the following conditional probability:

$$\hat{\mathbf{X}}^{\text{pred}}_{p\times N\times C} = \underset{\mathbf{X}^{\text{real}}_{p\times N\times C}}{\arg\max}\ Pr(\mathbf{X}^{\text{real}}_{p\times N\times C} | \mathbf{X}^{\text{obs}}_{m\times N\times C};\ \mathcal{G}) \tag{2}$$

On a spatiotemporal graph $\mathcal{G}$, we note the set of all nodes within k walks of edges from a node v_i as $\mathcal{N}_i^k$. If the latent representation of node v_i after applying graph convolution, noted as $\vec{y}_i$, is a function of all feature vectors within $\mathcal{N}_i^k$:

$$\vec{y}_i = \varphi_{\text{agg}}(\{\vec{x}_j \mid j \in \mathcal{N}_i^k\};\ \mathcal{G}) \tag{3}$$

then $\varphi_{\text{agg}}()$ is called a *k-walk graph aggregator*. The dimension of $\vec{y}_i$ depends on $\varphi_{\text{agg}}()$. It can be different from the dimension of inputs.

A. Dynamic Graph Convolution (DGC)

Dynamic Filters Networks (DFN) [17] was one of the earliest dynamic convolutional neural networks. Our design is similar to DFN but extends it to graph topology. One dynamic graph convolutional module has three parts: (I) filter generation network generates graph convolutional kernels from real-time inputs; (II) a regular spatial-domain graph convolution using the generated kernel; (III) post-processing layer adjusts output dimension. From the view of a central node i the formula of DGC k-walk graph aggregator is:

$$\begin{cases} s_{j,i} = (\sum\limits_{j\in\mathcal{N}_i^k} \beta_{j,i}\langle\vec{\alpha}_i,\ \vec{x}_j\rangle) + b_i \\ w_{j,i} = \dfrac{\exp(s_{j,i})}{\sum\limits_{j\in\mathcal{N}_i^k}\exp(s_{j,i})} \\ \vec{h}_i = \sum\limits_{j\in\mathcal{N}_i^k} w_{j,i}\vec{x}_j \\ \vec{y}_i = FC_{\theta_{C'}}(\vec{h}_i) \end{cases} \tag{4}$$

where $\beta_{j,i}$, b_i are trainable scalars and $\vec{\alpha}_i$ is a trainable vector that has the same dimension as node feature vector $\vec{x}_j$; $\langle\ ,\ \rangle$ represents the inner product of two vectors. The adjacency matrix, which is not explicitly given in (4), is used to calculate which nodes belong to $\mathcal{N}_i^k$. The first equation calculates the initial value of each weight $s_{j,i}$ from all feature vectors in $\mathcal{N}_i^k$. Shared weights $\vec{\alpha}_i$ and b_i effectively reduce the total number of trainable parameters and the location-specific weight $\beta_{j,i}$ preserves the uniqueness of each node. The second equation is a nonlinear *softmax* normalization. All normalized weights $w_{j,i}$ are non-negative and the sum of them in the receptive field $\mathcal{N}_i^k$ equals to 1. The third equation is a regular extended graph convolution [18] using the generated kernel. The last equation is a shared fully-connected (FC) layer with C' output units. The FC layer is applied on every node to adjust the output dimension to C'. The DGC graph aggregator is visualized in Fig.1. For simplification, we briefly note a DGC module in the form of matrices with $N \times C'$ output dimension and k-walk graph aggregator as:

$$\mathbf{Y}^{\text{out}}_{N\times C'} = \text{DGC}_k(\mathbf{X}^{\text{in}}_{N\times C}, C'; \mathcal{G}) \tag{5}$$

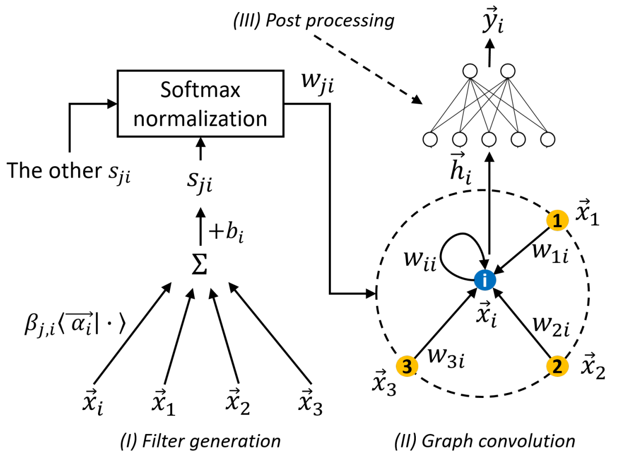


Fig. 1: An example of the DGC graph aggregator structure: $k = 1$

DGC is a variant of spatial attention mechanism [15]. Compared to another type of spatial attention, graph attention networks (GAT) [13], DGC is more sensitive to the order of adjacent nodes because before applying the *softmax* normalization, the initial weights $s_{j,i}$ are already k-walk graph aggregators. In GAT aggregator, if two adjacent nodes exchange their feature vectors, the latent representation of the central node does not change. Because both the computation of pairwise similarities and *softmax* normalization are independent from the order of nodes in one receptive field.

B. Dynamic Graph Filters Networks (DGFN)

Next, we combine DGC modules with RNN encoder-decoder models [19] for multistep speed prediction. Scheduled sampling strategy [20] is employed to enhance the learning process. Based on DGC modules, we propose a simple RNN cell where only a single recurrent link is added. It is similar to [17] so we give our model a similar name:

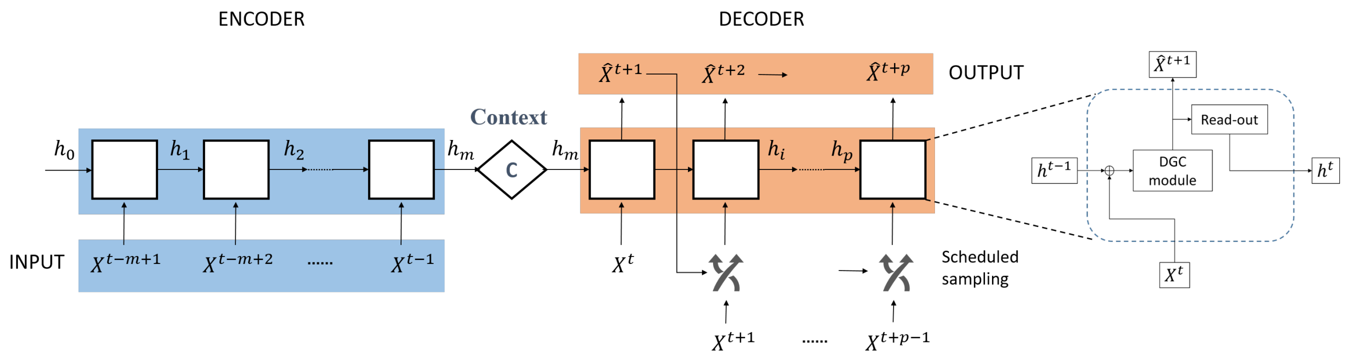


Fig. 2: Architecture of DGFN model (left) and the structure of the corresponding RNN cells (right)

Dynamic Graph Filters Networks (DGFN). The architecture is shown in Fig.2 and the mathematical formula of the DGFN cell is given by (6). The DGC module outputs the prediction of the next time step and a two layer fully-connected feedforward neural network (FFNN) generates the new hidden state. There are two hyperparameters: the receptive field k in the DGC module, and the number of neurons of the hidden layer in FFNN u.

$$\begin{cases} \mathbf{X}^{t+1} = \mathrm{DGC}_k([\mathbf{X}^t, \mathbf{h}^{t-1}], 1; \mathcal{G}) \\ \mathbf{h}^t = \mathrm{FC}_{\theta_1}(\mathrm{FC}_{\theta_u}(\mathbf{X}^{t+1})) \end{cases} \tag{6}$$

where [ ] represents the concatenation of two feature tensors along the nodes. Two DGFN cells with different parameters are implemented in the encoder and the decoder respectively. The encoding and step-by-step decoding process can be written as:

$$\begin{cases} \mathbf{h}^0 = \mathbf{C} = \mathrm{Enc}(\mathbf{X}^{\mathrm{obs}}_{(m-1)\times N\times C}) \\ \hat{\mathbf{X}}^{t+1} = \mathrm{DGC}_k([\mathbf{X}^t, \mathbf{h}^{t-1}], 1; \mathcal{G}) \quad 1 \le t \le p \\ \mathbf{h}^t = \mathrm{FC}_{\theta_1}(\mathrm{FC}_{\theta_u}(\hat{\mathbf{X}}^{t+1})) \quad 1 \le t \le p \end{cases} \tag{7}$$

Here the input is the observation of the past m time steps, output is the predictions of the following p time steps.

III. EXPERIMENTS

In this section, we validate the proposed model on selected freeway datasets. The data is collected and pre-processed by *National Data Warehouse for Traffic Information* (NDW). Missing points in raw data are estimated to give the complete evolution of traffic conditions.

A. Data preparation

The frequently congested urban freeway around Rotterdam is selected as a case study. We consider both directions, clockwise ("CL") and counter-clockwise ("CC"). Both ring freeways are uniformly partitioned into $200\,\mathrm{m}$ length links. Freeway CL has 199 links and freeway CC contains 208 links. We use one node to represent the traffic state on that link and construct graphs.

Speed data of the entire year of 2018 were prepared. The time interval between adjacent two time steps is $2\,\mathrm{min}$. All holidays and weekends are removed from the dataset due to the lack of congestion. 27 weeks of highly congested work-days (135 days) are selected. The 27 weeks are randomly shuffled and divided into 3 groups: 18 weeks as train set, 4 weeks as validation set, and 5 weeks for testing. We particularly focus on models' performances during evening rush hours, so only the data in the most congested period, 14:00-19:00, are included in the final dataset. For the forecasting task, observation window is $30\,\mathrm{min}$ (15 steps) and prediction horizon is $20\,\mathrm{min}$ (10 steps). The network of the ring freeway CC and an example of its congestion patterns are shown in Fig.3.

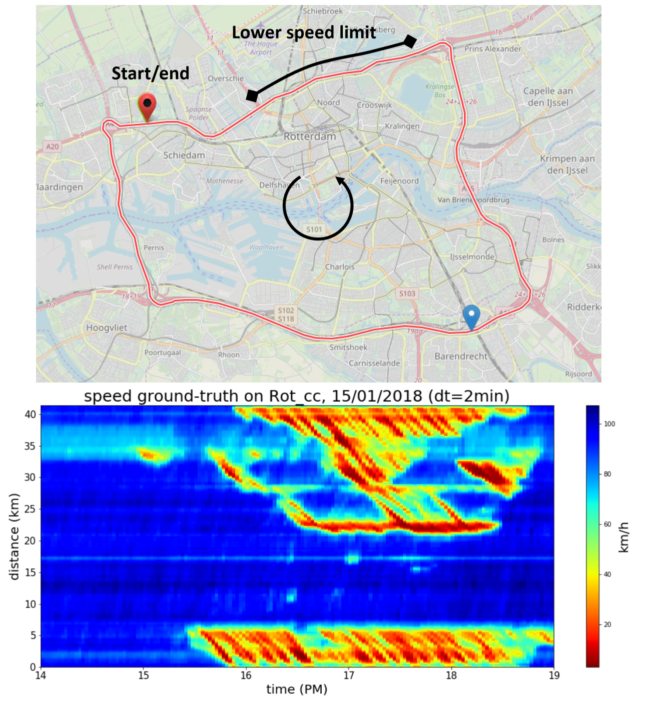


Fig. 3: The network of ring freeway CC (top) and an example of congestion patterns during evening peak hours (bottom). There exists a $5\,\mathrm{km}$ segment with lower speed limit ($70\,\mathrm{km\,h^{-1}}$) on the north.

In training, data is normalized between 0 and 1 by dividing the speed by $120\,\mathrm{km\,h^{-1}}$. For testing, the input keeps as normalized data, but the output is multiplied by $120\,\mathrm{km\,h^{-1}}$

to give real-value predictions. Then the multistep errors are calculated between the ground-truth and predicted speed evolution.

B. Benchmark models

DGFN is compared with some selected benchmark models. Their details and the hyperparameters setting of DGFN are listed below:

- **Historical average** (HA): HA is the simplest model that calculates the average daily speed evolution in historical datasets as predictions for the future. HA gives the same prediction everyday and its forecasting accuracy does not change with prediction horizon. It is chosen as a baseline.
- **K-nearest neighbors** (KNN): K-nearest neighbors regression [21] is a widely used statistical model. Given a query input, it firstly searches the most similar speed patterns up to the k-th, then calculates the prediction by the weighted average of the corresponding future traffic states of these k patterns. In this study, the similarity is measured by Euclidean distances between two speed heat maps, where one axis is time and the other is the spatial coordinate. We scan the entire train set, including all time periods, to find the most similar patterns and their weights are normalized by the inverse of Euclidean distances. We do not consider the similarities across different nodes. The optimal hyperparameter $K = 25$ is chosen by grid searching and cross validation.
- **AGC-seq2seq** (simplified): [12] proposed the Attention Graph Convolution (AGC)-seq2seq model. To avoid inducing additional data, we keep the framework but modify some unnecessary parts: exogenous information is not added in the encoder, statistics of historical record is not used in the decoder, and normal scheduled sampling training strategy is employed.
- **GAT**: Different GAT [13] modules are implemented in Gated Recurrent Units (GRU) cells by replacing the gates. We use the similar encoder-decoder framework and scheduled sampling training strategy. The only hyperparameter is $k = 3$.
- **DGFN**: Two hyperparameters are receptive field $k = 3$ and the hidden neurons in the read-out unit $u = 64$.

Mean average error (MAE) measures the average precision in all areas; mean average percentage error (MAPE) is more sensitive to the contours of low-speed congested areas; root mean square error (RMSE) credits a disproportionately large effect for the points with bigger speed errors. The three metrics are used to evaluate models' performances. In training, we force the models to focus more on low-speed congested areas because traffic managers are more interested in predicting congestion instead of free-flow cases. Thus, MAPE between multistep predictions and the ground-truth is chosen as the loss function (The minimal speed is $1\,\mathrm{km\,h^{-1}}$):

$$loss = \frac{1}{D}\sum_{i=1}^{D}\frac{|\hat{\mathbf{X}}^i - \mathbf{X}^i|}{\mathbf{X}^i} \qquad (8)$$

where D is the number of data points.

Adam optimizer is chosen to minimize the loss function. Initial learning rate, decay rate, and scheduled sampling parameters are well tuned for each model. Early stopping on validation set is used to mitigate overfitting. Our experimental platform has one CPU (Intel(R) Core$^{\mathrm{TM}}$i7-7700 CPU @ 3.60GHz, 3601 Mhz, 4 Core(s), 8 Logical Processors), 32-GB installed RAM, and one GPU (NVIDIA GeForce GTX 1070, 16GB). All deep learning models are trained in parallel and tested on GPU. Source code and details of parameters setting is open online: `https://github.com/RomainLITUD`.

C. Results

Table I lists the performances of the proposed model and the benchmark models on different freeways. The running time is the sum of training time and testing time. Fig.4 shows the relation between multistep MAPE and the prediction horizon. HA is not given because its performance does not change with prediction horizon. We have the following conclusions:

- DGFN outperforms the other models in terms of MAE, MAPE and RMSE in both forecasting tasks. Especially, even with simpler recurrent structure, DGFN consistently gives better predictions than GAT within $20\,\mathrm{min}$.
- Fig.4 shows that DGFN has bigger advantage over GAT and AGC-seq2seq(s) in terms of MAPE for shorter prediction horizons. But compared to GAT and AGC-seq2seq(s), DGFN's MAPE increases faster with the increasing of prediction horizon because its simple recurrent structure is not as good as GRU for capturing long-term temporal dependencies.
- Among the three deep learning models, DGFN takes more running time than AGC-seq2seq(s) because of the filter generation network in DGC modules. But DGFN runs much faster than GAT. The complex GAT gate structures in GRU cells consume more time.
- KNN's performance is not satisfying, especially for MAPE. This indicates that the congestion patterns on these two freeways are very diverse. There are many unique patterns in the database.
- Generally speaking, selected deep learning models give better predictions than HA and KNN, but consume more training time.

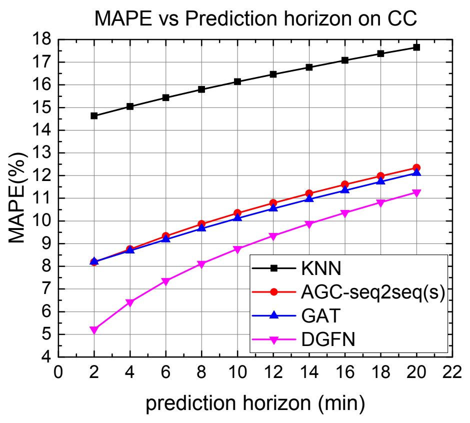
In brief, in this section we validate the proposed model on real-world datasets. By comparing the forecasting precision of the proposed model with benchmark models, we show that DGFN is able to give short-term predictions of good quality during evening peak hours.

IV. SPATIAL CORRELATIONS

In this section, we will interpret how DGFN cells learn dynamic spatial correlations from a theoretical view. In most deep learning based traffic forecasting models, temporal dependencies and spatial correlations are modeled separately by different attention layers, e.g. [12], [18], [22], but this disentangled interpretation does not conform to traffic flow

TABLE I: Prediction performances comparison on two ring freeways during peak hours 14:00 - 19:00

Model	MAE($\mathrm{km\,h^{-1}}$)		MAPE(%)		RMSE($\mathrm{km\,h^{-1}}$)		Time(s)
freeway	CC	CL	CC	CL	CC	CL	Average
HA	9.83	9.81	12.92	13.74	16.39	16.02	72
KNN	6.95	6.29	17.66	14.9	12.94	11.38	240
AGC-seq2seq(s)	6.35	6.23	12.34	10.15	12.01	10.64	350
GAT	6.50	5.69	12.11	9.74	12.28	10.55	1350
DGFN	**5.94**	**5.07**	**11.27**	**9.00**	**11.66**	**9.76**	600



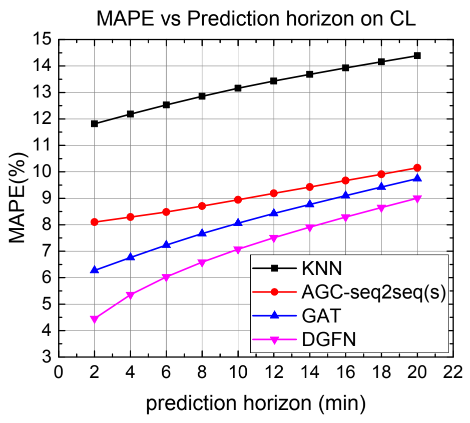


Fig. 4: Accumulation of MAPE with prediction horizon on two ring freeways

theories. DGFN actually re-defines dynamic spatial dependencies based on graph convolution, and provides a more convenient alternative tool.

For simplification, we consider multistep observations but only one step prediction. The encoder encrypts observed traffic conditions in the past $m-1$ time steps into the context vector $\mathbf{C}$, and $\mathbf{C}$ is concatenated with the current input $\mathbf{X}^t$. Thus, each node is associated with two features, (c_i, x_i), which represent the historical evolution and the current traffic condition respectively. Because of the *softmax* normalization in (4), all weights are positive and the sum of them in each receptive field equals to 1. It is reasonable to define the generated dynamic graph convolutional kernel $\mathbf{W}$ as real-time spatial correlations. For example, $w_{j,i}$ represents the impact of node-j on node-i for the prediction of next time step (see Fig.1). This definition is dynamic and input-dependent. We do not need to treat temporal and spatial dependencies separately.

Further, because the DGC module contains location-specific trainable parameters $\beta_{j,i}$, DGFN can automatically learn which adjacent node has the highest influence on the central node not only under different traffic conditions but also at different locations. It enables us to finely study the varying spatial correlations.

In summary, DGFN is theoretically not a pure black-box model. It is possibly able to let us extract real-time dynamic spatial correlations, compare them with predicted traffic states or network topology, and help researchers to better understand how spatial correlations evolve on a road network.

V. CONCLUSION

In this study, we built a novel deep learning based multistep traffic forecasting model that is able to dynamically capture spatial correlations. First we designed dynamic graph convolution modules to extend dynamic convolution from normal Euclidean spaces to graph structures. Based on DGC modules, a multistep deep learning framework, dynamic graph filters networks, was proposed. DGFN outperforms the selected benchmark models in terms of different metrics on two ring freeway data bases. Then we presented a brief explanation on how to define dynamic spatial correlations in DGFN. It permits tracking spatial dependencies for each node and comparing them with real-time traffic states. DGFN is a gray-box model that learns how condition/location-dependent information propagates in a network.

The further researches will focus on more detailed model interpretations. For example, build a statistical relation between spatial correlations and traffic states, select some special positions such as on-ramps or off-ramps and study how DGFN distributes spatial attention to predict splitting and merging of congestion, find the optimal receptive field k for different forecasting tasks and study its influence on models' performances, etc. These answers can help designing more explainable and accurate data-driven traffic forecasting models.

REFERENCES

[1] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 316–324.

[2] T. Liebig, N. Piatkowski, C. Bockermann, and K. Morik, “Dynamic route planning with real-time traffic predictions,” *Information Systems*, vol. 64, pp. 258–265, 2017.

[3] A. Ermagun and D. Levinson, “Spatiotemporal traffic forecasting: review and proposed directions,” *Transport Reviews*, vol. 38, no. 6, pp. 786–814, 2018.

[4] Y. Kamarianakis and P. Prastacos, “Space-time modeling of traffic flow,” *Computers Geosciences*, vol. 31, no. 2, pp. 119–133, 2005.

[5] D. Deng, C. Shahabi, U. Demiryurek, L. Zhu, R. Yu, and Y. Liu, “Latent space model for road networks to predict time-varying traffic,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1525–1534.

[6] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, “Long short-term memory neural network for traffic speed prediction using remote microwave sensor data,” *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[7] N. G. Polson and V. O. Sokolov, “Deep learning for short-term traffic flow prediction,” *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 1–17, 2017.

[8] Y. Li, R. Yu, C. Shahabi, and Y. Liu, “Diffusion convolutional recurrent neural network: Data-driven traffic forecasting,” in *ICLR 2018 : International Conference on Learning Representations 2018*, 2018.

[9] B. Yu, H. Yin, and Z. Zhu, “Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting,” in *IJCAI 2018: 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 3634–3640.

[10] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *ICLR 2017 : International Conference on Learning Representations 2017*, 2017.

[11] W. L. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[12] Z. Zhang, M. Li, X. Lin, Y. Wang, and F. He, “Multistep speed prediction on traffic networks: A deep learning approach considering spatio-temporal dependencies,” *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 297–322, 2019.

[13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” in *ICLR 2018 : International Conference on Learning Representations 2018*, 2018.

[14] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D. yan Yeung, “Gaan: Gated attention networks for learning on large and spatiotemporal graphs,” in *UAI 2018: The Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018, pp. 339–349.

[15] X. Zhu, D. Cheng, Z. Zhang, S. Lin, and J. Dai, “An empirical study of spatial attention mechanisms in deep networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6688–6697.

[16] L. N. Do, H. L. Vu, B. Q. Vo, Z. Liu, and D. Phung, “An effective spatial-temporal attention based neural network for traffic flow prediction,” *Transportation research part C: emerging technologies*, vol. 108, pp. 12–28, 2019.

[17] B. D. Brabandere, X. Jia, T. Tuytelaars, and L. V. Gool, “Dynamic filter networks,” in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 667–675.

[18] B. Yu, Y. Lee, and K. Sohn, “Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network (gcn),” *Transportation Research Part C: Emerging Technologies*, vol. 114, pp. 189–204, 2020.

[19] S. Kumar and M. Asger, “A study of state-of-the-art neural machine translation approaches,” *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 4, no. 1, pp. 135–139, 2018.

[20] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, “Scheduled sampling for sequence prediction with recurrent neural networks,” in *Advances in Neural Information Processing Systems*, 2015, pp. 1171–1179.

[21] T. Denoeux, “A k-nearest neighbor classification rule based on dempster-shafer theory,” *IEEE transactions on systems, man, and cybernetics*, vol. 25, no. 5, pp. 804–813, 1995.

[22] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, “Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2019.