DL-Traff: Survey and Benchmark of Deep Learning Models for Urban Traffic Prediction

Renhe Jiang\textsuperscript{1,2,*}, Du Yin\textsuperscript{2,*}, Zhaonan Wang\textsuperscript{1}, Yizhuo Wang\textsuperscript{2}, Jiewen Deng\textsuperscript{2}, Hangchen Liu\textsuperscript{2}, Zekun Cai\textsuperscript{1}, Jinliang Deng\textsuperscript{2,3}, Xuan Song\textsuperscript{2,1†}, Ryosuke Shibasaki\textsuperscript{1}

\textsuperscript{1}The University of Tokyo, Japan
\textsuperscript{2}Southern University of Science and Technology, China
\textsuperscript{3}University of Technology Sydney, Australia

\texttt{jiangrh@csis.u-tokyo.ac.jp; yind7@outlook.com; songxuan@csis.u-tokyo.ac.jp}

\section*{ABSTRACT}
Nowadays, with the rapid development of IoT (Internet of Things) and CPS (Cyber-Physical Systems) technologies, big spatiotemporal data are being generated from mobile phones, car navigation systems, and traffic sensors. By leveraging state-of-the-art deep learning technologies on such data, urban traffic prediction has drawn a lot of attention in AI and Intelligent Transportation System community. The problem can be uniformly modeled with a 3D tensor (T, N, C), where T denotes the total time steps, N denotes the size of the spatial domain (i.e., mesh-grids or graph-nodes), and C denotes the channels of information. According to the specific modeling strategy, the state-of-the-art deep learning models can be divided into three categories: grid-based, graph-based, and multivariate time-series models. In this study, we first synthetically review the deep traffic models as well as the widely used datasets, then build a standard benchmark to comprehensively evaluate their performances with the same settings and metrics. Our study named DL-Traff is implemented with two popular deep learning frameworks, i.e., TensorFlow and PyTorch, which is already publicly available as two GitHub repositories https://github.com/deepkashiwa20/DL-Traff-Grid and https://github.com/deepkashiwa20/DL-Traff-Graph. With DL-Traff, we hope to deliver a useful resource to researchers who are interested in spatiotemporal data analysis.

\section*{CCS CONCEPTS}
- Information systems → Information systems applications; Geographic information systems; Computing methodologies → Artificial intelligence.

\section*{KEYWORDS}
traffic prediction, multivariate time-series, deep learning, ubiquitous and mobile computing, survey and benchmark

\textsuperscript{*} Equal contribution; † Corresponding author.
This work was supported by Grant-in-Aid for Early-Career Scientists (20K19859) of Japan Society for the Promotion of Science (JSPS).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

\section*{1 INTRODUCTION}
Nowadays, with the rapid development of IoT (Internet of Things) and CPS (Cyber-Physical Systems) technologies, big spatiotemporal data are being generated from mobile phones, car navigation systems, and traffic sensors. Based on such data, urban traffic prediction has been taken as a significant research problem and a key technique for building smart city, especially intelligent transportation system. From 2014 to 2017, encouraged by the huge success of deep learning technologies in the Computer Vision and Natural Language Processing field, researchers in the Intelligent Transportation System community, started to apply Long-Term Short Memory (LSTM) and Convolution Neural Network (CNN) to the well-established traffic prediction task[17, 27–29], and also achieved an unprecedented success. Following these pioneers, researchers have leveraged the state-of-the-art deep learning technologies to develop various prediction models and publish a big amount of studies on the major AI and transportation venues as listed in Table 1. Although the prediction tasks may slightly differ from each other, they can all be categorized as deep traffic models.

Figure 1: Grid-Based Traffic and Graph-Based Traffic.

No matter based on grid or graph, the traffic data illustrated in Fig.1 can be uniformly represented with a 3D tensor $\mathbb{R}^{T \times N \times C}$, where T denotes the size of the temporal domain (i.e., timeslots with constant sampling rate), N denotes the size of the spatial domain (i.e., mesh-grids or graph-nodes), and C denotes the number of information channels. For instance, assuming 300 traffic sensors are
| Grid-Based | Venue | Cite | Dataset (* means Open) | Prediction Task | Metric |
|------------|-------|------|------------------------|-----------------|--------|
| ST-ResNet[47] | AAAI17 | 867 | TaxiBJ*, BikeNYC* | Taxi In-Out Flow | RMSE  |
| DeepSD[35] | ICDER17 | 125 | Didi Taxi (Hangzhou) | Taxi Demand | MAE, RMSE |
| DMIIVT-Nef[41] | AAAI18 | 486 | Didi Taxi (Guangzhou) | Taxi Demand | MAE, MAPE |
| Periodic-GRN[54] | IJCAI18 | 57 | TaxiBJ*, TaxiSG | Taxi Density/In-Out Flow | RMSE |
| Hetero-ConvtLSTM[45] | KDD18 | 121 | Vehicle Crash Data* | Traffic Accident | MSE, RMSE, CE |
| DeepSIN[25] | AAAI19 | 53 | MobileBJ, BikeNYC-I* | Crowd/Taxi In-Out Flow | MAE, RMSE |
| STDN[40] | AAAI19 | 204 | TaxiNYC*, BikeNYC-I* | Taxi/Bike O-D Number | MAE, MAPE |
| MBL[49] | TKDE19 | 92 | TaxiBJ, BikeNYC | Taxi Transition/In-Out Flow | MAE, MAPE |
| DeepUrbanEvent[18] | KDD19 | 32 | Komatsu Toukei | Crowd Density/Flow | MSE |
| Curb-GAN[51] | KDD20 | 1 | Taxi Speed/Inflow (Shenzhen)* | Traffic Speed/Inflow | MAE, RMSE, MAPE |

| Multivariate Time-Series | Venue | Cite | Dataset (* means Open) | Prediction Task | Metric |
|-------------------------|-------|------|------------------------|-----------------|--------|
| STGCN[42] | IJCAI18 | 660 | BJER4, PeMSD7(M)*, PeMSD7(L) | Traffic Volume/Speed | MAE, RMSE, MAPE |
| DCRNN(GGGRU)[23] | ICLR18 | 691 | METR-LA*, PeMS-BAY* | Traffic Volume/Speed | MAE, RMSE, MAPE |
| Multi-graph[4] | SIGSPATIAL18 | 89 | Bike Flow (New York and Chicago) | Bike In-Out Flow | RMSE |
| ASTGCN[14] | AAAI19 | 239 | PeMSD4-I*, PeMSD8-I* | Traffic Volume/Speed | MAE, RMSE, MAPE |
| DGCNN[11] | AAAI19 | 56 | TrafficNYC, PeMS | Traffic Volume/Speed | MAE, RMSE |
| ST-MGCN[12] | AAAI19 | 182 | Bike Demand (Beijing and Shanghai) | Bike Demand | MAE, MAPE |
| Graph WaveNet[38] | IJCAI19 | 144 | METR-LA*, PeMS-BAY* | Traffic Volume/Speed | MAE, RMSE, MAPE |
| STGCN[2] | IJCAI19 | 33 | DidiNY, BikeNYC*, TaxiBJ* | Taxi/Bike Demand | MAE, RMSE, MAPE |
| T-GCN[52] | TITST19 | 195 | TaxiSZ*, METR-LA* | Traffic Volume/Speed | MAE, RMSE, Acc, R², var |
| TGC-LSTM[58] | TITST19 | 166 | Seattle-Loop*, INRIX Traffic | Traffic Volume/Speed | MAE, RMSE, MAPE |
| GCGA[44] | TITST19 | 27 | Cologne Traffic | Traffic Volume/Speed | MAE, RMSE |
| GMAN[53] | AAAI20 | 73 | Taxi Xiamen, PeMS-BAY* | Traffic Volume/Speed | MAE, RMSE, MAPE |
| MBA-BGCN[5] | AAAI20 | 28 | METR-LA*, PeMS-BAY* | Traffic Volume/Speed | MAE, RMSE, MAPE |
| STSGCN[33] | AAAI20 | 39 | PeMS03*, PeMS04*, PeMS07*, PeMS08* | Traffic Volume/Speed | MAE, RMSE, MAPE |
| SLGCN[50] | AAAI20 | 13 | METR-LA*, PeMS-BAY*, PeMSD7(M)*, BjF, BRF, BRF-L | Traffic Volume/Speed | MAE, RMSE, MAPE |
| STGNN[56] | WWW20 | 24 | METR-LA*, PeMS-BAY* | Traffic Volume/Speed | MAE, RMSE, MAPE |
| H-STGCN[9] | KDD20 | 9 | W3-715, E-2907 (Beijing) | Traffic Volume/Speed | MAE, RMSE |
| AGCRN[5] | NeurIPS20 | 9 | PeMSD4*, PeMSD8* | Traffic Volume/Speed | MAE, RMSE |
| T-MGCN[26] | TITST20 | 7 | HZJD*, PeMSD10* | Traffic Volume/Speed | MAE, RMSE, MAPE |
| DGCN[13] | TITST20 | 1 | PeMSD4*, PeMSD8*, PHILADELPHIA | Traffic Volume/Speed | MAE, RMSE |

* Citation number was referred from Google Scholar by 2021/6/13*

Table 1: Summary of The State-Of-The-Art Models

deployed to record traffic speed (channel1) and volume (channel2) every 30 minutes for 100 consecutive days, then the total data can be represented by tensor $\mathbb{R}^{4800 \times 300 \times 2}$. Besides traffic volume and speed, channels can also be used to store crowd density, taxi demand, traffic accident, car/ride-hailing order, and crowd/taxi/bike inflow and outflow. More specifically, grid-based model meshes the entire spatial domain into $H \times W$ fine-grained mesh-grids and converts the 3D representation into 4D tensor $\mathbb{R}^{T \times H \times W \times C}$. Graph-based model introduces directed or undirected graph $G = (V, E)$ to utilize the topological structure of the urban road network for modeling, where $v \in V$ is a node, $|V| = N$, and $e \in E$ is an edge. Multivariate time-series model naturally takes the N spatial units as N time-series variates and shares the same representation, i.e., $\mathbb{R}^{T \times N \times C}$ with graph-based model. Thus, the deep learning models listed in Table 1 can be divided into three groups according to the specific modeling strategy along the spatial axis.

Through the citation number in Table 1, we can know how much attention these studies have drawn in our AI and data science
community. But due to the huge amount of the related works, researchers are often too exhausted to follow up with the specific details of each model. More importantly, the evaluations on this family of models are still confusing and not well organized. For instance, some models demonstrated superior performances to the existing ones by using different datasets or metrics as shown in Table 1, while some models utilized a self-designed objective function or employed extra data sources such as Point-of-Interest (POI) data [25] or navigation app data [9] to achieve better prediction accuracy. To address the problems above, a concise but precise survey will be a great help for researchers involved in this emerging topic. But only a survey is not enough. It is also significant to conduct standard performance evaluations to examine the true function of each spatial and temporal component by using the same datasets, metrics, and other experimental settings. This paper fills these needs by providing a concise survey followed by a comprehensive benchmark evaluation on the recent deep traffic models.

We first define two benchmark tasks in Section 2, one is single-step prediction for inflow and outflow based on grid-based traffic data, another is multi-step prediction for traffic speed based on graph-based data. Second, in Section 3, we investigate both of the grid-based and graph-based datasets and pick up some open and widely used ones as our benchmark data including TaxiBJ, BikeNYC, TaxiNYC, METR-LA, PeMS-BAY, and PeMSD7M. Next, in Section 4, we decompose the models into spatial and temporal units and give the roadmap that how the models evolve along the spatial and temporal axis. Further, we draw the architectures for a bunch of representative models (e.g., ST-ResNet[47], DMVST-Net[41], STDN[40], DeepSTN-[25], STGCN[42], DCRNN[23], Graph WaveNet[38]) in an intuitive and comparative manner. Then, in Section 5, we do a comprehensive evaluation on both the grid-based and graph-based models by using the benchmark tasks and datasets under the same settings and metrics (RMSE, MAE, MAPE). In Section 6, we briefly introduce the implementation details, the availability, and the usability of our benchmark. Finally, we give our conclusion in Section 7. The contributions of our work are summarized as follows:

- We give a concise but concrete survey on the recent deep traffic models. The technique detail and the evolution are clearly summarized along spatial and temporal axes.
- We carefully select two traffic flow prediction tasks, four grid-based traffic datasets, and three graph-based traffic datasets, and implement plenty of grid/graph-based state-of-the-arts to form a complete benchmark called DL-Traff.
- On this benchmark, we conduct a comprehensive evaluation of the effectiveness and efficiency performances of the state-of-the-arts.
- Our benchmark is implemented with the two most popular deep learning frameworks, i.e., TensorFlow and PyTorch. DL-Traff is already publicly available as two GitHub repositories https://github.com/deepkashiwa20/DL-Traff-Graph and https://github.com/deepkashiwa20/DL-Traff-Grid.

With DL-Traff, (1) users can quickly grasp the technical details about the state-of-the-art deep spatiotemporal models; (2) users can smoothly reproduce the prediction results reported in this paper and use them as the baselines; (3) users can easily launch a new deep solution with either TensorFlow or PyTorch for not only traffic flow prediction tasks, but also for other spatiotemporal problems such as anomaly/accident, electricity consumption, air quality, etc.

2 PROBLEM

In this paper, we employ the following two prediction tasks into our benchmark.

(1) Grid-based inflow and outflow prediction proposed by[15, 48]. The problem is to predict how many taxis/bikes will flow into or out from each mesh-grid in the next time interval. It takes $\alpha$ steps of historical observations as input and gives the next step prediction as follows: $[X_t-(\alpha-1),...X_{t-1},X_t] \rightarrow X_{t+1}$, where $X_t \in \mathbb{R}^{H\times W\times C}$, $H, W$ are the indexes for the mesh, and $C$ is equal to 2, respectively used for inflow and outflow.

(2) Graph-based traffic speed prediction as defined in [23, 42]. In order to make a variation to the first task, we define this task as multi-step-to-multi-step one as follows: $[X_t-(\alpha-1),...X_{t-1},X_t] \rightarrow [X_{t+1}, X_{t+2},...,X_{t+\beta}]$, where $X_t \in \mathbb{R}^{N \times C}$, $\alpha/\beta$ is the number of steps of observations/predictions, $N$ is the number of traffic sensors (i.e., nodes), and $C$ is equal to 1 that only stores the traffic speed.

3 DATASET

The public datasets for urban traffic prediction are summarized in Table 2, where the reference, source, and spatial and temporal spec are enumerated. We pick up some widely used ones as our benchmark datasets.

![Figure 2: Visualization of METR-LA.](image-url)
locations, trip distances, itemized fares, driver-reported passenger counts, etc.

### 3.2 Graph-Based Traffic Dataset

**METR-LA.** This is a Los Angeles traffic data published by [23]. The data are collected from 207 highway sensors within 4 months from 2012/3/1 to 2012/6/30. A quite number of studies used this dataset as shown in Table 2. To be intuitive, a data visualization has been made as Fig.2.

**PeMS-BAY.** This is a traffic flow dataset collected from California Transportation Agencies Performance Measurement System (PeMS). It contains 325 traffic sensors in the Bay Area from 2017/1/1 to 2017/5/31. Massive studies also generate a variety of PeMS datasets by using the same source.

**PeMSD7(M)**. This traffic dataset is created and published by [42], also collected from PeMS. It covers 228 traffic sensors lasting from 2012/5/1 to 2012/6/30 with a 5-minute sampling rate on weekdays. **Summary.** The taxi and bike trip data published by Citi Bike and TLC of New York City and the traffic sensor data from PeMS of California are taken as three trustworthy and wildly-used data sources for traffic prediction. Researchers can easily access the data through the URLs listed in Table 2.

### 4 MODEL

Complex spatial and temporal dependencies are the key challenges in urban traffic prediction tasks. Temporally, future prediction depends on the recent observations as well as the past periodical patterns; Spatially, the traffic states in certain mesh-grid or graph-node are affected by the nearby ones as well as distant ones. To capture the temporal dependency, LSTM[16] and its simplified variant GRU[7] are respectively utilized by the models as shown in Table 3. In parallel with the RNNs, 1D CNN and its enhanced version TCN [43] are also employed as the core technology for temporal modeling, and demonstrate the superior time efficiency and matchable effectiveness to LSTM and GRU.

On the other hand, to capture the spatial dependency, grid-based models simply use the normal convolution operation[21] thanks to the natural euclidean property of grid spacing; graph-based models leverage the graph convolution in non-euclidean space [10, 19] by involving the adjacency relation \( A \in \mathbb{R}^{N \times N} \) between each pair of spatial units. Meanwhile, attention mechanism[34] also known as Transformer has rapidly taken over the AI community from natural language (GPT-3) to vision since 2020. Thus, attention is also introduced as base technology for modeling both spatial and temporal dependencies.

#### Table 2: Summary of The Public Traffic Datasets

| Grid-Based | Reference | Data Description / Data Source | Spatial Domain | Time Period | Time Interval |
|------------|-----------|--------------------------------|----------------|-------------|--------------|
| TaxiBJ*    | [47, 54]  | Taxi In-Out Flow / Taxi GPS Data of Beijing | 32×32 grids | 2013/7/1~2016/4/10 | Four Time Periods |
| TaxiBJ-1*  | [2]       | Taxi In-Out Flow / Taxi GPS Data of Beijing (TDrive) | 32×32 grids | 2015/2/1~2015/6/2 | 60 minutes |
| BikeNYC*   | [2, 47]   | Bike In-Out Flow / Bike Trip Data of New York City Citi Bike: https://www.citibikenyc.com/system-data | 16×8 grids | 2014/1/1~2014/9/30 | 60 minutes |
| BikeNYC-I* | [25]      | Bike In-Out Flow / Bike Trip Data of New York City | 21×12 grids | 2014/1/1~2014/9/30 | 60 minutes |
| BikeNYC-II*| [40]      | Bike In-Out Flow / Bike Trip Data of New York City | 10×20 grids | 2016/7/1~2016/8/29 | 30 minutes |
| TaxiNYC*   | [40]      | Taxi In-Out Flow / Taxi Trip Data of New York City The New York City Taxi&Limousine Commission (TLC) https://www1.nyc.gov/site/tlc/about/data.page | 10×20 grids | 2015/1/1~2015/3/1 | 30 minutes |

| Graph-Based | Reference | Data Description / Data Source | Spatial Domain | Time Period | Time Interval |
|-------------|-----------|--------------------------------|----------------|-------------|--------------|
| METR-LA*    | [36]      | Traffic Speed Sensors in Los Angeles County | 207 sensors | 2012/3/1~2012/6/30 | 5 minutes |
| PeMS-BAY*   | [20, 23]  | Traffic Speed Sensors in California Caltrans Performance Measurement System (PeMS) PeMS: http://pems.dot.ca.gov/ | 325 sensors | 2017/1/1~2017/5/31 | 5 minutes |
| PeMSD7(M)*  | [42]      | Traffic Speed Sensors in California (PeMS) | 228 sensors | 2012/5/1~2012/6/30 | 5 minutes |
| PeMS9*      | [33]      | Traffic Speed Sensors in California (PeMS) | 358 sensors | 2018/9/1~2018/11/30 | 5 minutes |
| PeMS9(4)    | [33]      | Traffic Speed Sensors in California (PeMS) | 307 sensors | 2018/1/1~2018/2/28 | 5 minutes |
| PeMS9*      | [33]      | Traffic Speed Sensors in California (PeMS) | 883 sensors | 2017/5/1~2017/8/31 | 5 minutes |
| PeMS8(M)    | [33]      | Traffic Speed Sensors in California (PeMS) | 170 sensors | 2016/7/1~2016/8/31 | 5 minutes |
| PeMS8(4-I)  | [14]      | Traffic Speed Sensors in California (PeMS) | 3848 sensors | 2018/1/1~2018/2/28 | 5 minutes |
| PeMS8(I)    | [14]      | Traffic Speed Sensors in California (PeMS) | 1979 sensors | 2016/7/1~2016/8/31 | 5 minutes |
| PeMS10*     | [26]      | Traffic Speed Sensors in California (PeMS) | 1370 sensors | 2010/12/1~2011/12/31 | 60 minutes |
| Traffic(PeMS)* | 32, 37 | Traffic Speed Sensors in California (PeMS) | 862 sensors | 2015/1/1~2016/12/31 | 60 minutes |
| LOOP-SEATTLE* | [8]    | Traffic Speed Sensors in Greater Seattle Area | 325 sensors | 2015/1/1~2015/12/31 | 5 minutes |
| TaxiSZ*     | [52]      | Taxi Speed on Roads / Taxi GPS Data of Shenzhen | 156 roads | 2015/1/1~2015/3/1 | 15 minutes |
| HZJTD*      | [26]      | Taxi Speed Sensors in Hangzhou Hangzhou Integrated Transportation Research Center | 202 sensors | 2013/10/16~2014/10/3 | 15 minutes |
Table 3: Base Technologies Employed for Spatial and Temporal Modeling

| models          | Spatial Axis | Temporal Axis | Spatial Axis | Temporal Axis |
|-----------------|--------------|---------------|--------------|---------------|
| ST-ResNet[47]   | ✓            | ✓             | ✓            | ✓             |
| DMVST-Net[41]   | ✓            | ✓             | ✓            | ✓             |
| STDN[40]        | ✓            | ✓             | ✓            | ✓             |
| DeepSTN+[25]    | ✓            | ✓             | ✓            | ✓             |
| LSTNet[20]      | ✓            | ✓             | ✓            | ✓             |
| GeoMAN[24]      | ✓            | ✓             | ✓            | ✓             |
| TPA-LSTM[32]    | ✓            | ✓             | ✓            | ✓             |
| Transformer[22] | ✓            | ✓             | ✓            | ✓             |
| ST-MetaNet[50]  | ✓            | ✓             | ✓            | ✓             |
| GMAN[53]        | ✓            | ✓             | ✓            | ✓             |
| STGCN[42]       | ✓            | ✓             | ✓            | ✓             |
| GaAN(GGRU)[46]  | ✓            | ✓             | ✓            | ✓             |
| DCRNN(GGCRU)[23]| ✓            | ✓             | ✓            | ✓             |
| Multi-graph[4]  | ✓            | ✓             | ✓            | ✓             |
| ASTGCN[14]      | ✓            | ✓             | ✓            | ✓             |
| TGCN[52]        | ✓            | ✓             | ✓            | ✓             |
| Graph WaveNet[38]| ✓           | ✓             | ✓            | ✓             |
| MTGN[37]        | ✓            | ✓             | ✓            | ✓             |
| STGNN[36]       | ✓            | ✓             | ✓            | ✓             |
| AGCRN[3]        | ✓            | ✓             | ✓            | ✓             |

We select the most representative models in Table 1 and summarize the base technologies employed by each model for spatial and temporal modeling as Table 3. On the other hand, for better understanding, we simplify and plot the network architectures in a unified manner for five grid-based models including ST-ResNet[47], DMVST-Net[41], Periodic-CRNN(PCRN)[54], STDN[40], and DeepSTN+[25] as Fig.3, and five graph-based models, namely STGCN[42], DCRNN[23], Graph WaveNet[38], ASTGCN[14], and GMAN[53] as Fig.4. Through Fig.3-4, we can easily understand how the spatial and temporal modules listed in Table 3 are assembled to form an integrated model.

Moreover, we describe how the employed technologies are evolving along the spatial and temporal axis for both grid-based and graph-based models in the next two subsections. Note that the multivariate time-series (MTS) models such as LSTNet[20], TPA-LSTM[32], GeoMAN[24], and Transformer[22] are also gradually evolving along the spatial and temporal axis. From the spatial perspective, they focus on correlation/dependence between variables; from the temporal perspective, they aim to utilize the periodic patterns occurred in time series. But due to space limitations, we don’t expand the details of those MTS models in this paper.

4.1 Roadmap for Grid-Based Model

ST-ResNet[47] is the earliest and the most representative grid-based deep learning method for traffic in-out flow prediction. It converts 4D tensor \((T,H,W,C)\) into 3D tensor \((H,W,T^*)\) by concatenating the channels at each time step so that CNN can be used to capture spatial dependency similarly to an image. Then, it creatively proposes a set of temporal features called Closeness, Period, and Trend, which correspond to the most recent observations, daily periodicity, and weekly trend respectively. Intuitively, the three parts of the features can be represented by:

\[
X_{\text{Closeness}} = [X_{t-L_c}, X_{t-(L_c-1)} , ..., X_{t-1}]
\]

\[
X_{\text{Period}} = [X_{t-L_p \times s_p}, X_{t-(L_p-1) 	imes s_p} , ..., X_{t-s_p}]
\]

\[
X_{\text{Trend}} = [X_{t-L_t \times s_t}, X_{t-(L_t-1) \times s_t} , ..., X_{t-s_t}]
\]

where \(L_c, L_p, L_t\) are the sequence length of \([\text{Closeness}, \text{Period}, \text{Trend}]\), \(s_p\) and \(s_t\) are the time span of \([\text{Period}, \text{Trend}]\), the Closeness span \(s_c\) is equal to 1 by default. This feature is not only inherited by the later grid-based models including STDN[40] and DeepSTN+[25], but also some graph-based models like ASTGCN[14], which is still regarded as the state-of-the-art temporal feature by now. To capture the long-range spatial dependency between mesh-grids, it employs Residual Learning to construct deep enough CNN networks. Additionally, it further utilizes external information including weather, event, and metadata (i.e. DayOfWeek, WeekdayOrWeekend) to auxiliary enhance spatiotemporal modeling.

Improvement along Spatial Axis. Different from ST-ResNet that takes the entire mesh-grids as input, DMVST-Net[41] and STDN[40] take one grid and its surrounding grids (i.e. SxS region) as input, thus a local CNN is enough to capture spatial dependency only among nearby grids. For the global spatial dependency, DMVST-Net introduces a weighted graph as an extra input, where nodes are the grids, and each edge represents the similarity of two time-series values (i.e. historical taxi demand) between any two grids. The graph will be manually embedded into a feature vector so that it can be concatenated with the other part. Through this, DMVST-Net gains the ability to capture long-range spatial dependency. Furthermore, STDN and [18, 49] consider the local flow information (i.e. flow from one central grid to its surrounding SxS grids) to facilitate predicting the traffic volume in the central grid, which is implemented with a flow gating mechanism in STDN and multitask learning in [18, 49]. DeepSTN+ [25] uses Point-Of-Interest (POI) data as external information (e.g., office/residential/shopping area) to take the influence of location function on the crowd/traffic flow into consideration.

Improvement along Temporal Axis. One major drawback of ST-ResNet is it does not explicitly handle the temporal axis, because it forces the video-like tensor \((T,H,W,C)\) to be converted into an image-like tensor \((H,W,T^*)\). To address this, DMVST-Net and STDN employ LSTM to connect with a separate and unshared CNN for each timestamp. STDN further considers the temporal shifting problem about periodicity (i.e. traffic data is not strictly periodic) and designs a Periodically Shifted Attention Mechanism to solve the issue. Specifically, it sets a small time window to collect \(Q\) time intervals right before and after the currently-predicting one. And the attention is used to obtain a weighted average representation \(h\) from the \(Q\) representations \([h_1, h_2, ..., h_Q]\) generated by LSTM. To this end, LSTM, and CNN work together to separately and sequentially model the spatial and temporal dependency. Convolutional LSTM [39] extends the fully connected LSTM (FC-LSTM) to have convolutional structures in both the input-to-state and state-to-state transitions and achieves a lot of successes on video modeling tasks. Motivated by this, ConvLSTM and its variant ConvGRU are
4.2 Roadmap for Graph-Based Model

STGCN[42] is one of the earliest models that use graph neural networks to predict traffic flow. Temporally, instead of RNN, it uses TCN [43] with a gated mechanism as shown in Fig. 4 to capture the dependency only from \( \text{Closeness} \). Spatially, it applies two spectral graph convolution, ChebyNet[10] and 1st-order approximation of ChebyNet [19]. TCN and GCN are stacked together as an ST-Conv block to sequentially do the spatial and temporal modeling. One major limitation of STGCN is that it uses a symmetrical adjacency matrix (i.e., undirected graph) that considers the euclidean distance between two road sensors. Thus it is difficult to model the difference of the two-way traffic flow in one road. DCRNN [23] is another pioneer to utilize graph convolution for traffic flow prediction. In contrast to the spectral convolution in STGCN, DCRNN applies a spatial graph convolution called Diffusion Convolution implemented with bidirectional random walks on a directed graph (i.e., non-symmetric adjacent matrix), so that it can capture the spatial influence from both the upstream and the downstream traffic flows. For the temporal axis, similar to ConvLSTM, it replaces the normal matrix multiplication in GRU with the proposed diffusion convolution, then a Diffusion Convolution Gated Recurrent Unit (DCGRU) is assembled that can simultaneously do the spatial and temporal modeling. With this DCGRU, it further implements an encoder-decoder structure to enable the multi-step-to-multi-step prediction. Inspired by STGCN and DCRNN, massive graph-based traffic models have been proposed as summarized in Table 1.

**Improvement along Temporal Axis.** For the temporal feature, ASTGCN[14] inherits \textit{Closeness, Period,} and \textit{Trend} from ST-ResNet, and improves STGCN that only takes \textit{Closeness}. Besides, STSGCN[33] constructs a localized temporal graph by connecting all nodes with themselves at the previous and the next steps, updating the adjacency matrix from \( \mathbf{A} \in \mathbb{R}^{N \times N} \) to \( \mathbf{A}' \in \mathbb{R}^{3N \times 3N} \), then only uses GCN to simultaneously do the spatial and temporal modeling. On the other hand, to get better ability of temporal modeling, T-GCN[52] and TGC-LSTM[8] respectively use GRU and LSTM instead of TCN to improve STGCN; GCGA[44] combines Generative Adversarial
Network (GAN) and Autoencoder with GCN; STGNN [36] adopts transformer (attention) for better global/long-term temporal modeling; STG2Seq [2] utilized GCN for temporal modeling, which is an interesting attempt.

**Improvement along Spatial Axis.** A lot of effort has been put on the spatial axis, that is the graph. (1) From single-graph to multi-graph. STGCN and DCRNN; DGCNN [11] proposes dynamic Laplacian; Adopt adaptive/learnable graph rather than a static one used in STGCN and DCRNN; DGCNN [11] proposes dynamic Laplacian.

For instance, [4] and ST-MGCN [12] consider spatial proximity, functional similarity, and road connectivity as multi-graph; MRA-BGCN [5] builds the node-wise graph according to the connectivity and competition. (2) From static graph to adaptive graph. Graph WaveNet [38], TGC-LSTM [8], and AGCRN [3] adopt adaptive/learnable graph rather than a static one used in STGCN and DCRNN; DGCNN [11] proposes dynamic Laplacian.
matrix learning through tensor decomposition; SLCNN[50] designs Structure Learning Convolution (SLC) to dynamically learn the global/local graph structure. In addition to the above, attention-augmented GCN also demonstrated better performance in terms of spatial modeling in GaAN[46], ASTGCN, and GMAN[53].

5 EVALUATION

5.1 Setting

Towards the benchmark tasks listed in Section 2, we pick up some representative models and conduct comprehensive evaluations about their actual performances. Besides the deep models, we also implement two naive baselines as follows: (1) HistoricalAverage (HA). We average the corresponding values from historical days as the prediction result; (2) CopyLastStep(s). We directly copy the last one or multiple steps as the prediction result. Our experiments were performed on a GPU server with four GeForce GTX 2080Ti graphics cards. As a benchmark evaluation, the following settings are kept the same for each model. The observation step is set to 6 for grid-based models, while the observation and prediction step are both set to 12 for graph-based models. The data ratio for training, validation, and testing is set as 7:1:2. Adam was set as the default optimizer, where the learning rate was set to 0.001 and the batch size was set to 64 by default. Mean Absolute Error is uniformly used as the loss function. The training algorithm would either be early-stopped if the validation error was converged within 10 epochs or be stopped after 200 epochs, and the best model on validation data would be saved. Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) are used as metrics, where zero values will be ignored.

5.2 Effectiveness Evaluation

The evaluation of grid-based models for single-step prediction is shown in Table 4; the evaluation of graph-based models for multi-step prediction is shown in Table 5.

**Evaluation for Grid-Based Model:** Table 4 shows that the state-of-the-art models did have advantages over the baselines (HA ∼ ConvLSTM). In particular, STDN showed better performances in general, PCRN and DeepSTN+ achieved the lowest MAE on BikeNYC-I and TaxiNYC respectively. None of these grid-based models could be acknowledged as a dominant one at the current stage. Through the experiment, we find that their main limitations are as follows: (1) ST-ResNet converts the video-like data to high-dimensional image-like data and uses a simple fusion-mechanism to handle different types of temporal dependency; (2) through the experiment, it was found PCRN took more epochs to converge and tended to cause overfitting; (3) DMVST-Net and STDN use local CNN to take grid (pixel) as computation unit, resulting in long training time; (4) DeepSTN+ utilized a fully-connected layer in ConvPlus block, which would result in a big number of parameters on TaxiBJ; (4) Multitask Learning model[18, 49] needs multiple data sources as the inputs, which hinders the applicability.

**Evaluation for Graph-Based Model:** Table 5 compares the prediction performances of our selected models at 15 minutes, 30 minutes, 60 minutes ahead on METR-LA, PeMS-BAY, PEMSD7M datasets. Through Table 5, we can find that: (1) Despite the effect of time-series model LSTNet in short-term prediction, its performance would deteriorate as the horizon gets longer; (2) Almost all of the graph-based models achieved better performance than traditional methods and time-series models on all metrics, which proved that the addition of spatial information would bring substantial performance improvements; (3) Although the models’ performances depended more or less on the dataset, the scores of DCRNN, Graph WaveNet, and MTGNN on all datasets ranked in the top 3, which also proved their robustness and versatility in traffic prediction tasks; (4) GMAN was found more prone to overfitting, due to which its performances on all three datasets were not as good as LSTNet; This is probably because GMAN adopted a global attention mechanism to capture the spatial dependency between each pair of nodes; (5) MTGNN and GraphWaveNet got most of the highest scores on different datasets and metrics. The self-adaptive/learnable graph demonstrated its great effectiveness for traffic prediction.

![Figure 5: Case Study on Graph-Based Datasets.](image-url)
datasets; (2) The graph-based models DCRNN and Graph WaveNet always outperformed the time-series model LSTNet, which proved the excellent performance of GCN in capturing spatial correlation and dependency; (3) Time lag could be observed on all of the three prediction results, especially when violent fluctuations occur in the original time series such as 2012/6/20 21:00 in PEMSD7M. This problem will be magnified in the longer-term forecast like 60 minutes lead time. Despite this, the graph-based models still show better performance in terms of volatility prediction errors, which further confirmed the effectiveness and robustness of GCN in traffic prediction.

5.3 Efficiency Evaluation

Besides comparing the effectiveness of these deep models, we also provide an analysis of their efficiency. The space complexity and time complexity of these approaches are significant for their future application, so we exhibit the number of the trainable parameters and training time of each model through bar charts Fig.6.

**Evaluation for Grid-Based Model:** From Fig.6-(a) and Fig.6-(b), we can observe that: (1) the parameter numbers of DeepSTN+ and STDN are more than other models, especially DeepSTN+, which captures the citywide spatial correlation by utilizing fully connected layer; (2) ST-ResNet has the fewest trainable parameters, and demonstrates its superiority to other models in terms of space complexity; (3) The training time of STDN and DMVST is longer than other models because they utilize the LSTM to capture the temporal dependency and take each mesh-grid as the computation unit rather than the entire mesh.

**Evaluation for Graph-Based Model:** From Fig.6-(c) and Fig.6-(d), we can conclude that: (1) STGCN and GMAN have relatively lower space complexity than others; (2) AGCRN and DCRNN need more parameters than other models because they are based on RNNs; (3) On PEMSBay, the parameter numbers of ASTGCN and MTGNN dramatically increase. The reason for this is that those two models have more GNN layers and they are more sensitive to the node number; (4) The training time of GMAN on PEMSBay outdistances other models because it applies a global attention mechanism to the entire graph (nodes). In summary, TCN-based models like STGCN and Graph WaveNet have higher computation efficiency.

6 AVAILABILITY AND USABILITY

DL-Traff is already available at GitHub as the following two repositories under the MIT License: one is for grid-based datasets/models https://github.com/deepkashiwa20/DL-Traff-Grid, and another is for graph-based datasets/models https://github.com/deepkashiwa20/DL-Traff-Graph. It is implemented with Python and the most popular deep learning frameworks: Keras[6] on TensorFlow[1] and PyTorch[31]. Fig.7 shows a use case by taking DCRNN model on METR-LA dataset as an example. To run the benchmark, the repository should be cloned locally and a conda environment with the necessary dependencies should be created. The directory is structured in a flat style and only with two levels. The traffic datasets are stored in DATA directories (e.g., METR-LA), and the python files are put in workDATA directories (e.g., workMETRLA). Entering the work directory for a certain dataset, we can find MODEL class file (e.g., DCRNN.py) and its corresponding running program named pred_MODEL.py (e.g., pred_DCRNN.py). We can run “python MODEL.py” to simply check the model architecture without feeding the training data and run “python pred_MODEL.py” to train and test the model. Additionally, Param.py file contains a variety of hyper-parameters as described in Section 5.1 that allow the experiment to be customized in a unified way. Metrics.py file contains the metric functions listed in Section 5.1. Utils.py file integrates a set of supporting functions such as pickle file reader and self-defined loss function. More details about the usability and implementation can be found at GitHub.

7 CONCLUSION

In this paper, we first survey the deep learning models as well as the widely used datasets for urban traffic prediction. Then we build a standard benchmark to comprehensively evaluate the deep traffic models on the selected open datasets. The survey and the benchmark combine together to form our study called DL-Traff, which is already available at https://github.com/deepkashiwa20/DL-Traff-Grid and https://github.com/deepkashiwa20/DL-Traff-Graph. With DL-Traff, we hope to deliver a useful and timely resource to researchers in AI and data science community.
[44] James Jian Qiao Yu and Jiatao Gu. 2019. Real-time traffic speed estimation with graph convolutional generative autoencoder. *IEEE Transactions on Intelligent Transportation Systems* 20, 10 (2019), 3940–3951.

[45] 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*. ACM, 984–992.

[46] Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit Yan Yeung. 2018. GaAN: Gated Attention Networks for Learning on Large and Spatiotemporal Graphs. In *34th Conference on Uncertainty in Artificial Intelligence* 2018. UAI 2018.

[47] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. In *AAAI*. 1655–1661.

[48] Junbo Zhang, Yu Zheng, Dekang Qi, Ruixuan Li, and Xiwen Yi. 2016. DNN-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 92.

[49] Junbo Zhang, Yu Zheng, Junkai Sun, and Dekang Qi. 2019. Flow Prediction in Spatio-Temporal Networks Based on Multitask Deep Learning. *IEEE Transactions on Knowledge and Data Engineering* (2019).

[50] Qi Zhang, Jianlong Chang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. 2020. Spatio-temporal graph structure learning for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 1177–1185.

[51] Yingxue Zhang, Yanhua Li, Xun Zhou, Xiangnan Kong, and Jun Luo. 2020. CurbGAN: Conditional Urban Traffic Estimation through Spatio-Temporal Generative Adversarial Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 842–852.

[52] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2019. T-gen: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems* (2019).

[53] 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.

[54] Ali Zonoozi, Jung-jae Kim, Xiao-Li Li, and Gao Cong. 2018. Periodic-CRN: A Convolutional Recurrent Model for Crowd Density Prediction with Recurring Periodic Patterns. In *IJCAI*. 3732–3738.