Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting

Authors: Bing Yu* [1], Haoteng Yin* [2, 3], Zhanxing Zhu† [3, 4]

[1] School of Mathematical Sciences, Peking University, Beijing, China

[2] Academy for Advanced Interdisciplinary Studies, Peking University, Beijing, China

[3] Center for Data Science, Peking University, Beijing, China

[4] Beijing Institute of Big Data Research (BIBDR), Beijing, China

* Equal contributions

† Corresponding author

Publication: IJCIA-18

Presentation: Jie Zhang

Outline

- Introduction
- Preliminary
- Proposed Model
- Experiments
- Conclusion and Future Work
- Related Works

Introduction

- Traffic forecast is generally classified into two scales: short-term (5 ~ 30 mins), medium and long term (over 30 mins).

- Studies on mid-and-long term traffic prediction can be divided into dynamical modeling and data-driven methods.

- Classic statistical and machine learning models are two major representatives of data-driven methods.

- Some researchers use convolutional neural network (CNN) to capture adjacent relations among the traffic network, along with employing recurrent neural network (RNN) on time axis.

- However, the normal convolutional operation applied restricts the model to only process grid structures (e.g. images).

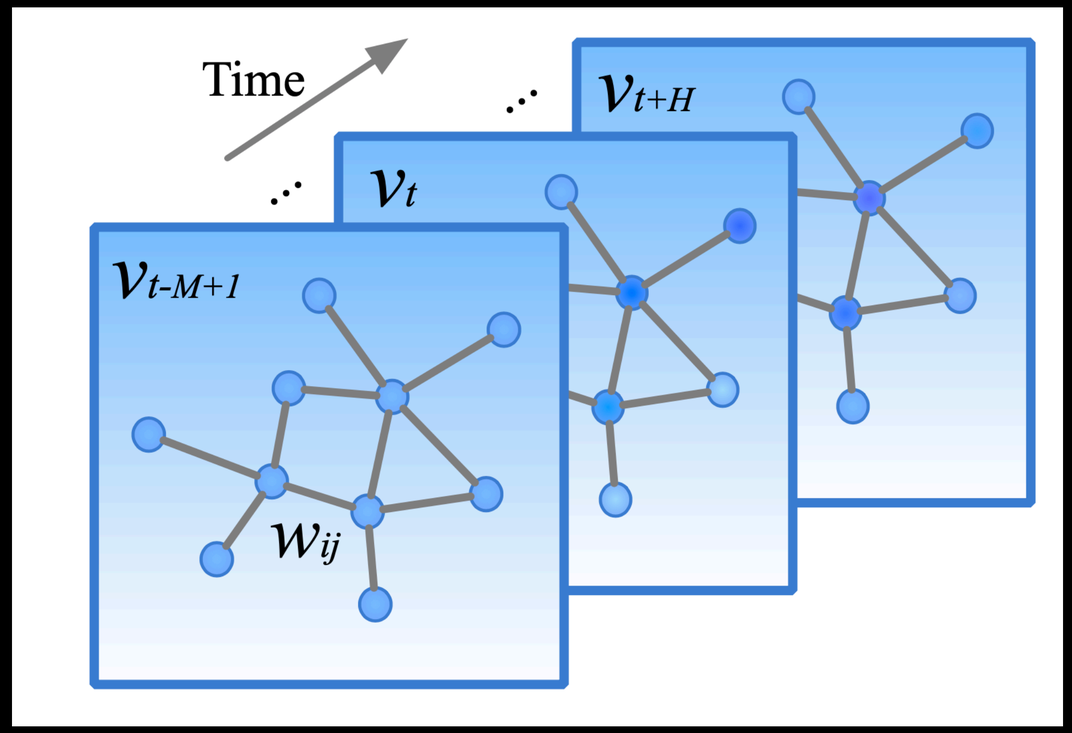
- RNN-based networks (including LSTM) are widely known to be difficult to train and computationally heavy.

- To fully utilize spatial information, we model the traffic network by a general graph instead of treating it separately (e.g. grids or segments).
- To handle the inherent deficiencies of recurrent networks, we employ a fully convolutional structure on time axis.
- We propose the spatio-temporal graph convolutional networks for traffic forecasting tasks.

Preliminary

Figure 1: Graph-structured traffic data

- At the tth time step,
- in the undirected graph $G_t = (V_t, E, W)$,
  - V_t is a finite set of vertices, corresponding to the observations from n monitor stations in a traffic network;
  - E is a set of edges;
  - $W \in R^{n \times n}$ denotes the weighted adjacency matrix.



Traffic Prediction on Road Graphs

- Predicting the most likely traffic measurements in the next H time steps given the previous M traffic observations as,
  - $\hat{v}_{t+1}, \ldots, \hat{v}_{t+H} = argmax_{v_{t+1}, \ldots, v_{t+H}} \; log \; P(v_{t+1}, \ldots, v_{t+H} | v_{t-M+1}, \ldots, v_t)$
  - where $v_t \in R^n$ is an observation vector of n road segments at time step t, each element of which records historical observation for a single road segment.

Proposed Model

Network Architecture

- STGCN consists of two spatio-temporal convolutional blocks and a fully-connected output layer in the end.
- Each ST-Conv block contains two temporal gated convolution layers and one spatial graph convolution layer in the middle.
- The residual connection and bottleneck strategy are applied inside each block.

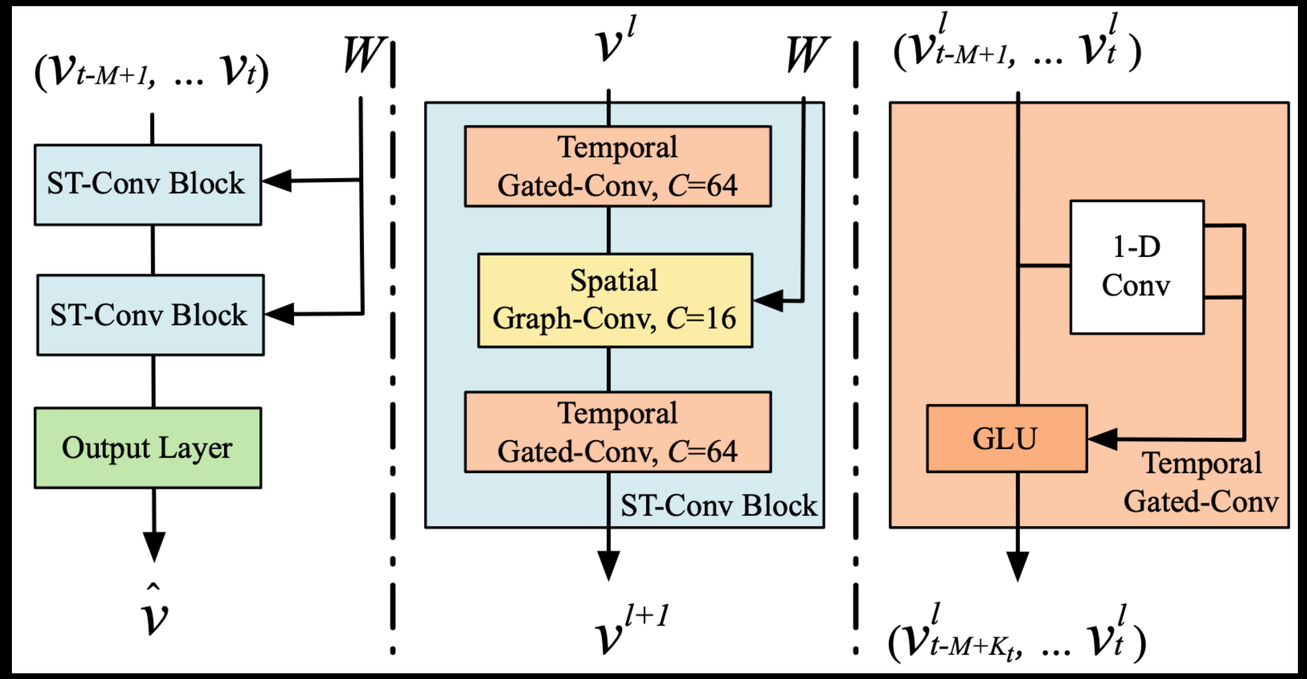


Figure 2: Architecture of spatio-temporal graph convolutional networks

Temporal Gated Convolution layer

- A temporal gated convolution layer contains
  - a Gated Linear Unit (GLU)
  - and two Temporal Convolution Networks (TCN) with Causal Convolution

Gated Linear Units (GLU)

- GLU was proposed in *Language Modeling with Gated Convolution Networks*
- The hidden layer defined as $h(X) = (X \circ W + b) \otimes \sigma(X \circ V + c)$

From *Language Modeling with Gated Convolutional Networks*

Gated Linear Units (GLU)

- When convolving inputs, we take care that h_i does not contain information from future words.
- We address this by shifting the convolutional inputs to prevent the kernels from seeing future context.
- Specifically, we zero-pad the beginning of the sequence with $k - 1$ elements, assuming the first input element is the beginning of sequence marker which we do not predict and k is the width of the kernel.

TCN with Causal Convolution

- The prediction $p(x_{t+1} \mid x_1, \ldots, x_t)$ emitted by the model at timestep t cannot depend on any of the future timesteps $x_{t+1}, x_{t+2}, \ldots, x_T$.
- For 1-D data can more easily implement this by shifting the output of a normal convolution by a few timesteps.
- At training time, the conditional predictions for all timesteps can be made in parallel.

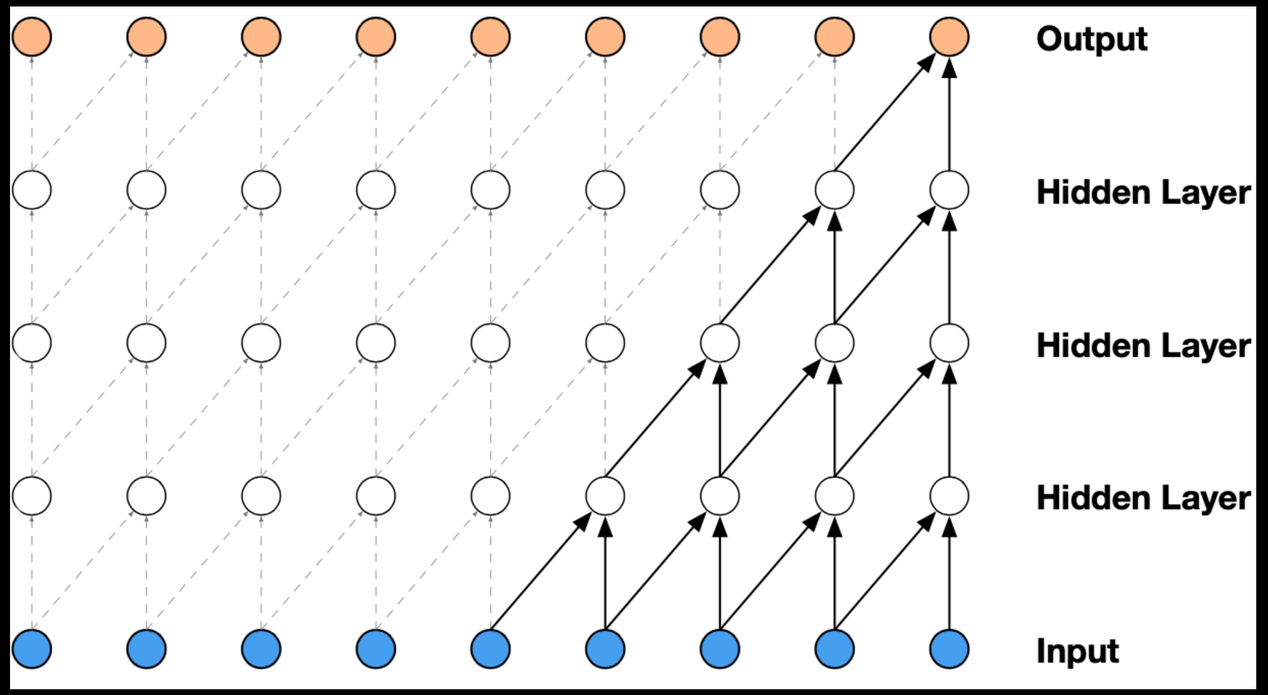


Figure 3: Visualization of a stack of causal convolutional layers

Gated CNNs for Extracting Temporal Features

- The temporal convolutional layer contains a 1-D causal convolution with a width-K_t kernel followed by GLU as a non-linearity.

- For each node in graph G, the temporal convolution explores K_t neighbors of input elements without padding which leading to shorten the length of sequences by $K_t - 1$ each time.

- Thus, input of temporal convolution for each node can be regarded as a length-M sequence with C_i channels as $Y \in R^{M \times C_i}$.

- The convolution kernel $\Gamma \in R^{K_t \times C_i \times 2C_o}$ is designed to map the input Y to a single output element $[P\ Q] \in R^{(M-K_t+1) \times (2C_o)}$ ($P,\ Q$ is spilt in half with the same size of channels).

- The temporal gated convolution can be defined as $\Gamma \star_T Y = P \circ \sigma(Q) \in R^{(M-K_t+1) \times C_o}$

Convolution on Graphs

- Spatial methods: define convolution in the spatial domain (vertex domain)
  - Convolution is defined as a weighted average function over all vertices located in the neighborhood of target vertex
  - The main challenge is that the size of neighborhood varies remarkably across nodes, e.g., power-law degree distribution
  - E.g., GraphSAGE
- Spectral methods: define convolution in the spectral domain
  - Convolution is defined via graph Fourier transform and convolution theorem
  - The main challenge is that convolution filter defined in spectral domain is not localized in vertex domain
  - E.g., GCN

Graph Convolution Layer

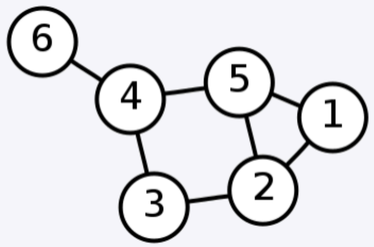
- Graph Convolutional Network
  - Spectral Convolution Neural Networks (2014)
  - Chebyshev Polynomials Approximation by ChebyNet (2016)
  - Graph Convolution Networks (2017)

Define Convolution

- Let f and g be two functions with convolution $f * g$.
- Let F denotes the Fourier transform operator, then $F\{f\}$ and $F\{g\}$ are the Fourier transform of f and g, respectively.
- And by applying the inverse Fourier transform F^{-1}, then
  - $f * g = F^{-1}\{F\{f\} * F\{g\}\}$ and $f \cdot g = F^{-1}\{F\{f\} \cdot F\{g\}\}$

Laplacian matrix

- Combinatorial Laplacian
  $L_{com} = D - A \in R^{n \times n}$,
  - where $D \in R^{n \times n}$ is the diagonal degree matrix with $D_{ii} = \sum_j A_{ij}$,
- Symmetric Normalized Laplacian matrix:
  $L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I_n - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$,
  - where I_n is the identity matrix.

Labelled graph	Degree matrix	Adjacency matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$

Spectral decomposition of Laplacian matrix

- $L = U\Lambda U^{-1} = U\Lambda U^T$,
  - where $\Lambda = diag([\lambda_0, \ldots, \lambda_{n-1}]) \in R^{n \times n}$
  - $U = [u_0, \ldots, u_{n-1}] \in R^{n \times n}$ is the Fourier basis

Graph Fourier Transform and Graph Fourier Inverse Transform

- The graph Fourier transform of a signal $x \in R^n$ is defined as
  - $\hat{x} = U^{\mathrm{T}} x$
- The graph Fourier inverse transform is defined as
  - $x = U\hat{x}$

Define convolution in spectral domain

- Given a signal x as input and the other signal y as a filter, graph convolution $\star_G$ could be defined as
  - $x \star_G y = U((U^{\mathrm{T}}x) \odot (U^{\mathrm{T}}y))$

Spectral filtering of graph signals

- A signal x is filtered by g_θ as
  - $g_\theta(L)x = g_\theta(U\Lambda U^{\mathrm{T}})x = Ug_\theta(\Lambda)U^{\mathrm{T}}x$
- A non-parametric filter, i.e. a filter whose parameters are all free, would be defined as
  - $g_\theta(\Lambda) = diag(\theta)$
  - where the parameter $\theta \in R^n$ is a vector of Fourier coefficients

Shortcomings of Spectral CNN

- Requiring eigen-decomposition of Laplacian matrix
  - Eigenvectors are explicitly used in convolution
- High computational cost
  - Multiplication with graph Fourier basis U is $O(n^2)$
- Not localized in vertex domain

Polynomial parametrization for localized filters

- Let $g_\theta(\Lambda) \approx \sum_{k=0}^{K-1} \theta_k \Lambda^k$, then
- $g_\theta \star_G x \approx U \sum_{k=0}^{K-1} \theta_k \Lambda^k U^{\mathrm{T}} x = \sum_{k=0}^{K-1} \theta_k U \Lambda^k U^{\mathrm{T}} x = \sum_{k=0}^{K-1} \theta_k L^k x$
- Time complexity is still $O(n^2)$ because of the multiplication with the graph Fourier basis U.

Polynomial parametrization for localized filters

- The Chebyshev polynomials of the first kind are recursively defined as
  - $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$
  - $T_0(x) = 1$
  - $T_1(x) = x$
  - $T_k(x) = cos(k \cdot arccos(x))$
- $g_\theta(\Lambda)$ can be well-approximated by a truncated expansion in terms of order K:
  - $g_\theta(\Lambda) \approx \sum_{K=0}^{K-1} \theta_k T_k(\widetilde{\Lambda})$
  - where the parameter $\theta \in R^K$ is a vector of Chebyshev coefficients
  - and $T_k(\widetilde{\Lambda}) \in R^{n \times n}$ is the Chebyshev polynomial of order k evaluated at $\widetilde{\Lambda} = \frac{2\Lambda}{\lambda_{max}} - I_n$,
  - a diagonal matrix of scaled eigenvalues that lie in $[-1, 1]$ because of the $arccos(\cdot)$ function.

Polynomial parametrization for localized filters

- The filtering operation can then be written as
  - $$g_\theta(L)x \approx \sum_{k=0}^{K-1} \theta_k T_k(\widetilde{L})x$$
  - with scaled Laplacian $\widetilde{L} = \frac{2L}{\lambda_{max}} - I_n$.

Strengths of Chebyshev Polynomials Approximation by ChebyNet

- Eigen-decomposition is not required
- Computational cost is $O(K|E|) \ll O(n^2)$

Graph Convolution Networks

- In this linear formulation of a GCN, we further approximate $\lambda_{max} \approx 2$.
- Under these approximations $g_\theta(L)x = \sum_{k=0}^{K-1} \theta_k T_k(\widetilde{L})x$ simplifies to:
  - $g_\theta \star_G x \approx \theta_0 x + \theta_1(L - I_n)x = \theta_0 x - \theta_1(D^{-\frac{1}{2}}AD^{-\frac{1}{2}})x$
  - with two free parameters θ_0 and θ_1. The filter parameters can be shared over the whole graph. Successive application of filters of this form then effectively convolve the Kth-order neighborhood of a node, where K is the number of successive filtering operations or convolutional layers in the neural network model.

Graph Convolution Networks

- Let $\theta = \theta_0 = -\theta_1$,
- then $g_\theta \star_G x \approx \theta(I_n + D^{-\frac{1}{2}}AD^{-\frac{1}{2}})x$,
- To alleviate the gradient exploding or vanishing issue, we introduce the renormalization trick:
  - $I_n + D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \rightarrow \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\,\widetilde{D}^{-\frac{1}{2}}$, with $\widetilde{A} = A + I_n$ and $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$
- Finally, let $\hat{L} = \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\,\widetilde{D}^{-\frac{1}{2}}$
  - $g_\theta \star_G x \approx \theta\hat{L}x$

Experiment Results

Model	BJER4(15/ 30/ 45 min)		
	MAE	MAPE (%)	RMSE
HA	5.21	14.64	7.56
LSVR	4.24/ 5.23/ 6.12	10.11/ 12.70/ 14.95	5.91/ 7.27/ 8.81
ARIMA	5.99/ 6.27/ 6.70	15.42/ 16.36/ 17.67	8.19/ 8.38/ 8.72
FNN	4.30/ 5.33/ 6.14	10.68/ 13.48/ 15.82	5.86/ 7.31/ 8.58
FC-LSTM	4.24/ 4.74/ 5.22	10.78/ 12.17/ 13.60	5.71/ 6.62/ 7.44
GCGRU	3.84/ 4.62/ 5.32	9.31/ 11.41/ 13.30	5.22/ 6.35/ 7.58
STGCN(Cheb)	**3.78/ 4.45/ 5.03**	**9.11/ 10.80/ 12.27**	**5.20/ 6.20/ 7.21**
STGCN(1st)	3.83/ 4.51/ 5.10	9.28/ 11.19/ 12.79	5.29/ 6.39/ 7.39

Experiment Results

Model	PeMSD7(M) (15/ 30/ 45 min)			PeMSD7(L) (15/ 30/ 45 min)		
	MAE	MAPE(%)	RMSE	MAE	MAPE(%)	RMSE
HA	4.01	10.61	7.20	4.60	12.50	8.05
LSVR	2.50/ 3.63/ 4.54	5.81/ 8.88/ 11.50	4.55/ 6.67/ 8.28	2.69/ 3.85/ 4.79	6.27/ 9.48/ 12.42	4.88/ 7.10/ 8.72
ARIMA	5.55/ 5.86/ 6.72	12.92/ 13.94/ 15.20	9.00/ 9.13/ 9.38	5.50/ 5.87/ 6.30	12.30/ 13.54/ 14.85	8.63/ 8.96/ 9.39
FNN	2.74/ 4.02/ 5.04	6.38/ 9.72/ 12.38	4.75/ 6.98/ 8.58	2.74/ 3.92/ 4.78	7.11/ 10.89/ 13.56	4.87/ 7.02/ 8.46
FC-LSTM	3.57/ 3.94/ 4.16	8.60/ 9.55/ 10.10	6.20/ 7.03/ 7.51	4.38/ 4.51/ 4.66	11.10/ 11.41/ 11.69	7.68/ 7.94/ 8.20
GCGRU	2.37/ 3.31/ 4.01	5.54/ 8.06/ 9.99	4.21/ 5.96/ 7.13	2.48/ 3.43/ 4.12 *	5.76/ 8.45/ 10.51 *	4.40/ 6.25/ 7.49 *
STGCN(Cheb)	**2.25/ 3.03/ 3.57**	5.26/ **7.33/ 8.69**	**4.04/ 5.70/ 6.77**	**2.37/ 3.27/ 3.97**	**5.56/ 7.98/ 9.73**	**4.32/ 6.21/ 7.45**
STGCN(1st)	2.26/ 3.09/ 3.79	**5.24**/ 7.39/ 9.12	4.07/ 5.77/ 7.03	2.40/ 3.31/ 4.01	5.63/ 8.21/ 10.12	4.38/ 6.43/ 7.81

For PeMSD7(L), GCGRU has to use the half of batch size since its GPU consumption exceeded the memory capacity of a single card (results marked as "*" in Table 2)

Experiment Results

Table 3: Time consumptions of training on the dataset PeMSD7.

Dataset	Time Consumption (s)		
	STGCN(Cheb)	STGCN(1st)	GCGRU
PeMSD7(M)	**272.34**	271.18	3834.54
PeMSD7(L)	1926.81	**1554.37**	19511.92

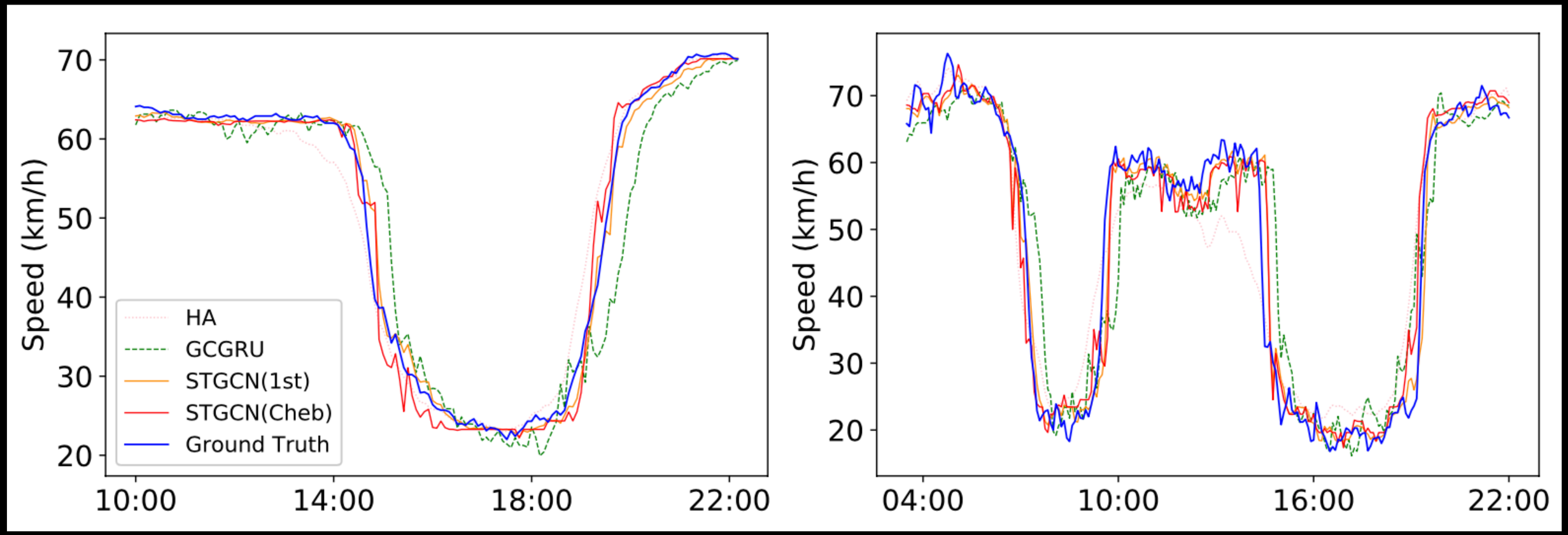


Figure 8: Speed prediction in morning peak and evening rush hours of the PeMSD 7.

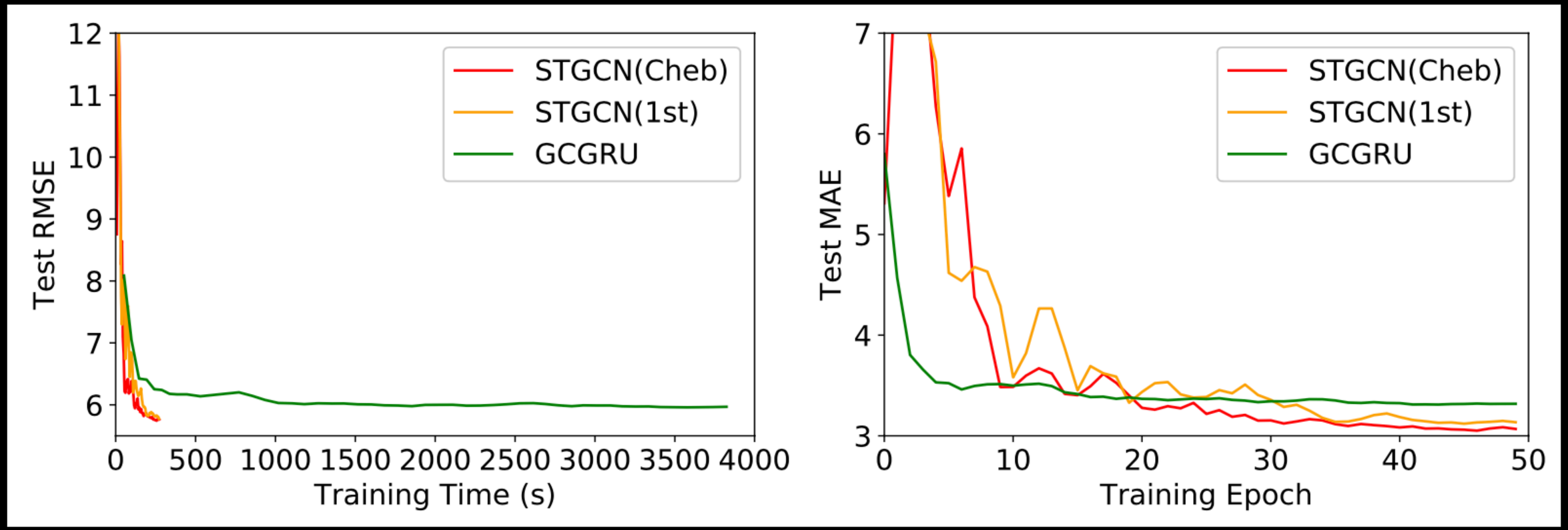


Figure 9: Test RMSE versus the training time (left); Test MAE versus the number of training epochs (right). (PeMSD 7 (M))

Conclusion and Future Work

- We propose a novel deep learning framework STGCN for traffic prediction, integrating graph convolution and gated temporal convolution through spatio-temporal convolutional blocks.

- Experiments show that our model achieves faster training, easier convergences, and fewer parameters with flexibility and scalability.

- This framework can be applied into more general spatio-temporal structured sequence forecasting scenarios

- In the future, we will further optimize the network structure and parameter settings.