Hybrid Deep Learning Models for Traffic Prediction in Large-scale Road Networks

Ge Zheng[a], Wei Koong Chai[b], Jing-Lin Duanmu[c], Vasilis Katos[b]

[a]*Institute of Manufacturing (IfM), Department of Engineering, University of Cambridge, 17 Charles Babbage Road, Cambridge, CB3 0FS, Cambridgeshire, United Kingdom*
[b]*Department of Computing and Informatics, Bournemouth University, Fern Barrow, Poole, BH12 5BB, Dorset, United Kingdom*
[c]*Business School, University of Exeter, Rennes Dr, Exeter, EX4 4PU, Devon, United Kingdom*

Abstract

Traffic prediction is an important component in Intelligent Transportation Systems(ITSs) for enabling advanced transportation management and services to address worsening traffic congestion problems. The methodology for traffic prediction has evolved significantly over the past decades from simple statistical models to recent complex integration of different deep learning models. In this paper, we focus on evaluating recent hybrid deep learning models in the task of traffic prediction. To this end, we first conducted a review and taxonomize the reviewed models based on their feature extraction methods. We analyze their constituent modules and architectural designs. We select ten models representative of different architectural choices from our taxonomy and conducted a performance comparison study. For this, we reconstruct the selected models and performed a series of comparative experiments under identical conditions with three well-known real-world datasets collected from large-scale road networks. We discuss the findings and insights based on our results, highlighting the differences in the achieved prediction accuracy by models with different design decisions.

Keywords: Intelligent transportation system, traffic prediction, hybrid deep learning model, large-scale road networks

1. Introduction

United Nations reported that currently more than half of world's population lives in urban area and this is projected to increase to 68% by 2050 [1]. Rapid urbanization has left many countries facing various challenges in meeting the need of increasing urban citizens and sustainable development. Transportation is one such challenge. For instance, Inrix reported on average, a driver lost 134 and 133 hours in Bucharest and Bogota each year due to traffic congestion [2]. Intelligent Transportation Systems (ITSs), as an integrated transportation management system, are proposed to arrest the exacerbating traffic situations. ITSs rely on accurate traffic prediction to enable services such as Advanced Traveler Information System (ATIS) to improve traffic conditions [3].

One of the earliest work on traffic prediction published in 1979 [4] proposed to apply Auto-Regressive Moving Average (ARMA) for short-term traffic flow prediction. Since then, many work ensued and the prediction methodology has evolved over time. In this paper, we broadly synthesize the evolution of traffic prediction methodologies into three main stages. In the first (early) stage, statistical methods such as ARMA and its variants [5][6] are proposed with the problem modeled as a pure time series process. These methods are commonly used in small and relatively simple traffic systems. They are not capable of handling traffic data with high dimensions and non-linear relationships.

The second stage emerged following the popularity of machine learning models in various fields such as pattern recognition [7][8], image classification [9][10] and natural language processing [11][12]. Machine learning models with non-linear kernels and activation functions such as Support Vector Regression (SVR) [13], K-Nearest Neighbors (KNN) [14], Artificial Neural Network (ANN) [15] and Bayesian Networks [16] were then used for solving traffic prediction problem. These models mostly treat traffic prediction as multiple classification problem. Following new sensor technologies allowing collection of richer traffic data, [17] found that road traffic has complex spatial-temporal correlation and these machine learning models are shallow and inadequate for analyzing such spatial-temporal relationships.

Then, the third stage emerged where deep learning models are applied for solving traffic prediction problems. With increasing computational resource and the development of sophisticated deep learning models, the focus of the problem has also shifted from predicting traffic states at specific road

station / segment to large-scale road networks. Expanding the scope of predictions to the entire road network has made the problem more challenging since network-wide traffic data has much more complicated spatial-temporal relationship [18]. Initial deep learning models such as Recurrent Neural Networks (RNN) and its variants (Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU)) [19], Convolutional Neural Network (CNN) [20], and Graph Convolutional Neural Network (GCN) [21], are unable to fully analyze such spatial-temporal dependencies of traffic data hidden in large-scale road networks.

Following the above, the evolution of the methodology continues with hybrid deep learning models[1] emerging in the latest literature. These models are constructed by combining several deep learning models to further improve prediction accuracy. In this paper, we focus on the performance of these latest hybrid deep learning models rather than the entire evolution of the traffic prediction methodology. We point interested readers to existing surveys on previous evolution. For instance, [22, 23, 24] reviewed models focusing on addressing short-term traffic prediction only. On the other hand, [25, 26, 27] targeted more classical approaches including statistical-based and machine learning (ML)-based models while [24, 28, 29] focused on deep learning models. We also note another work [30] which reviewed traffic prediction from the perspective of smart cities and highlighted existing traffic data sources and data models for fusing traffic data in ITSs. Furthermore, [31] explored and explained different GCNs, including graph convolutional and graph attention networks, for solving various traffic prediction problems (i.e., road traffic flow and speed predictions, network-wide traffic flow and speed predictions, and traffic demand prediction) in detail. Meanwhile, [32] surveyed and categorized deep learning models for urban traffic prediction into three: grid-based, graph-based, and multivariate time-series models. The authors provided benchmarks to evaluate the performances of the selected models. We have adopted a different approach and focused on hybrid deep learning models. We create a different taxonomy (see Section 3 and further analyze and evaluate representative models of different category of our taxonomy in terms of architecture designs, Mathematical foundations and performances.

Our work here then are twofold, a specific targeted review of the hybrid

[1]In this paper, we consider "hybrid deep learning models" as inclusive of "ensemble models" which is another term used in the literature.

deep learning models and a comprehensive performance comparison study. The main contributions of this work are summarized as follows:

- We review and taxonomize hybrid deep learning models for traffic prediction in the literature.
- We analyze the common architectural choices and the constituent submodels of these hybrid models based on mathematical theories. Ten representative models are then selected for the model architecture analysis in detail based on our taxonomy.
- We conduct extensive comparative experiments on the selected models. The evaluation results presented in the literature use different settings and datasets. It is then difficult to compare their performance under common conditions. As such, we reconstruct the models and compare their performance fairly under identical experiment setup and datasets. In our experiments, we use datasets with different traffic profiles including both frequent and infrequent congestion from real road networks. We consider both short- and long-term traffic prediction tasks.
- Furthermore, to facilitate future research, we also collected and summarized publicly available traffic datasets frequently used in the literature.

The rest of this paper is organized as follows. First, in Section 2, we present a generalized traffic prediction problem formulation based on existing work. Next, in Section 3, we review the latest hybrid deep learning models from recent literature and create a taxonomy with three main types of models: 1) CNN-based models, 2) GCN-based models and 3) transformer-based models. In this section, we discuss the evolution of traffic prediction models from CNN-based models to GCN-based models. In Section 4, we synthesize the common main modules exploited in recent hybrid deep learning models and review the fundamentals of these modules individually. Then, we select ten representative hybrid deep learning models for deep architecture analysis in Section 5. Section 6 summarizes commonly used public data sources for traffic prediction to help researchers further develop more valuable works while Section 7 presents our comparative performance evaluation experiments in which we reconstructed the selected models and evaluate them under equal conditions using three large traffic datasets from real-world road networks. Finally, we conclude our work and discuss future challenges of traffic prediction problem on large-scale road networks in Section 8.

2. General Traffic Prediction Problem Formulation

Before we delve into the models, we first describe the two essential inputs and then provide a general formulation of the traffic prediction problem considered.

2.1. Road Network Data

Consider a road network, $\mathcal{G} = (\mathbf{v}, \mathbf{e})$ where $\mathbf{v}$ is the set of nodes representing sensor locations or road segments with $|\mathbf{v}| = N$ and $\mathbf{e}$ is the set of edges representing physical connectivity between sensor locations or road segments. Conventionally, $\mathcal{G}$ can be represented by the $N \times N$ symmetric adjacency matrix, $\mathbf{A} \in \mathbb{R}^{N \times N}$, with its element $\mathbf{A}_{i,j} = 1$ if there exists a link between node i and j and 0 otherwise. Further, the degree matrix of graph $\mathcal{G}$, $\mathbf{D} \in \mathbb{R}^{N \times N}$ is defined as $\mathbf{D}_{i,i} = \sum_j \mathbf{A}_{i,j}$, which sums the number of edges connected to each node. However, in the context of road networks used for traffic prediction, besides traffic states of neighboring nodes, the future traffic states of a node is also influenced by its own current state. Hence, to account for this, the road network, $\mathcal{G}$ is often represented instead by $\tilde{\mathbf{A}} = (\mathbf{A} + \mathbf{I_N}) \in \mathbb{R}^{N \times N}$ where $\mathbf{I_N}$ is the $N \times N$ identity matrix. For instance, [33, 34] use $\tilde{\mathbf{A}}$, that represents the physical connections between the node and its adjacent nodes, to analyze spatial dependencies. In some models (e.g., [35, 36]), a neighborhood considering all nodes within k hops away from the node of interest is defined using $\tilde{\mathbf{A}}^k = (\mathbf{A} + \mathbf{I_N})^k$, that represents the physical connections between the node and its $k - hop$ neighbors.

2.2. Traffic Data

Traffic data (e.g., traffic speed, traffic count) from a road network with N sensors is written as $\mathbf{x}_t = \{x_t^1, x_t^2, \ldots, x_t^i, \ldots, x_t^{N-1}, x_t^N\}; \mathbf{x}_t \in \mathbb{R}^N, (i = 1, 2, 3, \ldots, N)$, where x_t^i denotes the traffic data measured at node i at t^{th} time step. Typically, a time step can represent 5, 15, 30, 45 and 60 minutes [37]. In this paper, a time step of 5 minutes is chosen for our experiments. Then $\mathbf{X} = \{\mathbf{x}_{t-T+1}, \mathbf{x}_{t-T+2}, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t\}; \mathbf{X} \in \mathbb{R}^{T \times N}$, $(T = 1, 2, 3, \ldots)$ gives the traffic data collected from N sensors in the network for the past T time steps. Conversely, the traffic data for the future is written as $\mathbf{X}' = \{\mathbf{x}_{t+1}, \mathbf{x}_{t+2}, \ldots, \mathbf{x}_{t+T'}\} \in \mathbb{R}^{T' \times N}$ where T' is the prediction horizon. Let B be the batch size, $\mathbf{X}$ and $\mathbf{X}'$ are reshaped into $\mathbb{R}^{B \times T \times N}$ and $\mathbb{R}^{B \times T' \times N}$ respectively during model training process. Generally, traffic prediction problems can be categorized into short- ($T' < 30$ minutes) and long-term ($T' \geq 30$ minutes).

Since we compare multi-step prediction problem, we cover both timescales whereby $T' = \{1, 3\}$ for short-term and $T' = \{6, 9, 12\}$ for long-term traffic prediction corresponding to $\{5, 15\}$ minutes and $\{30, 45, 60\}$ minutes respectively [38].

2.3. General Problem Formulation

Based on graph and traffic data above, the traffic prediction problem can generally be formulated following Eq. (1) for prediction problem without explicit consideration of the graph and Eq. (2) for network-wide problems, respectively.

$$\mathbf{X}' = \mathrm{F}\Big(\mathbf{X}\Big) \tag{1}$$

$$\mathbf{X}' = \mathrm{F}\Big(\mathbf{X}; \mathcal{G}\big(\mathbf{v}, \mathbf{e}, \tilde{\mathbf{A}}(or \tilde{\mathbf{A}}^k)\big)\Big) \tag{2}$$

The objective is to learn the mapping function F(.) and compute the traffic state in the next T' time steps given the traffic data in the past T time steps and road network information $\mathcal{G}$.

3. Taxonomy

We present a summary of the hybrid deep learning models in TABLE 1 according to their main constituent models and further segregated chronologically by year of publication. We also show the prediction task (either predicting traffic flow, speed and/or occupancy), the prediction horizon considered and the dataset(s) used in these works. From our review as well as insights from [18, 39, 40, 41], existing hybrid deep learning models commonly consist of two main modules, respectively for analyzing spatial and temporal dependencies. Furthermore, these models commonly contain CNN or GCN for spatial dependency analysis and LSTM or GRU for temporal dependency analysis. To facilitate the building of the taxonomy, we further define the following:

Definition 1 (Fixed Spatial-Temporal feature). *The fixed spatial-temporal feature does* ***not*** *consider the different influences of:*

- *different road segments on the targeted road segment in space domain, and*
- *different previous time intervals on the targeted time interval in time domain.*

Definition 2 (Dynamic Spatial-Temporal feature). *The dynamic spatial-temporal feature considers both*

- *the different road segments' contributions to the targeted road segment in space domain, and/or*
- *the different contributions of previous time intervals to the targeted time interval in time domain.*

From the above, we first classify the reviewed models into three categories based on the technologies used for analyzing spatial dependencies: (1) CNN-based models, (2) GCN-based models, and (3) transformer-based models. We then further segregate them based on Definitions 1 and 2. Our taxonomy is shown in Fig. 1. In the following, we detail our taxonomy.

Table 1: Recent literature of hybrid deep learning models for traffic prediction.

Ref.	Methodology	Task	Prediction Horizon	Dataset	Year
[42]	CNN+LSTM	flow	short-term	PeMS data	2016
[43]	CNN+LSTM+Bi-LSTM	flow	short-term	PeMS data	2017
[44]	CNN+LSTM	speed	short- & long-term	GPS data	2017
[45]	CNN+LSTM	speed	short-term	GPS data	2018
[46]	Attention+CNN+GRU	flow	long-term	PeMS data	2018
[47]	CNN+LSTM	flow	long-term	Taxis GPS data	2018
[48]	CNN+LSTM+ANN	flow	short- & long-term	Mobile & Taxi data	2018
[49]	CNN+LSTM	speed	long-term	Highways' data	2018
[50]	CNN+LSTM	occupancy & flow	short- & long-term	Bike sharing & parking datasets	2019
[51]	CNN+Self-attention-LSTM	flow	short-term	Abilene & GEANT datasets	2019
[52]	CNN+LSTM	flow	long-term	Tian Chi platform data	2019
[53]	CNN+LSTM	flow	long-term	Greater Manchester data	2019
[54]	CNN+LSTM	flow	long-term	TaxiBj & BikeNYU	2019
[55]	LocalCNN+LSTM+Attention	flow	long-term	Taxi-NYU & Bike-NYU	2019
[56]	CNN+LSTM	flow	long-term	Abilene dataset	2019
[57]	CNN+LSTM	speed	long-term	Taxi GPS data	2019
[58]	CapsNet+LSTM	speed	short-term	GPS data	2020
[33]	GCN+Seq2Seq(GRU)	speed	short- & long-term	Pems-bay & Metr-la	2017
[59]	GCN+Seq2Seq	speed	short- & long-term	Baidu map traffic data	2018
[35]	GCN+LSTM	speed	short-term	Greater Seattle traffic data and INRIX GPS data	2019

[60]	GCN+Seq2Seq(RNN)	speed	short- & long-term	GPS data	2019
[61]	GCN+RNN	speed	short-term	Santander traffic data	2019
[62]	Attention-GCN +GRU	flow	short- & long-term	Melbourne data	2019
[34]	GCN+Seq2Seq(GRU) with Attention mechanism in the decoder	speed	short- & long-term	Beijing traffic data	2019
[63]	GCN+LSTM+Soft-attention	flow	short-term	PeMS data	2019
[64]	Attention-GCN+GRU (the Seq2Seq framework)	flow & speed	short- & long-term	TDrive & Metr-la	2019
[65]	GCN+GRU	speed	short- & long-term	Metr-la & Pems-Bay	2019
[66]	ARMA+GCN+LSTM	speed	short-term	GPS data	2019
[36]	GCN+GRU	speed	short- & long-term	SZ-taxi & Los-Loop	2019
[67]	Attention-GCN+LSTM	average queue length & speed	short- & long-term	Taxi GPS data	2021
[68]	Attention-GCN+Attention-LSTM	flow	short- & long-term	PeMSD4 & PeMSD8	2021
[69]	Self-attention+GCN+Seq2Seq(GRU)	speed	short- & long-term	Loop-Seattle & Metr-la	2022
[70]	spatial- & temporal-transformer	speed & flow	short- & long-term	Pems-bay & Xiamen datasets	2020
[71]	spatial- & temporal-transformer	flow	short- & long-term	PeMSD7(M) & Pems-Bay	2020

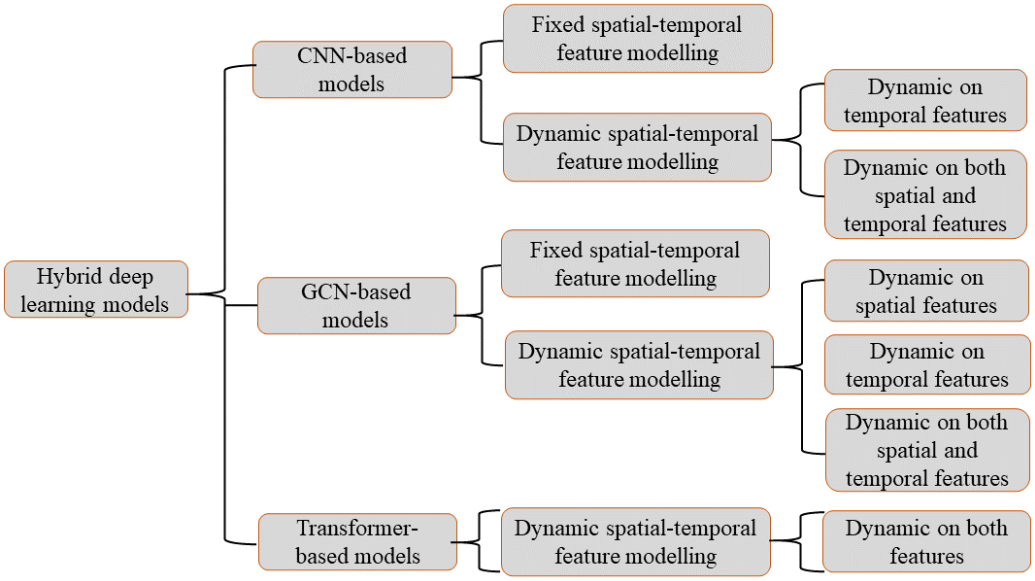


Figure 1: The classification of hybrid deep learning models for traffic prediction reviewed in this paper.

3.1. CNN-based Models

The CNN model has been applied in different fields including sentence classification [72], image classification [73], video classification [74] and human action recognition [75]. It extracts local spatial features via its convolutional kernels. CNN was introduced into traffic prediction solutions specifically for processing spatial correlation [76]. For example, [77] exploited CNN to predict traffic speed on large-scale road networks by representing traffic data as images. However, since CNN focuses on extracting spatial feature, [77] has neglected the importance of temporal features. As such, to supplement CNN, LSTM and GRU are often combined with CNN to form hybrid deep learning models for extracting temporal features (cf. TABLE 1). These CNN-based hybrid models can be further divided into two main approaches as follows:

3.1.1. Fixed Spatial-Temporal Feature Modeling

CNN-based hybrid deep learning models based on fixed spatial-temporal feature modeling consider the spatial-temporal dependencies of traffic data being fixed via sharing of parameters. Hence, the traffic states of different

neighboring road segments in the space domain are considered to have the same effect on the traffic state of the targeted road segment while, in time domain, traffic data in different previous time intervals also affect traffic state in the future time interval in the same level.

In terms of extracting fixed spatial-temporal features for traffic prediction, [42] used a 1D CNN and two LSTMs to build a hybrid deep learning model named CLTFP. The 1D CNN is used to extract inner spatial dependencies of the road network. One LSTM is used to capture short-term temporal features from previous hours while the other LSTM is utilized to extract periodic features from past days and weeks. The spatial, short-term temporal and periodic features are fused into a feature vector and then sent to a regression layer to perform predicting. Two similar models (named TreNet and U-Net) were developed by [50] and [57], respectively. Compared to [42, 50, 57] that extract spatial and temporal features using CNN and LSTM, respectively, [43] combined CNN and LSTM to generate a Conv-LSTM module for spatial-temporal feature modeling and also adopt a Bi-LSTM to analyze periodical features from historical traffic data. Considering the influence of external factors on traffic prediction, such as weather conditions and the physical characteristics of roads, [52] developed the DELA model. This model not only includes an integrated model composed of CNN and LSTM for spatial-temporal feature extraction but also contains a fully-connected layer based embedding component for learning external factors such as route structure, weather conditions and date information.

To analyze and extract more detailed features for improved prediction accuracy, [48] and [78] proposed models based on CNN and LSTM, namely STRCNs and ALLSCP respectively, for traffic flow prediction based on hourly, daily and weekly spatial-temporal feature extraction. The differences between STRCNs and ALLSCP are that the former model feeds external factors, such as weather conditions, wind and holidays, into a fully-connected layer for external feature extraction while the latter one considers different types of roads. Specifically, ALLSCP differentiates between linear roadways and road intersections and designs different input matrices for those two types roadways when considering that traffic states on linear roadways are affected by the traffic states of upstream and downstream while, for road intersections, the traffic states are affected by the traffic states of the different entrances and exits. Another model that considered the types of roads when predicting traffic states is [45] where it firstly used a Spatial-Temporal Correlation Algorithm (STCA) to identify and extract the critical road sections

and then utilized a hybrid deep learning model (named CRS-ConvLSTM NN) based on CNN and LSTM for traffic speed prediction on these critical road sections. Considering the success of CNN in the area of image recognition, some of CNN-based models proposed to convert traffic data to images and then, learn spatial-temporal features from those images. SRCNs [44] and MSTFLN [49] are two examples following this approach. Both models combine CNN and LSTM to analyze spatial-temporal features from generated traffic images. SRCNs uses CNN to learn spatial features and LSTM to learn temporal features. MSTFLN utilizes three ConvLSTM modules composed of CNN and LSTM to learn spatial-temporal features and then concatenate those features to pass to another CNN module for final prediction. Based on the encoder-decoder architecture, [53] developed a stacked autoencoder, including an encoder based on CNNs and a decoder based on Bi-LSTMs, for traffic flow and speed prediction. Another model (named STCNN) was proposed in [54] for predicting long-term traffic flow. The encoder consists of a ConvLSTM to learn the spatial-temporal traffic dependencies and a Skip-ConvLSTM to learn the periodic traffic patterns. The decoder consists of another ConvLSTM for decoding the spatial-temporal dependencies from the output of the encoder.

3.1.2. Dynamic Spatial-Temporal Feature Modeling

In reality, different neighboring road segments impact traffic state on the targeted road segment differently. Similarly, traffic states in different previous time intervals also bring different effects to the traffic state in the future. Due to these, the attention mechanism [79], that represents a major breakthrough in the natural language processing field, was introduced into traffic prediction problems. It defines different weights in space and/or time dimensions for modeling dynamic spatial-temporal dependencies to account for the non-uniform contributions of neighboring road segments and time intervals to the final prediction. CNN-based models, that exploit attention mechanism in CNN and/or LSTM (GRU), are able to analyze dynamic spatial and/or temporal features for traffic prediction.

Dynamic on Temporal Features: Wu *et al.* [46] proposed the Deep Neural Network-Based Traffic Flow prediction (DNN-BTF) model, which uses the attention mechanism to select near-term data from previous time intervals that is highly correlated to the future traffic flow and then compute the corresponding weights to previous traffic flows. The weighted traffic flows in time domain help CNN and GRU to learn spatial and dynamic temporal

features, respectively. Similar to [46], [80] also built an attention module to generate weighted past traffic data. The weighted traffic data is first sent to CNN module for spatial feature learning and then to an LSTM module for dynamic temporal feature learning. WSTNet [51] is another model that uses the attention mechanism for dynamic temporal feature extraction. It is an end-to-end deep learning model based on wavelet multi-scale analysis for network-wide traffic prediction. WSTNet firstly decomposes original traffic data into multi-level time frequency traffic matrix at different time scales by the discrete wavelet decomposition and then applied CNN for spatial feature extraction before finally sent to the LSTM with attention mechanism for dynamic temporal feature learning. Another work in [81] also joined attention mechanism with LSTM for dynamic temporal feature extraction. Specifically, [81] uses two additional LSTMs for daily and weekly periodicity analysis and this enables the model to have more temporal features for the final prediction.

Dynamic on Both Spatial and Temporal Features: Yao *et al.* [55] proposed the Spatial-Temporal Dynamic Networks (STDN) to analyze dynamic spatial dependencies by a CNN-based Flow Gating Mechanism (FGM) and dynamic temporal dependencies by integrating the self-attention mechanism into LSTM, for traffic prediction. On the other hand, [82] developed a Convo-Recurrent Attentional Neural Network (CRANN) model for traffic prediction. The idea behind CRANN is the use of classical time-series decomposition in which CRANN consists of several modules to exploit different patterns or characteristics of traffic data and then aggregates them to make predictions. Dynamic spatial and dynamic temporal features are separately extracted by a dedicated attention-based spatial and an attention-based temporal module and then concatenated with exogenous data (i.e. weather condition) via a fully-connected layer for the final prediction.

3.2. GCN-based Models

The CNN-based models in Section 3.1 consider road networks as regular grids and traffic data with regular Euclidean structure. In fact, road networks are inherently irregular and traffic data should be treated as non-Euclidean data [83]. Therefore, GCN with the advantage of dealing with non-Euclidean structured data has been introduced into the task of traffic prediction. Similar to CNN-based models, GCN-based models can be also classified into two: fixed and dynamic.

3.2.1. Fixed Spatial-Temporal Feature Modeling

Zhao *et al.* [36] proposed a Temporal Graph Convolutional Network (T-GCN) model, which combines GCN and GRU for traffic speed prediction. GCN is used to learn complex topological structures from the $k-hop$ neighborhood matrix for capturing spatial dependencies and GRU is utilized to learn changes of traffic data along the time dimension for capturing temporal dependencies. To enrich traffic information, [35] proposed the Traffic Graph Convolutional Long Short-Term Memory Neural Network (TGC-LSTM) model, based on GCN and LSTM, to learn the interactions of road segments in a large-scale road network from all $k-hop$ neighborhood matrices. A L1-norm on graph convolution weights and a L2-norm on graph convolution features are added to the loss function for enhancing the interpretability of the model.

Following the proposal of the sequence-to-sequence (Seq2Seq) architecture [84] that is capable of dealing with long-term sequence problems, several hybrid deep learning models started to exploit it for analyzing long-term traffic dependency. For example, [33] developed a deep learning framework based on Diffusion Convolutional Recurrent Neural Network (DCRNN) under the Seq2Seq framework for traffic speed prediction. Both the encoder and the decoder inside the Seq2Seq framework consist of GRUs embedded by the diffusion convolutional process. Two other hybrid models exploiting the Seq2Seq framework are [59] and [60]. The hybrid model in [59] incorporates offline geographical and social attributes, spatial dependencies and online crowd queries with a deep fusion. The model in [60] considers temporal attributes including public holidays, working days, peak hours and off-peak hours for contributing to final prediction in the decoder of the Seq2Seq framework. Pan *et al.* [64] developed a deep-meta-learning model named ST-MetaNet under the Seq2Seq framework to predict network-wide traffic. Both the encoder and the decoder in ST-MetaNet have the same network structure consisting of four components: 1) basic RNN for learning long temporal dependencies, 2) Meta-knowledge learner for learning the meta-knowledge of nodes and edges from node and edge attributes respectively, 3) Meta-GAT for capturing diverse spatial correlations by individually broadcasting locations' hidden states along edges, and 4) Meta-RNN for capturing diverse temporal correlations associated with the geographical information of locations.

3.2.2. Dynamic Spatial-Temporal Feature Modeling

Fixed GCN-based models address the traffic prediction problem as a fixed spatial-temporal process. Similarly to dynamic CNN-based models, we also find dynamic GCN-based models in the literature that treat the traffic prediction problem as a dynamic spatial-temporal process via introduction of the attention mechanism, due to considering the fact that different neighboring sensors or road segments and different previous time steps individually affect differently the targeted road segment and the future time intervals, respectively.

Dynamic on Spatial Features: The SAGCN-SST model [69] that is designed for multi-interval network-wide traffic speed prediction falls within this category. It combines the attention mechanism into GCN layers for analyzing dynamic spatial dependencies in different neighborhoods and uses GRU under an encoder-decoder architecture for analyzing temporal features. The Dynamic Graph Convolutional Recurrent Network (DGCRN) model [85] is another model in this category. In DGCRN, the node embedding technology [70] is used to embed the node attributes and then sent to a hypernetwork to generate dynamic graph at each time interval. GCN and GRU under an encoder-decoder architecture are then utilized to respectively capture spatial and temporal features based on generated dynamic graphs and their historical traffic data. Meanwhile, [86] built a graph neural network architecture, named Graph WaveNet, to exploit dynamic and hidden spatial features by using a novel adaptive dependency matrix. It captures long-term temporal features by temporal convolution layers consisting of the dilated causal convolution [87].

Dynamic on Temporal Features: Li *et al.* [63] proposed a graph and attention-based long short-term memory network (named GLA), to capture the spatial-temporal features of traffic flow data. GLA uses GCN to mine the spatial relationships of traffic data, and then the output of GCN is fed to LSTM with the soft attention mechanism for the dynamic temporal feature extraction. Another model, the AGC-Seq2Seq-Att [34], also joins the attention mechanism into LSTM for dynamic temporal feature analysis and uses GCN for spatial feature analysis on the adjacent matrix. He *et. al.* [88] built a Graph Attention Spatial-Temporal Network (GASTN) model for citywide mobile traffic prediction. Compared to [63] and [34], [88] adopts Dynamic Time Warping (DTW) algorithm [89] to calculate the similarities of traffic data between two nodes, and then clusters nodes into different groups and

builds weighted graph based on computed similarities of these groups, before using GCN and attention-based RNN to extract spatial and dynamic temporal features.

Dynamic on Both Spatial and Temporal Features: Shi *et. al.* [90] designed an Attention-based Periodic-Temporal Neural Network (APTN) to learn dynamic spatial and dynamic temporal features for traffic flow prediction. APTN is built based on an encoder-decoder architecture, in which an LSTM-based encoder with the attention mechanism is used to capture dynamic spatial correlations of each node with the entire graph by learning individual weights and the other LSTM-based decoder with the attention mechanism is utilized to decode the encoded information. It also adaptively select the relevant encoder hidden states with individual weights to produce the output. The Spatial and Temporal Attention based Neural Network (STANN) [62] using an encoder-decoder architecture based on CNN on traffic graphs and GRU is another work in this category. The difference of this work compared to [90] is that the spatial attention matrices used to represent the relationships of road segments in a traffic network are generated before using the STANN model with attention mechanism to extract both dynamic spatial and dynamic temporal features. Yin *et. al.* [68] proposed a Multi-stage Attention Spatial-Temporal Graph Network (MASTGN) that captures dynamic temporal features via the interactions among multiple time series by an internal attention mechanism and extracts dynamic spatial features by a dynamic neighborhood-based attention mechanism.

3.3. Transformer-based Models

Inspired by the newly proposed transformer in [79] for efficiently modeling long-range dependencies in natural language processing, researchers have introduced the transformer architecture to replace CNN (or GCN) for spatial correlation analysis and LSTM (or GRU) for temporal dependency analysis. Self-attention-based transformer can directly learn dynamic spatial and temporal dependencies by distributing different weights on neighbors in space dimension and on previous time intervals in time dimension for contributing to the targeted road segment. Besides, transformer can exploit more hidden information in traffic data by defining multi-heads and accelerate training phase by paralleling process.

3.3.1. Dynamic Spatial-Temporal Feature Modeling

Dynamic on Both Spatial and Temporal Features: Xu *et al.* [71] developed a Spatial-Temporal Transformer Networks (STTNs) to improve the accuracy of long-term traffic prediction, which consists of spatial transformer for modeling dynamic spatial dependencies with self-attention mechanism and temporal transformer for modeling dynamic long-range temporal dependencies across previous time intervals. Another transformer-based model (named GMAN) was developed under an encoder-decoder architecture in [70]. Both encoder and decoder consist of multiple spatial-temporal attention blocks to model the impact of the spatial-temporal factors on traffic state. In addition, it also includes a spatial-temporal embedding to encode vertices into vectors using the *node2vec* approach [91] for the vertex representation learning.

4. Common Models Used in Hybrid Deep Learning Models

Hybrid deep learning models reviewed in Section 3 have strong ability to exploit more features in both space and time domains compared to single models. Here, we summarize common modules used in these hybrid deep learning models and categorize them into three groups based on their abilities to extract features in space and/or time domain.

4.1. Common Models for Spatial Feature Extraction

4.1.1. Convolution Neural Network (CNN)

Convolutional Neural Network (CNN) is a type of neural networks inspired by biological processes where the neuron connectivity pattern resembles the organization of animal visual cortex [92]. It was initially used for image recognition where each neuron extracts features only in a restricted region of the image by the filters that are able to find relationships between neighboring inputs. In terms of addressing traffic prediction problem, most existing works firstly integrate historical traffic data into the shape of grid data based on the locations of sensors or road segments, and then learn it like image data. The spatial features can be extracted by implementing convolutional operation on the restricted region of the traffic image data by using Eq. (3).

$$\mathbf{CN} = \text{pool}(\text{ReLU}(\mathbf{W}_{cn}^{j}\mathbf{X} + \mathbf{b}_{cn}^{j})); j = 1, ..., C \tag{3}$$

where $\mathbf{W}_{cn}^{j} \in \mathbb{R}^{c1 \times c2}$ is the parameter matrix of the j^{th} filter with the kernel size $c1 \times c2$ and C is the number of filters. $\mathbf{b}_{cn}^{j}$ is the bias of the j^{th} filter. ReLU is the activation function, and pool is pooling layer. **CN** is the output of this CNN layer.

4.1.2. Graph Convolution Neural Network (GCN)

Graph convolutional neural networks (GCNs) were developed to analyze and learn non-Euclidean data in the space domain, and has been applied in different problems including classification in citation networks [93], syntax-aware neural machine translation [94], 3D human pose regression [95], traffic prediction [96], etc. GCNs used for solving traffic prediction problem include spectral GCN, diffusion GCN, and traffic GCN.

Spectral GCN [97], implements convolution operation on graph data from the spectral domain by eigendecomposition of the Laplacian matrix $\mathbf{L} = (\mathbf{D} - \mathbf{A}) \in \mathbb{R}^{N \times N}$[2] where $\mathbf{D}$ is the diagonal degree matrix and $\mathbf{A}$ is the adjacency matrix. Based on this, the spectral GCN can be defined as Eq. (4):

$$\mathbf{GC}_{spectr} = (\mathbf{U}\mathbf{g}_{\boldsymbol{\theta}}\mathbf{U}^{T})\mathbf{X} = \mathbf{U}diag(\boldsymbol{\theta})\mathbf{U}^{T}\mathbf{X} \tag{4}$$

where $\mathbf{U} \in \mathbb{R}^{N \times N}$ is the eigenvectors of $\mathbf{L}$ and $\mathbf{U}^{T}$ is the transpose of $\mathbf{U}$. $\mathbf{g}_{\boldsymbol{\theta}} = diag(\boldsymbol{\theta})$ is the filter parameterized by $\boldsymbol{\theta} \in \mathbb{R}^{N}$ and $\mathbf{GC}_{spectr} \in \mathbb{R}^{B \times T \times N}$ is the output of spectral GCN.

Diffusion GCN was proposed in [33] and performed on a directed graph. Diffusion GCN models the bidirectional diffusion process, which enables the model to capture the influence of upstream and downstream traffic. The diffusion GCN can be defined as Eq. (5) and Eq. (6):

$$\mathbf{X}_{t *_{\mathcal{G}} f_{\boldsymbol{\theta}_d}} = \sum_{k=0}^{K-1} \Big(\boldsymbol{\theta}_{k,1} (\mathbf{D}_o^{-1}\mathbf{A}_w)^{k} + \boldsymbol{\theta}_{k,2} (\mathbf{D}_I^{-1}\mathbf{A}_{w}{}^{T})^{k} \Big) \mathbf{x}_t \quad for\ t = 1, ...T \tag{5}$$

$$\mathbf{DGC} = \sigma\Big(\sum_{t=1}^{T} \mathbf{X}_{t *_{\mathcal{G}} f \boldsymbol{\Theta}_{d_{t;t'}}} \Big) for\ t' = 1, ...T' \tag{6}$$

[2]Note that some works use the normalized graph Laplacian matrix $\mathbf{L} = (\mathbf{I}_N - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}) \in \mathbb{R}^{N \times N}$).

where $\mathbf{D}_o$ and $\mathbf{D}_I$ are the out-degree and in-degree diagonal matrices respectively, and $\boldsymbol{\theta}_{k,1} \in \mathbb{R}^K$ and $\boldsymbol{\theta}_{k,2} \in \mathbb{R}^K$ are the corresponding weight matrices. $\mathbf{A}_w$ is the weighted adjacent matrix and ${\mathbf{A}_w}^T$ is the transpose of $\mathbf{A}_w$. $\boldsymbol{\theta}_d \in \mathbb{R}^{K\times 2}$ are the parameters for the filter while $(\mathbf{D}_o^{-1}\mathbf{A}_w)$ and $(\mathbf{D}_I^{-1}{\mathbf{A}_w}^T)$ represent the transition matrices of the diffusion process and the reverse one, respectively. $\mathbf{X}_{t*_{\mathcal{G}} f_{\boldsymbol{\theta}_d}}$ presents spatial features extracted from $k-hop$ neighborhoods. $\mathbf{DGC} \in \mathbb{R}^{B\times T'\times N}$ is the output of diffusion GCN and σ is the activation function (e.g., Sigmoid or ReLU). $\mathbf{f\Theta}_{d_{t;t'}}$ are the filters and their parameters are $\boldsymbol{\Theta} \in \mathbb{R}^{T'\times T\times K\times 2}$.

Traffic GCN in [35] directly conducts convolutional operation on the road graph via Eq. (7).

$$\mathbf{GC} = \sigma\Big((\mathbf{W}_{gc} * \tilde{\mathbf{A}}^k)\mathbf{X}\Big) \tag{7}$$

where $\mathbf{W}_{gc} \in \mathbb{R}^{N\times N}$ is a trainable weight matrix and $\mathbf{GC} \in \mathbb{R}^{B\times T\times N}$ is the output of GCN. σ is the activation function such as ReLU [98] and $*$ is the Hadamard product operator.

4.2. Common Models for Temporal Feature Extraction

4.2.1. Long Short-Term Memory (LSTM)

LSTM is one of the most commonly used model for extracting temporal features. It is an extension of the RNN model [99]. Compared to RNN that has one part (i.e., the tanh layer), LSTM consists of four parts: three gates (namely, input gate $\mathbf{i}_t$, output gate $\mathbf{o}_t$ and forget gate $\mathbf{f}_t$) and a cell state ($\mathbf{c}_t$). The forget gate $\mathbf{f}_t$ with a sigmoid layer σ_g firstly determines the part of information in current traffic data $\mathbf{x}_t$ and in the last hidden state $\mathbf{h}_{t-1}$ that needs to be forgotten and update this to the cell state $\mathbf{c}_t$ via Eq. (8). To supplement the forgotten information by the gate $\mathbf{f}_t$, the input gate $\mathbf{i}_t$ with a sigmoid layer σ_g is used to decide the information from current traffic data $\mathbf{x}_t$ to be added into the cell state $\mathbf{c}_t$ via Eq. (9). Then, the cell state $\mathbf{c}_t$ is updated using the tangent layer σ_c (cf. Eq. (11)) for integrating the traffic information provided from the forget gate, the input gate and the last cell state $\mathbf{c}_{t-1}$. Meanwhile, the output gate with a sigmoid layer σ_g selects previous information remembered by $\mathbf{h}_{t-1}$ and the current information $\mathbf{x}_t$ by Eq. (10) for contributing to final output. Finally, the predicted result is computed by combining remembered information from the output gate $\mathbf{o}_t$ and the cell state $\mathbf{c}_t$ with a tangent layer σ_h by Eq. (12).

$$\mathbf{f}_t = \sigma_g(\mathbf{w}_f \times \mathbf{x}_t + \mathbf{u}_f \times \mathbf{h}_{t-1} + \mathbf{b}_f) \tag{8}$$

$$\mathbf{i}_t = \sigma_g(\mathbf{w}_i \times \mathbf{x}_t + \mathbf{u}_i \times \mathbf{h}_{t-1} + \mathbf{b}_i) \tag{9}$$

$$\mathbf{o}_t = \sigma_g(\mathbf{w}_o \times \mathbf{x}_t + \mathbf{u}_o \times \mathbf{h}_{t-1} + \mathbf{b}_o) \tag{10}$$

$$\mathbf{c}_t = \mathbf{f}_t * \mathbf{c}_{t-1} + \mathbf{i}_t * \sigma_c(\mathbf{w}_c \times \mathbf{x}_t + \mathbf{u}_c \times \mathbf{h}_{t-1} + \mathbf{b}_c) \tag{11}$$

$$\mathbf{h}_t = \mathbf{o}_t * \sigma_h(\mathbf{c}_t). \tag{12}$$

where $\mathbf{w}_f, \mathbf{w}_i, \mathbf{w}_o$ and $\mathbf{w}_c$ are the weight matrices of the forget gate, the input gate, the output gate and the cell state respectively while $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_o$ and $\mathbf{b}_c$ are the corresponding bias for each gate and state. Furthermore, $\mathbf{u}_f, \mathbf{u}_i, \mathbf{u}_o$ and $\mathbf{u}_c$ are the weight matrices of the last hidden state $\mathbf{h}_{t-1}$. σ_g is used to denote a sigmoid function ($= \frac{1}{1+e^{-x}}$) in the three gates and the operator $*$ denotes Hadamard product. σ_c and σ_h are hyperbolic tangent functions ($\tanh(x)$) for the cell state and the final output.

4.2.2. Gated Recurrent Unit (GRU)

GRU [100] is developed based on LSTM. It incurs shorter processing time and less Central Processing Unit (CPU) cycles. The reason is that GRU combines LSTM's forget and input gates into a single "update gate", and merges the memory cell and hidden state. This makes GRU simpler than the standard LSTM but still effective. GRU consists of three parts: the update gate $\mathbf{z}_t$, the reset gate $\mathbf{r}_t$ and the hidden state $\mathbf{h}_t$. The update gate, $\mathbf{z}_t$, extracts the long-term dependency of the traffic data. It decides how much information it needs to update from the input $\mathbf{x}_t$ and the hidden state at the previous time step $\mathbf{h}_{t-1}$ (cf. Eq. (13)). The reset gate, $\mathbf{r}_t$, captures the short-term dependency of traffic features. It decides how much information from the hidden state at the previous time step is retained for updating the current hidden state. It is computed in a similar manner as the update gate via Eq. (14). Then, the input $\mathbf{x}_t$, the reset gate $\mathbf{r}_t$ and the hidden state at the previous time step $\mathbf{h}_{t-1}$ are used to activate the candidate hidden state $\tilde{\mathbf{h}}_t$ via Eq. (15).

$$\mathbf{z}_t = \sigma(\mathbf{w}_z \times \mathbf{x}_t + \mathbf{u}_z \times \mathbf{h}_{t-1} + \mathbf{b}_z) \tag{13}$$

$$\mathbf{r}_t = \sigma(\mathbf{w}_r \times \mathbf{x}_t + \mathbf{u}_r \times \mathbf{h}_{t-1} + \mathbf{b}_r) \tag{14}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{w}_h \times \mathbf{x}_t + \mathbf{u}_h \times (\mathbf{r}_t * \mathbf{h}_{t-1}) + \mathbf{b}_h) \tag{15}$$

where $\mathbf{w}_z$, $\mathbf{w}_r$ and $\mathbf{w}_h$ are the weight matrices of the update gate, the reset gate and the candidate hidden state respectively while $\mathbf{b}_z$, $\mathbf{b}_r$ and $\mathbf{b}_h$ are the corresponding bias for each gate and state. Furthermore, $\mathbf{u}_z$, $\mathbf{u}_r$ and $\mathbf{u}_h$ are the weight matrices of the hidden state at the previous time step $\mathbf{h}_{t-1}$ in the update gate, the reset gate and the candidate hidden state, respectively. Finally, the current hidden state can be calculated using the update gate $\mathbf{z}_t$, the hidden state at the previous time step $\mathbf{h}_{t-1}$ and the current candidate hidden state $\tilde{\mathbf{h}}_t$ using Eq. (16).

$$\mathbf{h}_t = (\mathbf{1} - \mathbf{z}_t) * \mathbf{h}_{t-1} + \mathbf{z}_t * \tilde{\mathbf{h}}_t \tag{16}$$

4.2.3. Sequence-to-Sequence (Seq2Seq) Architecture

Seq2Seq [84] has been found to offer good performance in the area of natural language processing. It consists of an encoder and a decoder with a context connecting the two. For traffic prediction problem, it encodes the spatially-fused time series using Eq. (17).

$$\mathbf{h}^e_{t-t_e} = \begin{cases} \mathrm{f}_{\mathrm{enc}}(\mathbf{h}^e_0, \mathbf{x}_{t-t_e}), & t_e = T \\ \mathrm{f}_{\mathrm{enc}}(\mathbf{h}^e_{t-t_e-1}, \mathbf{x}_{t-t_e}), & t_e \in 0, \ldots, T-1 \end{cases} \tag{17}$$

where $\mathbf{h}^e_{t-t_e}$ is the hidden state in the encoder at $(t-t_e)^{th}$ time step. The initial hidden state is $\mathbf{h}^e_0$. The hidden state $\mathbf{h}^e_{t-t_e-1}$ at $(t-t_e-1)^{th}$ time step and the spatially-fused time series $\mathbf{x}_{t-t_e}$ at $(t-t_e)^{th}$ time step are used to calculate the hidden state $\mathbf{h}^e_{t-t_e}$ at $(t-t_e)^{th}$ time step.

The hidden state $\mathbf{h}^e_t$ ($t_e = 0$) at the t^{th} time step is considered as the context vector $\mathbf{C}$ (cf. Eq. (18)) which encodes all information from the input $\mathbf{X}$ in the encoder.

$$\mathbf{C} = \mathbf{h}^e_t \tag{18}$$

In the decoder, the context vector $\mathbf{C}$ as the initial hidden state $\mathbf{h}^d_0$ is decoded to the target sequence. The hidden state $\mathbf{h}^d_{t+t_d-1}$ at $(t+t_d-1)^{th}$ time step and the target traffic speed $\mathbf{x}_{t+t_d}$ at $(t+t_d)^{th}$ time step are utilized to calculate the hidden state $\mathbf{h}^d_{t+t_d}$ at $(t+t_d)^{th}$ time step. The hidden state $\mathbf{h}^d_{t+t_d}$ at $(t+t_d)^{th}$ time step in the decoder is considered as the final prediction $\tilde{\mathbf{x}}_{t+t_d}, t_d = 1, \ldots, T'$.

4.3. Common Models for Spatial-Temporal Feature Extraction

4.3.1. Attention Mechanism

In field of neural machine translation [101], to improve the accuracy of natural language translation with increasing length of input sentence, [102] developed an extension of encoder–decoder model. It searches for a set of positions in a source sentence where most relevant information is concentrated, and then aligns those most relevant positions to the targeted positions. This method solves the long-term dependencies by measuring the similarity of the input sentence at each observed position and the output sentence at the targeted position. This approach has evolved and been applied to address long-term dependency challenges in different areas such as video captioning [103], image classification [104], speech recognition [105], traffic flow prediction [106], etc. For solving traffic prediction problem, the attention mechanism is usually used for strengthening the effect of important features while weakening the unimportant ones in space and/or time domains towards to final prediction. We briefly elaborate below the basic working principle of the attention mechanism in the decoder of the Seq2Seq framework [34] as an example.

The similarity of traffic data between observed time step t and targeted time step t', $\mathbf{u}_{t,t'}$, is computed via a tanh function as given by Eq. (19).

$$\mathbf{u}_{t,t'} = \mathbf{q}^T \tanh\Big(\mathbf{h}_t^e \mathbf{w}_{t,t'} \mathbf{h}_{t'}^d\Big); t = 1, 2, \ldots, T \tag{19}$$

where $\mathbf{w}_{t,t'}$ is a trainable weight matrix and $\mathbf{q}^T$ is the transposition/reshaping operations to adjust the dimensions. We then compute the attention weights as probabilities (i.e., $\mathbf{a}_{t,t'} \in [0.0, 1.0]$) via a softmax function given in Eq. (20).

$$\mathbf{a}_{t,t'} = \text{softmax}\Big(\mathbf{u}_{t,t'}\Big) = \frac{\exp\big(\mathbf{u}_{t,t'}\big)}{\sum_{t=1}^{T} \exp\big(\mathbf{u}_{t,t'}\big)} \tag{20}$$

The attention weights $\mathbf{a}_{t,t'}$ are mapped to hidden state $\mathbf{h}_t^e$ for achieving targeted traffic state by Eq. (21).

$$\mathbf{x}_{t'} = \sum_{t=1}^{T} \mathbf{a}_{t,t'} \times \mathbf{h}_t^e. \tag{21}$$

4.3.2. Transformer

Transformer [79] makes use of the self-attention mechanism under the encoder and decoder architecture. It has been found in [79] to outperform

the Google Neural Machine Translation model in specific tasks. Based on such success, transformer is applied into solving network-wide traffic prediction problem by using self-attention mechanism to capture dynamic-spatial and dynamic-temporal dependencies (e.g., [71][70]). For a single-head self-attention mechanism used to analyze dynamic-spatial dependencies, the input consists of queries $\mathbf{q}^s \in \mathbb{R}^{N \times d_q}$ of dimension d_q, keys $\mathbf{k}^s \in \mathbb{R}^{N \times d_k}$ of dimension d_k, and values $\mathbf{v}^s \in \mathbb{R}^{N \times d_v}$ of dimension d_v, and they are computed by Eq. (22).

$$\begin{aligned} \mathbf{q}^s &= \mathbf{w}_q^s \mathbf{x}_t \\ \mathbf{k}^s &= \mathbf{w}_k^s \mathbf{x}_t \\ \mathbf{v}^s &= \mathbf{w}_v^s \mathbf{x}_t \end{aligned} \tag{22}$$

where $\mathbf{w}_q^s$, $\mathbf{w}_k^s$ and $\mathbf{w}_v^s$ are learnable weight matrices for $\mathbf{q}^s$, $\mathbf{k}^s$, and $\mathbf{v}^s$, respectively, and its random initial values are updated by the back propagation. After obtaining the three high dimensional spatial features, dynamic-spatial dependencies $\mathbf{S}^s \in \mathbb{R}^{N \times N}$ are calculated by dot-product using Eq. (23)

$$\mathbf{S}^s = \text{softmax}(\frac{\mathbf{q}^s (\mathbf{k}^s)^T}{\sqrt{d_k}}) \tag{23}$$

where $\mathbf{S}^s$ is the learned dynamic dependency matrix and defined by how each sensor is influenced by all the other sensors in the road network. Then the new spatial features $\mathbf{M}^s \in \mathbb{R}^{N \times N}$ are updated via Eq. (24).

$$\mathbf{M}^s = \mathbf{S}^s \mathbf{v}^s \tag{24}$$

Note that multiple spatial dependencies can be learned with multi-heads attention mechanism by learning multiple pairs so as to reveal different hidden spatial dependencies from various latent spaces. Furthermore, two feed-forward neural network layers with non-linear activation are used to improve the prediction ability, and its output is added to the input of the self-attention mechanism to build the residual connection for stable training using Eq. (25).

$$\mathbf{Y}^s = \text{ReLU}(\mathbf{W}_1^s \text{ReLU}(\mathbf{W}_0^s \mathbf{M}^s)) + \mathbf{X} \tag{25}$$

where $\mathbf{W}_0^s$ and $\mathbf{W}_1^s$ are weight matrices of two feed-forward neural network layers. Similar to the analysis of dynamic-spatial dependencies $\mathbf{Y}^s$, dynamic-temporal dependencies $\mathbf{Y}^t$ can be obtained by the same idea. To

fuse dynamic-spatial and dynamic-temporal features, the gated mechanism (cf. Eq. (26)) is introduced based on the gated mechanism in GRU and the final output is calculated by Eq. (27).

$$\mathbf{Z} = \text{Sigmoid}(\mathbf{Y}^s\mathbf{W}_{zs} + \mathbf{Y}^t\mathbf{W}_{zt} + \mathbf{b}_z) \tag{26}$$

$$\mathbf{X}' = \mathbf{Z} \odot \mathbf{Y}^s + (\mathbf{1} - \mathbf{Z}) \odot \mathbf{Y}^t \tag{27}$$

where $\mathbf{W}_{zs}$ and $\mathbf{W}_{zt}$ are weight matrices of the dynamic-spatial and dynamic-temporal features respectively and $\mathbf{b}_z$ is the bias term. $\mathbf{Z}$ is the gate used to fuse both dynamic features and $\mathbf{X}'$ is the output.

5. Hybrid Deep Learning Model Architecture

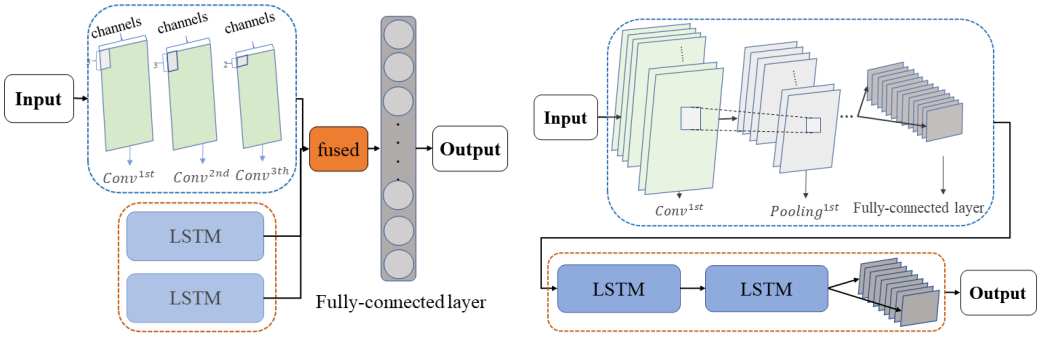
In this section, we review and analyze the architecture of the various hybrid deep learning models for traffic prediction. Based on structures of the models (i.e., how the constituent modules are interconnected), we categorized them into three main architecture designs: (1) paralleled, (2) stacked, and (3) hybrid. TABLE 2 presents the hybrid deep learning models based on their architectural choices. From the table, it can be observed that most of hybrid deep learning models adopt the stacked architecture. The following subsections will present and discuss architectures from these three groups in detail.

Table 2: Categories of architectures for CNN-, GCN- and Transformer-based models

Methodology	Paralleled	Stacked	Paralleled & Stacked
CNN-based models	[42][46][50][52] [57][58][78][82]	[44][45][47][48] [51][53][54][55] [56][80]	[43][49][81]
GCN-based models	[107]	[33][34][35][36] [59][60][62][63] [64][65][68][85] [88][90][108][109] [110][111][112]	[61][66][67][69]
Transformer-based models	[70][113]	[71][114][115]	[116]

5.1. The Paralleled Architectures

The paralleled architecture has the modules extract the spatial and temporal features separately and in parallel before merging the individual outputs for final prediction. Fig. 2(a) presents an example paralleled architecture. This architecture is proposed in [42] and in this design, a CNN and two LSTMs separately extract spatial, short-term temporal and long-term temporal features in parallel. These extracted features are then fused before sent to a fully-connected layer for computing its final prediction. Similarly to [42], [78] designed a hybrid deep learning model in paralleled architecture to analyze and extract temporal-spatial features for traffic flow prediction. When compared to [42], [78] has ability to analyze more detailed and specific features. For example, [78] uses ARIMA and two LSTMs to focus on short-, middle- and long-term temporal feature extraction while utilizes SAE and CAPSNET for global- and local-spatial feature capture separately before sending those extracted features to a fully-connected layer for final prediction.



(a) Paralleled (b) Stacked

Figure 2: The architectures of CLTFP (paralleled) and SRCNS (stacked).

5.2. The Stacked Architectures

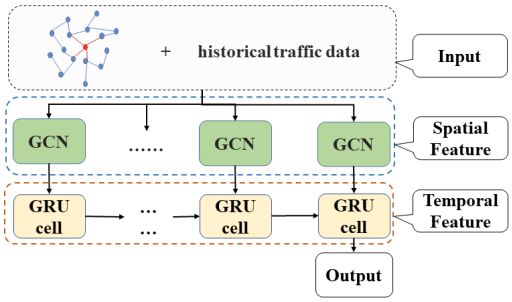
In contrast to the paralleled architecture, the stacked architecture refers to those models that has its modules deal with different patterns and/or features of traffic data serially. Fig. 2 (b) shows an example stacked architecture, named SRCNs, from [44]. Although it also used the same constituent models (i.e., the CNNs and LSTMs like Fig. 2 (a)), it has adopted a different architectural choice – i.e., the SRCNs arranged its constituent modules in

a stacked manner. Deep CNNs are used to explore spatial features among road segments in the whole traffic network, and then those spatial features are fed to LSTMs for temporal feature extraction before being fed to a fully-connected layer for final prediction.

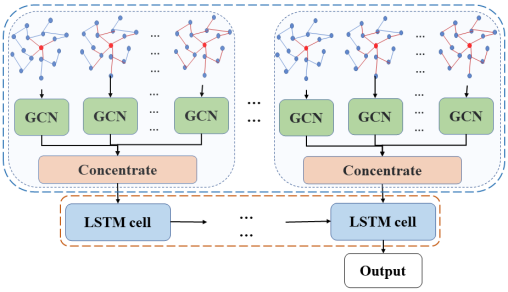
In the following, we review two categories of models adopting such stacked architecture. These models are GCN-based and designed (1) without and (2) with Seq2Seq framework.

5.2.1. Stacked Architecture without the Seq2Seq Framework

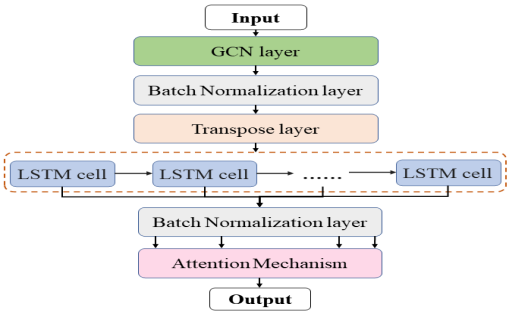
In Fig. 3, we illustrates three GCN-based models adopting the stacked architectures. These models did not exploit the Seq2Seq framework. Fig. 3 (a) shows the architecture of T-GCN [36] for network-wide traffic speed prediction. T-GCN consists of GCN and GRU, in which GCN is used to model the topological structure of large-scale road networks for capturing spatial dependencies and GRU is then utilized to model the changes of traffic data along the time dimension for capturing temporal dependencies. Similar to T-GCN, TGC-LSTM [35], shown in Fig. 3 (b) consists of GCN (instead of GRU in T-GCN) and LSTM. Compared to T-GCN that analyzes spatial features from the $k-hop$ neighborhood, TGC-LSTM separately extracts spatial features from $k-hop$ neighborhoods and then concatenates them as the final spatial features. The main idea is that capturing spatial features from different neighborhoods could provide more information for helping the model to achieve better predictions. Fig. 3 (c) presents the architecture of GLA [63] for traffic flow prediction. The input is first sent to GCN for capturing spatial relationships of traffic flow from observation locations, and then its output is fed to LSTM for the temporal dependency analysis. Compared to T-GCN in Fig. 3 (a) and TGC-LSTM in Fig. 3 (b), each GCN and LSTM in GLA is followed by a batch normalization layer to avoid over-fitting. The attention mechanism following LSTM just after a batch normalization layer is responsible of distributing different weights to hidden states so as to account for their different contributions to the targeted output. Similar to T-GCN, GLA only extracts the spatial dependencies on the $k-hop$ ($k=1$) neighborhood by the GCN module. Please refer to Section 7 for our comparative evaluations of these differences in their architecture constructions affect their performance under the same conditions and datasets.



(a) T-GCN



(b) TGC-LSTM



(c) GLA

Figure 3: GCN-based models with stacked architectures: (a) T-GCN, (b) TGC-LSTM – models spatial features from $1-hop$ and $k-hop$ neighborhoods; (c) GLA – model dynamic-temporal dependency with attention mechanism.

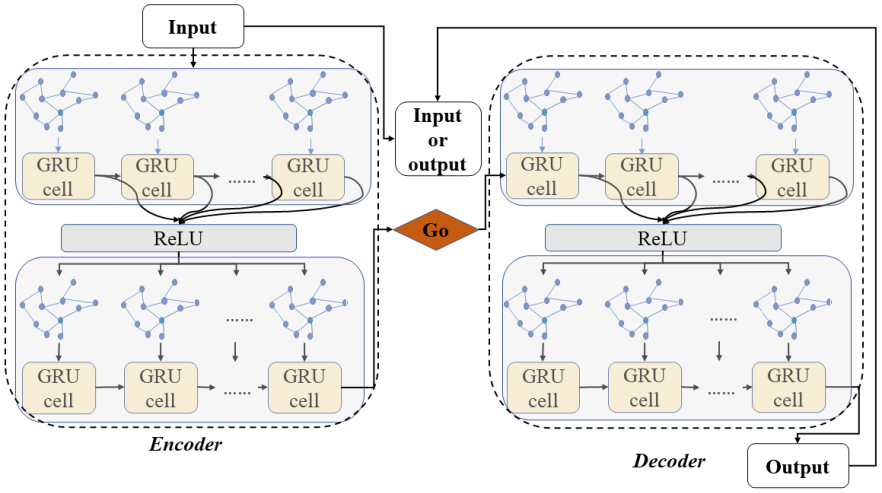
5.2.2. Stacked Architecture Under the Seq2Seq Framework

Fig. 4 presents three models built under the Seq2Seq frameworks, namely (a) DCRNN [33], (b) AGC-Seq2Seq-Att [34], and (c) STANN [62]. They implement traffic prediction based on different types of features. DCRNN, AGC-Seq2Seq-Att and STANN respectively extract fixed spatial-temporal features, spatial and dynamic-temporal features, and dynamic-spatial and dynamic-temporal features for final predictions.

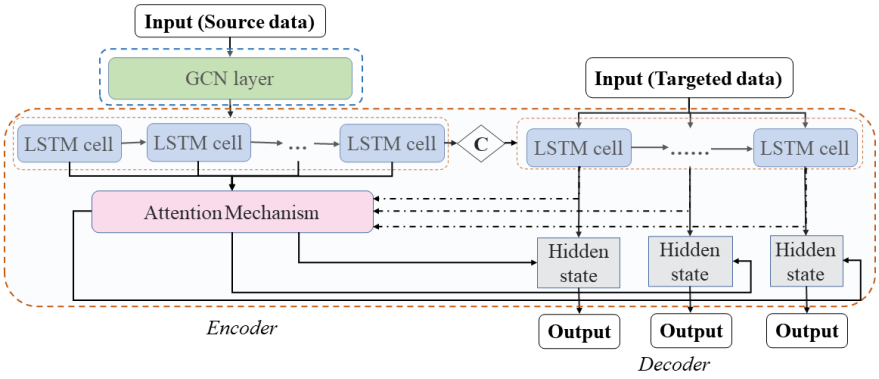
Under the Seq2Seq framework, the architecture of DCRNN (cf. Fig. 4 (a)) includes four diffusion convolutional GRUs: two as the encoder and the other two as the decoder. Firstly, the diffusion convolution operates on a weighted matrix for extracting spatial features. Then the extracted spatial features are sent to GRU for the temporal feature extraction in both encoder and decoder. The weighted matrix that contains the local traffic data information of each node in its $k - hop$ neighborhood is generated by the thresholded Gaussian kernel on the real distances between neighboring nodes. The core part of DCRNN is that the matrix multiplication in the conventional GRU is replaced by the diffusion convolution operation (named as diffusion convolutional GRU).

The architecture of AGC-Seq2Seq-Att in Fig. 4 (b) includes two parts: GCN layers for capturing the spatial characteristics based on the topology of the road network and the Seq2Seq framework for extracting the temporal features by encoding the spatially-fused time series from GCN and then decoding it to the targeted multi-step outputs. In addition, it also adopts the attention mechanism to allocate different weights to the historical sequences in the decoder of the Seq2Seq framework. This method makes historical traffic data of each time interval with different probabilities to contribute to the targeted outputs. AGC-Seq2Seq-Att employs GRU as the inner structure for both the encoder and decoder under the Seq2Seq framework. Compared to DRCNN in Fig. 4 (a) that uses **Diffusion GCN** (cf. Section 4.1.2), AGC-Seq2Seq-Att adopts **Traffic GCN** (cf. Section 4.1.2) and also introduces the attention mechanism into the decoder for capturing dynamic temporal dependencies.

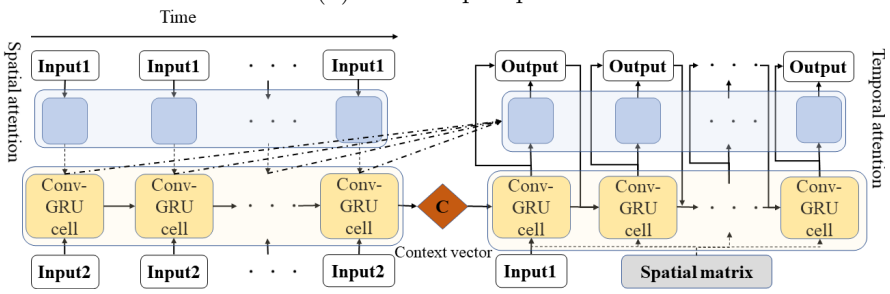
The architecture of STANN (Fig. 4(c)) consists of two convolutional gated recurrent unit components and the attention mechanism. STANN employs the attention mechanism on the spatial matrix, that concatenates $k - hop$ neighborhood matrices, to build the spatial attention matrix. This spatial attention matrix is sent to the encoder consisting of a convolutional GRU un-



(a) DCRNN



(b) AGC-Seq2Seq-Att



(c) STANN

Figure 4: Three stacked architectures from GCN-based models under the Seq2Seq framework include (a) DCRNN – modeling fixed spatial-temporal features, (b) AGC-Seq2Seq-Att – modeling spatial and dynamic-temporal features and (c) STANN – modeling dynamic-spatial and dynamic-temporal features.

der the Seq2Seq framework for spatial-temporal feature extraction. The last hidden state in the encoder is considered as the context vector to connect the decoder that decodes the context vector to the output via another convolutional GRU and attention mechanism. While AGC-Seq2Seq-Att (Fig. 4(b)) adds the attention mechanism to model dynamic-temporal dependency in the decoder, STANN instead applies the attention mechanism for dynamic-spatial and dynamic-temporal features towards final prediction.

5.3. Hybrid Architectures

The third architectural approach is based on a mixed of both paralleled and stacked approach. Fig. 5 and Fig. 6 show two models adopting this hybrid architecture; namely SAGCN-SST [69] and GMAN [70], respectively. The architecture of SAGCN-SST (shown in Fig. 5) consists of three blocks: input block, spatial block and temporal block. From the figure, we can see that both the spatial and temporal blocks have separate sub-blocks adopting the paralleled approach while the these spatial and temporal blocks are serially connected (i.e., adopting the stacked approach). Specifically, the input block is responsible for preparing the raw traffic and graph data into a trainable format as input to the spatial block. The spatial block, which contains k paralleled sub-spatial blocks, each corresponding to a $k-hop$ neighborhoods, extracts both fixed and dynamic spatial characteristics through GCN blocks under the sub-spatial block. Each sub-spatial block consists of GCN layers with the attention mechanism (named Self-AGCN layers) and a feed forward neural network (named FFNN) layer. This attention mechanism is able to learn the degree of influence of different individual neighbor to the targeted node. The k spatially-fused time series from sub-spatial blocks are concatenated as input to the temporal block. The temporal block is built based on the Seq2Seq framework with an encoder and a decoder for extracting the long-temporal dependency of traffic data. GRU is considered as an inner structure for both the encoder and decoder. Compared to AGC-Seq2Seq-Att (cf. Fig. 4 (b)) that applies the attention mechanism into the decoder for capturing dynamic temporal dependencies, SAGCN-SST adopts the attention mechanism into the GCN layers for extracting dynamic spatial features. In addition, AGC-Seq2Seq-Att is only able to capture fixed-spatial features from the $k-hop$ neighborhood, but SAGCN-SST is not only capable to extract fixed-spatial features but also dynamic-spatial features from $k-hop$ neighborhoods through k paralleled sub-spatial blocks.

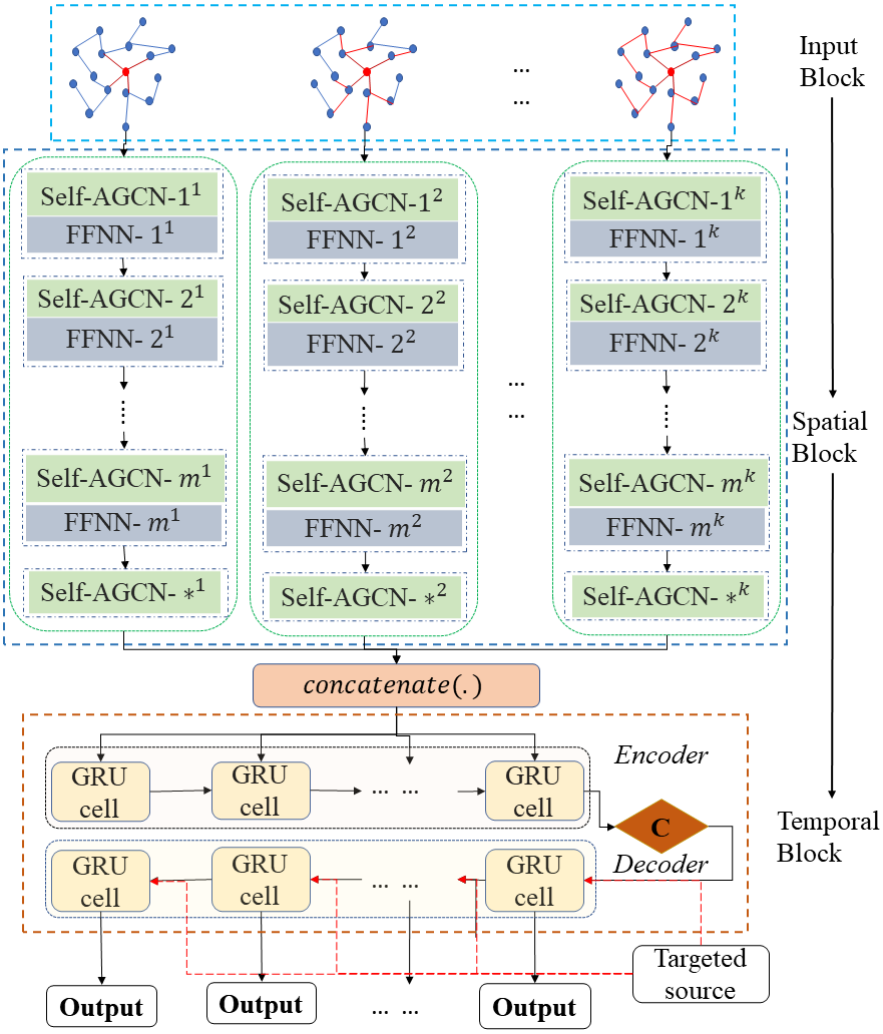


Figure 5: The architecture of SAGCN-SST that extracts dynamic-spatial and temporal features. It consists of Input Block, Spatial Block with an paralleled inner structure and Temporal Block with a stacked inner structure.

The architecture of GMAN in Fig. 6 is also built based on the Seq2Seq framework, and both the encoder and the decoder consist of L ST-Attention Blocks adopting the stacked approach. Each ST-Attention Block contains paralleled spatial and temporal attention mechanisms with gated fusions. The spatial attention mechanism is used to adaptively capture the correlations between sensors in the road network by dynamically assigning different

weights to different sensors at different time intervals and then summing them for the targeted sensor. If each sensor is targeted with different weights by all sensors in the network, the time and memory consumption are very high. To reduce the costs of time and computer memory, GMAN randomly partition N sensors to G groups, and each group has $M = N/G$ sensors and the learnable parameters inside of each group are shared with others. The temporal attention mechanism is utilized to adaptively model the correlations among different time intervals by assigning different weights to previous observations and then summing them for the future time intervals. Both spatial and temporal features are fused by a gated fusion based on gated mechanism in GRU [100]. A transform attention layer is used to connect the encoder and the decoder by converting the encoded features to the decoder. Besides, GMAN also includes a Spatial-Temporal Embedding (STE) module consisting of a spatial embedding part and a temporal embedding part, which are used to encode vertices into vectors as spatial embedding by the *node2vec* approach [91] and encode each timestamp into a vector as temporal embedding by the one hot encoder. The spatial embedding features describe correlations among traffic sensors on the road network and the temporal embedding features contain timestamps of the day-of-week and time-of-day. Furthermore, the temporal embedding features include both historical and future timestamps.

6. Public Data Sources

To help researchers further develop more valuable works, we summarize below commonly used public data sources for traffic prediction from our review of the literature.

- **Caltrans Performance Measurement System (PeMS):** The traffic data from PeMS is collected in real-time from over 39,000 detectors across major metropolitan areas in the State of California. It includes traffic speed, flow, occupancy, vehicle miles traveled (VMT), vehicle hours traveled (VHT) and Q (VMT/VHT). The time step in PeMS is 5 minutes [117]. The traffic datasets provided by this system and used in the recent literature include `PEMS03` [118], `PEMS04` [118], `PEMS07` [118], `PEMS08` [118], `PEMS-SF` [119] and `Pems-Bay` [33].
- **UK traffic data:** This data source provides average journey time, traffic speed and flow information for 15-minute periods since April

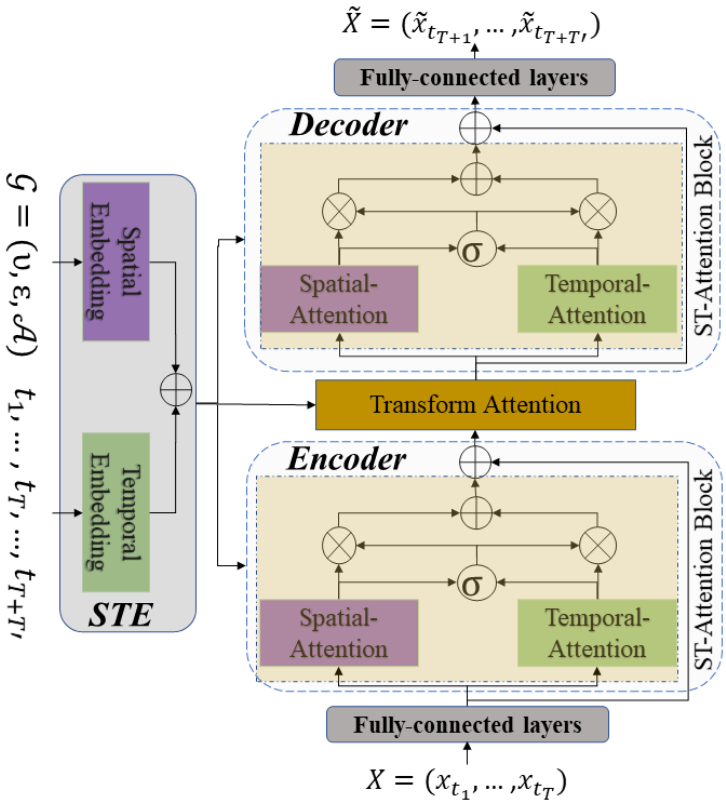


Figure 6: The architecture of GMAN that consists of an encoder and a decoder. Both encoder and decoder have the same structure and consists of a spatial-attention block and a temporal-attention block in a paralleled way.

2015 until now on all motorways. **"A"** roads are managed by Highways England, known as the Strategic Road Network in England [120].

- **Beijing traffic data:** This is a large-scale real-world traffic speed prediction dataset (named Q-traffic dataset) collected through Baidu map. It consists of three sub-datasets: 1) query sub-dataset recording the starting timestamp, coordinates of the starting location, coordinates of the destination and estimated travel time; 2) traffic speed sub-dataset recording traffic speed from 15,073 road segments covering approximately 738.91 km; 3) road network sub-dataset recording the topology of the road network [121].

- **NYC traffic data:** This dataset collected from taxi trips in New York City includes pick-up and drop-off dates/times, pick-up and drop-off locations, trip distances, itemized fares, rate types, payment types and driver-reported passenger counts from 2009 to now [122].

- **Los Angeles traffic data:** This dataset is collected from 207 loop detectors in Los Angeles, named Metr-La, and consists of traffic speed data covering 4 months [33].

- **San Francisco taxi traffic data:** This dataset contains mobility traces of taxi cabs in San Francisco and is collected from about 500 taxis over 30 days [123]. It is similar to NYC traffic data.

- **Traffic data from the Illinois Department of Transportation:** This system provides traffic flow, speed and travel time [124]. These traffic data can be used for the tasks of traffic flow, speed and travel time prediction, similar to the functions of UK traffic data.

7. Comparative Experiments

Based on our taxonomy, we conduct a comparative study to evaluate the performance of the recent hybrid deep learning models fairly under same conditions. We select ten representative models covering the different branches of our taxonomy including CNN-, GCN- and transformer-based models for our experiments as well as the different architectural approaches including paralleled, stacked and hybrid. We re-implement these selected models and test them on two large public datasets. The following subsections will describe two datasets, evaluation metrics, and results and discussion.

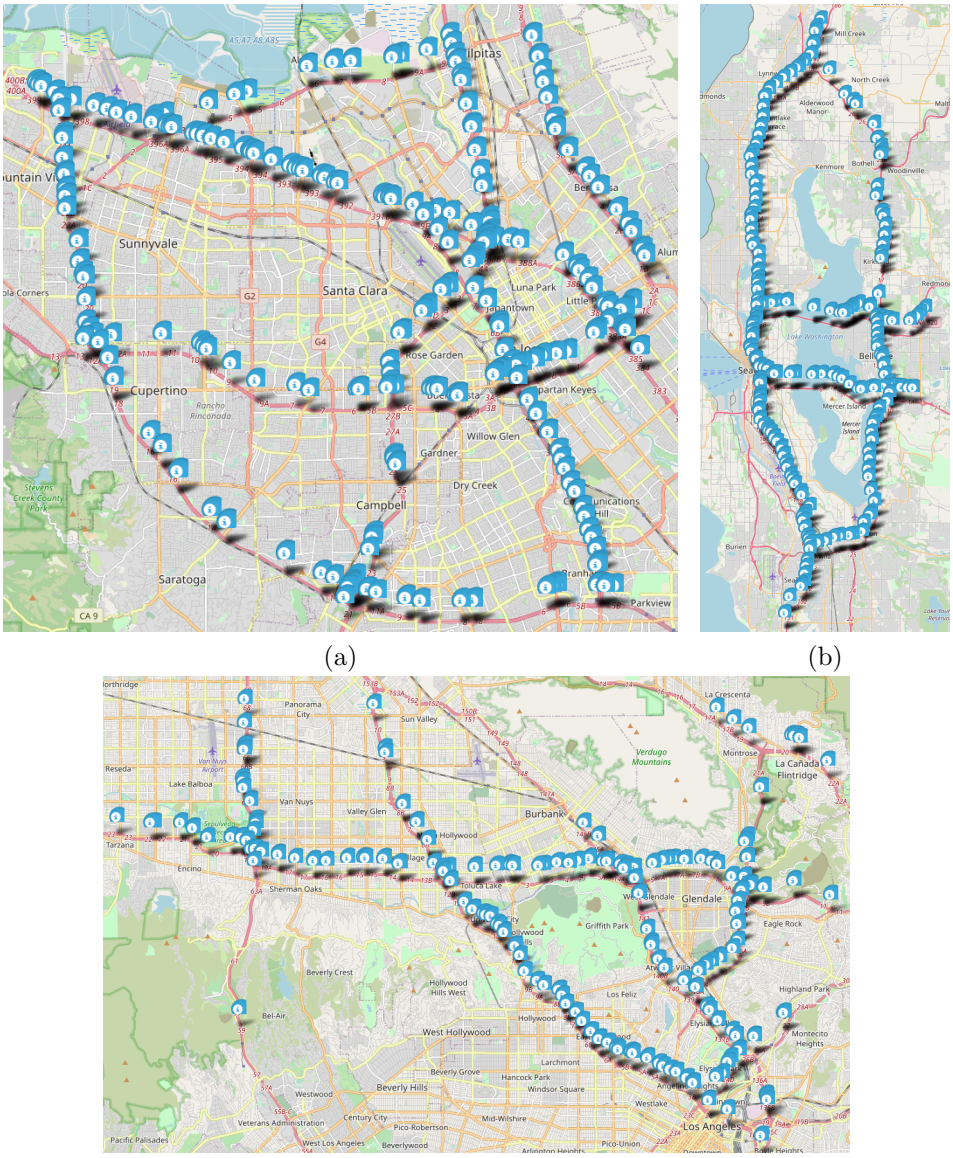
7.1. Data Description

For our comparison study, three real-world datasets, named `Pems-Bay`, `Loop-Seattle` and `Metr-La` that were made available by Li *et al.* [33], Cui *et al.* [35][125], and Li *et al.* [33] respectively, are used to evaluate the performances of the selected models described in Section 5. Fig. 7 displays the locations of loop detectors for (a) `Pems-Bay`, (b) `Loop-Seattle`, and (c) `Metr-La`. `Pems-Bay` is collected from California Transportation Agencies (CalTrans) Performance Measurement System (PeMS) [126] and contains the traffic speed information from 325 detectors and the physical connections of these detectors in Bay Area. Specifically, this dataset covers traffic information for six months ranging from 1^{st} Jan, 2017 to 30^{th} Jun, 2017. A time step is 5 minutes and the number of observed traffic data points are 16,937,700 (= 52116 $\times$ 325). `Loop-Seattle` is collected from 323 detectors in the Greater Seattle area and covers the entire year of 2015. A time step is also 5 minutes and the number of traffic data points are 33953760 (= 105120 $\times$ 323). `Metr-La` is collected from the highways of Los Angeles County during the 1^{st} of March and the 30^{th} of June in 2012 and consists of 207 sensors. Same as the other two, a time step is 5 minutes and the number of traffic data points is 7,094,304 (= 34,272 $\times$ 207). The difference to the other two is that `Metr-La` missed 575,302 data points, accounting for 8.11% of all data points.

TABLE 3 summarizes the information regarding all three datasets including maximum (Max), minimum (Min), mean value (Mean), standard deviation (Std), variance (Var) and size (MB). Comparatively, `Metr-La` is more complex with more fluctuations as its standard deviation and variance are larger than the other two. In addition, the size of `Loop-Seattle` data is twice of `Pems-Bay` and five times of `Metr-La`.

Table 3: Characteristics of both datasets

Dataset	Max	Min	Mean	Std	Var	Size
`Pems-Bay`	85.10	0.00	62.62	8.56	85.41	135.90 MB
`Loop-Seattle`	158.19	0.74	56.57	11.43	147.25	274.40 MB
`Metr-La`	70.00	0.00	53.72	19.19	374.85	57.00MB



(a) (b)

(c)

Figure 7: Locations of loop detectors in (a) `Pems-Bay`, (b) `Loop-Seattle` and (c) `Metr-La` datasets.

7.2. Evaluation Metrics

To compare the models, two performance metrics commonly used in the literature [127][128], namely Mean Absolute Error (MAE) Eq. (28) and Root-Mean Square Error (RMSE) Eq. (29), are used in our evaluation.

$$MAE = \frac{1}{N \times T'} \cdot \sum_{i=1}^{N} \sum_{t=1}^{T'} |x_t^i - \tilde{x}_t^i| \tag{28}$$

$$RMSE = \left[\frac{1}{N \times T'} \cdot \sum_{i=1}^{N} \sum_{t=1}^{T'} (x_t^i - \tilde{x}_t^i)^2 \right]^{\frac{1}{2}} \tag{29}$$

MAE presents the average absolute difference between the real and predicted traffic data. It is used to measure absolute prediction error. RMSE is the standard deviation of the residuals, which is the difference between the real and predicted traffic data.

Furthermore, following [127][129], we also define the accuracy of traffic prediction as $(100 - MAPE)\%$ where Mean Absolute Percentage Error (MAPE) is given by Eq. (30).

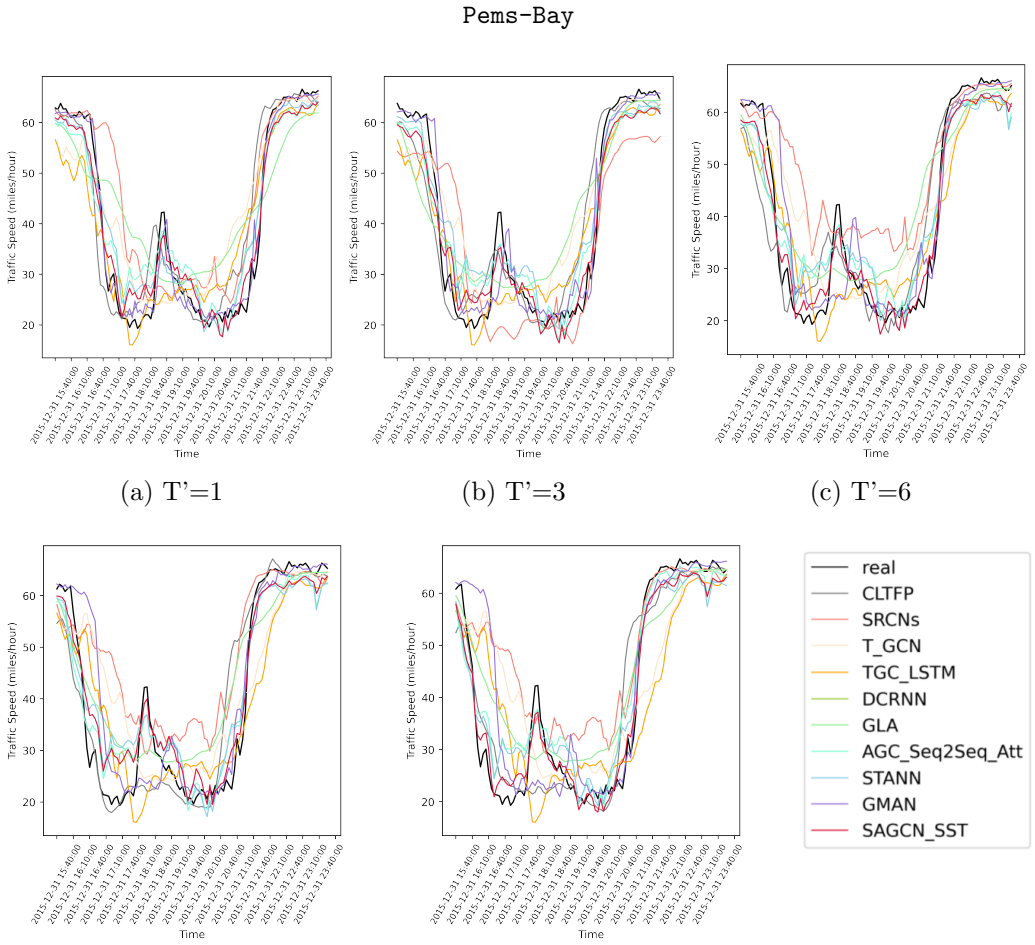
$$MAPE = \frac{1}{N \times T'} \cdot \sum_{i=1}^{N} \sum_{t=1}^{T'} \frac{|x_t^i - \tilde{x}_t^i|}{x_t^i} \times 100\%. \tag{30}$$

where MAPE is the percentage of MAE and is utilized to measure the prediction error.

7.3. Comparison against Ground Truth Data

This section compares the predicted traffic against real data and analyzes the CDF of MAEs from all representative models. Fig 8 presents the real traffic condition based on the data (i.e., the black line) and the predicted traffic condition (other colored lines) by the chosen models from 2015-12-31 15:10:00 to 2015-12-31 23:40:00 under busy traffic condition for `Pems-Bay`. Fig. 8 (a), (b), (c), (d) and (e) are respectively corresponding to prediction horizon, $T' = \{1, 3, 6, 9, 12\}$ related to $\{5, 15, 30, 45, 60\}$ minutes. The X-axis and the Y-axis represent the time and traffic speed (miles/hour) respectively. Traffic speed is lower between 16:10:00 and 20:40:00 due to congestion. From the figure, we observe that all models can broadly follow the general patterns of the real traffic. However, the abilities of models following the changes of real traffic data decrease for long-term prediction (i.e., when prediction horizon is increased).

Fig. 9 presents the CDF of MAEs for $T' = \{1, 6, 12\}$ from all models on `Pems-Bay` (Fig. 9 (a), (b) and (c)), on `Loop-Seattle` (Fig. 9 (d), (e) and (f)) and on `Metr-La` (Fig. 9 (g), (h) and (i)). For `Pems-Bay`, the best models for



(a) T'=1 (b) T'=3 (c) T'=6

(d) T'=9 (e) T'=12 (f)

Figure 8: The real and predicted traffic from all models with busy traffic situation in `Pems-Bay`.

$T' = \{1, 6, 12\}$ are GMAN, SAGCN-SST and CLTFP, and SAGCN-SST respectively. However, from TABLE 4 that presents experimental results of all compared models for different horizons, the best model for $T' = 1$ is DCRNN. Note that in TABLE 4, MAE is calculated by averaging individual MAEs while Fig. 9 uses individual MAE directly. GMAN is better than DCRNN in terms of MAEs (cf. Fig.9) but worse than DCRNN for the average of MAEs (cf. TABLE 4). This implies that the ability of GMAN to predict abnormal values, like noisy data or data with sudden changes, is worse than DCRNN

so as to result in the large average MAEs. In addition, for $T' = 1$, the CDF curves for all models are closer because short-term prediction is less complex compared to long-term prediction. For $T' = \{6, 12\}$, the differences of performances for all models are more obvious, especially for $T' = 12$. We can observe three distinct groups: 1) SAGCN-SST, STANN and AGC-Seq2Seq-Att; 2) DCRNN, GMAN and CLTFP; and 3) T-GCN, TGC-LSTM, SRCNs and GLA. SAGCN-SST, STANN and AGC-Seq2Seq-Att that are built based on GCN, Seq2Seq framework and attention mechanism, achieve the best results among the three groups due to their ability to capture dynamic-spatial, dynamic-temporal and long-temporal features. For `Loop-Seattle`, most of results are similar to `Pems-Bay`, but the obvious difference is that SAGCN-SST is the best model for $T' = 1$ whereas GMAN is the best performer for `Pems-Bay`. This is due to `Loop-Seattle` being larger and more complicated than `Pems-Bay` (cf. TABLE 3), and these requires more complex models to offer better results. For `Metr-La`, the trend of results are similar to the results achieved on `Loop-Seattle` while actual results are generally worse than the results achieved on the other two datasets due to having higher number of missing data points. Furthermore, GLA is considered as the worst across three different prediction horizons.

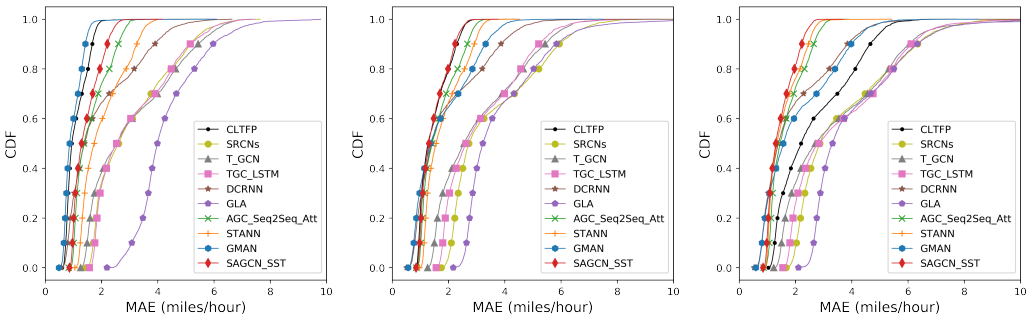
From the above, we summarize the observations as follow:

- The performances of all representative models for traffic prediction decrease as the prediction horizon increases.
- The differences of performances for all compared models are more obvious for larger prediction horizons.
- Models built based on GCN, Seq2Seq framework and attention mechanism could perform better for long-term traffic prediction.

7.4. Short- and Long-term Traffic Prediction

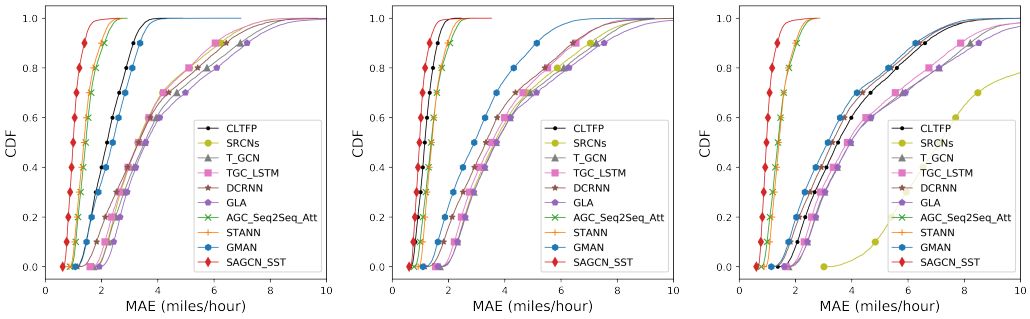
This section compares the performance for all representative models on short- and long-term traffic prediction on both datasets. We show the prediction accuracy (100%-MAPE) of all models for short- and long-term traffic predictions for `Pems-Bay` (Fig. 10 (a)), `Loop-Seattle` (Fig. 10 (b)) and `Metr-La` (Fig. 10 (c)). Here, the prediction accuracy for short- and long-term are the average of (100%-MAPE) for $T' = \{1, 3\}$ and the average of (100%-MAPE) for $T' = \{6, 9, 12\}$ respectively. For `Pems-Bay`, all models perform better for short-term traffic prediction than for long-term, but the differences

Pems-Bay



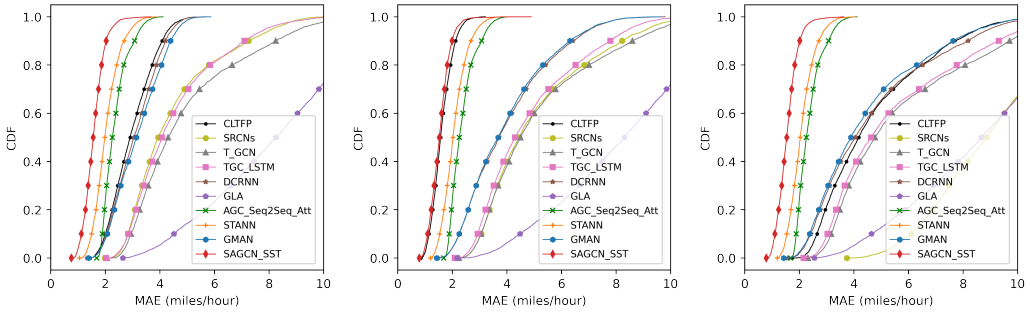
(a) T'=1 (b) T'=6 (c) T'=12

Loop-Seattle



(d) T'=1 (e) T'=6 (f) T'=12

Metr-La



(g) T'=1 (h) T'=6 (i) T'=12

Figure 9: The CDF curves of MAEs for all models on all three datasets for traffic prediction with different prediction horizons.

are small. Those small differences imply that the targeted traffic data with longer horizon still has close relationship with historical traffic data. For short-term prediction, CLTFP is the best model with 97% accuracy followed by DCRNN, AGC-Seq2Seq-Att, STANN, GMAN and SAGCN-SST which with over 95% accuracy. SRCNs is the worst performing model with 87% accuracy. For long-term prediction, SAGCN-SST is the best model (96% accuracy) and SRCNs is still the worst.

For `Loop-Seattle`, SAGCN-SST, AGC-Seq2Seq-Att and STANN achieve best results and keep similar accuracy between short- and long-term predictions. Similar to `Loop-Seattle`, for `Metr-La`, SAGCN-SST, AGC-Seq2Seq-Att and STANN are still the top three achieving best results. Other models obtain lower accuracy for long-term prediction than for short-term prediction. It indicates SAGCN-SST, AGC-Seq2Seq-Att and STANN built under the Seq2Seq architecture are more capable of dealing with long-term dependencies. In addition, among three datasets, the difference of accuracy for short- and long-term are larger on `Metr-La` and smaller on `Pems-Bay`, which suggests that `Pems-Bay` exhibits more obvious long-term temporal dependencies when compared to `Loop-Seattle` and `Metr-La`.

The findings by comparing short- and long-term traffic prediction could be summarized as follows:

- The differences of performances for all models for short-term traffic prediction are smaller than for long-term traffic prediction.
- Traffic has longer temporal dependencies for `Pems-Bay` when compared against `Loop-Seattle` and `Metr-La`.
- All representative models perform better on `Pems-Bay` than on `Loop-Seattle` and on `Metr-La`.

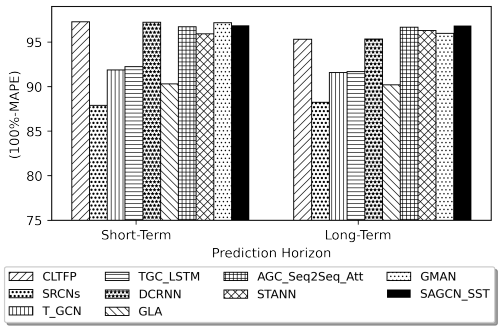
7.5. Model Architecture and Constituent Modules

This section discusses the results from the perspective of model's architecture design, in which all compared models are grouped as follows:

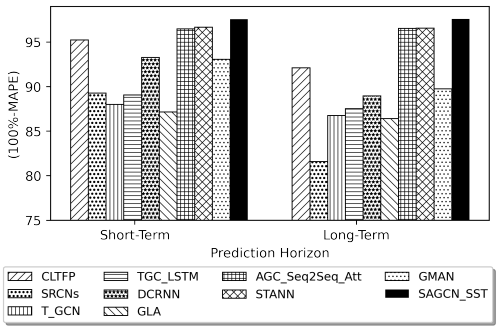
(a) CNN-based models including CLTFP and SRCNs.

(b) GCN-based models including T-GCN, TGC-LSTM, DCRNN, GLA, AGC-Seq2Seq-Att, STANN and SAGCN-SST.

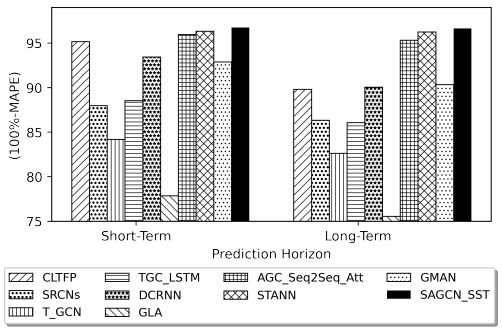
(c) Transformer-based model including GMAN.



(a) `Pems-Bay`



(b) `Loop-Seattle`



(c) `Metr-La`

Figure 10: The prediction accuracy (100%-MAPE) of short- and long-term traffic prediction from all models on three datasets.

TABLE 4 presents the detailed results of all models on all three datasets based on the three error metrics (i.e., MAE, MAPE and RMSE).

Table 4: Results from all models on both datasets

Model Name	Pems-Bay			Loop-Seattle			Metr-La		
	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE
(a) 5-min future prediction (T'=1)									
CLTFP	1.1069	**2.24**	1.7428	2.2525	5.52	3.1492	2.3416	5.57	3.2713
SRCNs	3.0012	7.88	4.9425	3.6932	10.39	5.5424	3.8341	11.02	6.1523
T-GCN	3.0045	8.19	5.1369	3.9918	11.87	6.0916	6.3234	15.67	9.2888
TGC-LSTM	3.0378	7.79	4.9819	3.6507	10.85	5.4992	4.6806	11.21	7.5362
DCRNN	**0.9232**	2.26	**1.6350**	2.3905	5.73	3.5506	2.3325	5.77	4.0161
GLA	4.3009	9.95	5.9600	4.2039	12.88	6.3319	11.8797	23.59	13.6322
AGC-Seq2Seq-Att	1.5656	3.29	2.2695	1.5146	3.54	2.0184	1.9618	3.95	2.9543
STANN	1.9643	4.30	2.9029	1.4434	3.34	1.9099	**1.0892**	3.66	2.8919
GMAN	0.9468	2.85	2.9861	2.4510	6.15	3.7890	2.5358	6.54	4.6822
*SAGCN-SST	1.4093	3.13	2.0184	**1.0468**	**2.46**	**1.3695**	1.5165	**3.23**	**2.1047**
(b) 15-min future prediction (T'=3)									
CLTFP	1.5640	3.19	2.3709	1.6257	4.00	2.2290	2.7933	4.12	4.1738
SRCNs	5.8743	16.34	9.3951	3.8732	11.07	5.9035	4.6913	13.02	8.8659
T-GCN	2.9609	8.05	5.0348	4.0576	12.12	6.1988	6.4863	15.94	9.7750
TGC-LSTM	3.0190	7.73	4.9323	3.7321	11.02	5.6377	4.8676	11.71	8.2545
DCRNN	**1.3489**	3.29	2.7286	2.8787	7.70	4.7072	2.7776	7.38	5.3535
GLA	3.7579	9.41	5.7377	4.1712	12.81	6.4248	11.2571	22.70	13.2856

AGC-Seq2Seq-Att	1.5562	3.29	2.2689	1.4629	3.49	1.9738	2.0400	4.13	3.1442
STANN	1.7678	3.85	2.5972	1.4327	3.32	1.8908	1.8351	3.70	2.8343
GMAN	1.3494	**2.81**	2.9138	2.8577	7.72	4.8653	2.8493	7.68	5.6404
*SAGCN-SST	1.4364	3.14	**2.0448**	**1.0258**	**2.39**	**1.3358**	**1.5216**	**3.26**	**2.1322**
(c) 30-min future prediction (T'=6)									
CLTFP	**1.1921**	**2.83**	**1.5834**	1.4690	2.91	2.1000	3.1638	6.81	6.9471
SRCNs	3.4198	9.05	5.5118	4.0975	12.05	6.3305	4.3469	12.61	6.9237
T-GCN	2.9909	8.09	5.0879	4.1756	12.49	6.4046	6.8020	16.56	10.5514
TGC-LSTM	3.0845	7.87	5.0698	3.8981	11.53	6.0007	5.3022	12.73	9.3802
DCRNN	1.6670	4.20	3.6067	3.3235	9.72	5.7323	3.1839	8.95	6.4789
GLA	3.7895	9.60	5.8467	4.3120	13.20	6.6893	11.6708	23.63	14.1950
AGC-Seq2Seq-Att	1.5840	3.36	2.2989	1.4577	3.45	1.9635	2.2958	4.71	3.4912
STANN	1.7589	3.83	2.5658	1.4740	3.41	1.9420	1.8659	3.74	2.8085
GMAN	1.6402	3.63	3.7687	3.2183	9.25	5.7687	3.1499	8.91	6.5055
*SAGCN-SST	1.4224	3.14	2.0381	**1.0188**	**2.38**	**1.3316**	**1.5405**	**3.34**	**2.1539**
(d) 45-min future prediction (T'=9)									
CLTFP	2.4238	5.22	4.0115	3.4425	9.70	5.1973	6.1009	10.49	10.9485
SRCNs	5.8642	16.34	9.3799	4.3141	12.82	6.6889	6.2910	10.76	11.21
T-GCN	3.1179	8.37	5.3479	4.3779	13.14	6.7734	7.1467	17.34	11.2733
TGC-LSTM	3.2477	8.25	5.4156	4.1632	12.40	6.5318	5.7707	13.90	10.3859
DCRNN	1.8438	4.72	4.0444	3.6521	11.14	6.3940	3.4493	10.02	7.1828

GLA	3.8586	9.77	5.9953	4.5164	13.78	7.0570	12.0324	24.43	14.9598
AGC-Seq2Seq-Att	1.5147	3.21	2.1976	1.4898	3.53	2.0057	2.4807	5.10	3.7515
STANN	1.7967	4.03	2.6325	1.4835	3.44	1.9612	1.8728	3.78	2.8972
GMAN	1.7917	4.08	4.1664	3.4745	10.32	6.3433	3.3546	9.75	7.0569
*SAGCN-SST	**1.4269**	**3.14**	**2.0503**	**1.0224**	**2.40**	**1.3378**	**1.5610**	**3.32**	**2.1661**
(e) 60-min future prediction (T'=12)									
CLTFP	2.6475	5.98	4.5727	3.9472	11.82	6.1794	7.6731	13.25	13.2743
SRCNs	3.5731	9.86	5.9204	8.4099	30.35	11.7965	9.0269	17.65	14.38
T-GCN	3.3143	8.82	5.7469	4.6508	14.08	7.2687	7.4969	18.24	11.9422
TGC-LSTM	3.4793	8.80	5.8997	4.4880	13.53	7.1516	6.2545	15.16	11.3245
DCRNN	1.9685	5.06	4.3070	3.9242	12.26	6.8891	3.6755	10.91	7.7280
GLA	3.9793	10.05	6.2410	4.7786	13.78	7.0570	12.3956	25.26	15.6969
AGC-Seq2Seq-Att	1.6141	3.41	2.3051	1.4551	3.45	1.9575	2.1084	4.28	3.2115
STANN	1.5066	3.25	2.1560	1.4857	3.44	1.9550	1.8802	3.79	2.8996
GMAN	1.8892	4.36	4.3754	3.6786	11.16	6.7498	3.5033	10.35	7.4248
*SAGCN-SST	**1.4229**	**3.14**	**2.0291**	**1.0292**	**2.42**	**1.3455**	**1.5636**	**3.39**	**2.1585**

7.5.1. CNN-based models

CLTFP and SRCNs are two CNN-based models. Both use CNN for spatial feature extraction and LSTM for temporal feature extraction. However, CLTFP adopts the paralleled architecture while SRCNs follows the stacked architecture.

From TABLE 4, we see that both models achieve better results on `Pems-Bay`. CLTFP consistently outperforms SRCNs on all datasets across different prediction horizons. The MAPEs of CLTFP for $T' = \{1, 3, 6, 9, 12\}$ are {2.24, 3.91, 2.83, 5.22, 5.98} on `Pems-Bay`, {5.52, 4.00, 2.91, 9.70, 11.82} on `Loop-Seattle` and {5.57, 4.12, 6.81, 10.46, 13.25} on `Metr-La`. The main reason is that CLTFP has the ability to capture more features including short-term variation, long-term periodicity and spatial relation of traffic data by three inputs corresponding to a CNN and two LSTMs. In comparison, SRCNs only has an input used to capture spatial features first and then its output is treated as the input of LSTM for temporal feature extraction. In addition, CLTFP, built with the paralleled architecture, analyzes different patterns of traffic data separately. This could reduce the temporal information distortion as opposed to SRCNs, adopting the stacked architecture, the temporal features are extracted after the spatial feature extraction.

7.5.2. GCN-based models

T-GCN, TGC-LSTM, DCRNN, GLA, AGC-Seq2Seq-Att, STANN and SAGCN-SST are seven GCN-based models. From TABLE 4, GLA, T-GCN and TGC-LSTM generally achieve larger errors (MAPE) compared to the other four models on all datasets for all different prediction horizons. Both GLA and TGC-LSTM are built using GCN and LSTM but GLA has an extra attention layer. On the other hand, T-GCN is built based on GCN and GRU. Between T-GCN and TGC-LSTM, the main difference is that TGC-LSTM captures spatial features from $k - hop$ neighborhoods instead of only using a single specific $k - hop$ neighborhood in T-GCN. Since traffic data from more neighborhoods can offer more features, TGC-LSTM achieves better results than T-GCN. In addition, GLA only uses the $1 - hop$ neighborhood for spatial feature extraction. This limits the consideration of the prediction to adjacent stations only. Compared to T-GCN ($k = 3$), the region limit to capture spatial features is smaller for GLA ($k = 1$). This is likely the reason why the predictions achieved by GLA have largest errors.

The four models achieving better performance, namely DCRNN, AGC-Seq2Seq-Att, STANN and SAGCN-SST, all share a common design decision -

i.e., they are built with Seq2Seq framework. This seems to be the main reason for them to achieve more accurate predictions compared to the previous three models. Among these four models, DCRNN's performance worsens when the prediction horizon is increased while the other three models maintain stable level of prediction accuracy over different prediction horizon. This observation is valid for all three datasets.

On `Pems-Bay`, DCRNN obtains the best results for $T' = 1$ and the worst for $T' = \{6, 9, 12\}$. This indicates that DCRNN is more suited for short-term prediction. We also observe that the results for AGC-Seq2Seq-Att, STANN and SAGCN-SST are similar on both short- and long-term predictions. These models share a common feature, i.e., they all use the attention mechanism in some manner. AGC-Seq2Seq-Att included the attention mechanism in its decoder for dynamic-temporal feature analysis while STANN included the attention mechanism in both its encoder and decoder for dynamic-spatial and dynamic-temporal feature extraction. SAGCN-SST integrated the attention mechanism in its GCN layers for dynamic-spatial feature analysis. Furthermore, SGCN-SST has the ability to deal with different $k-hop$ neighborhoods as opposed to only using one specific $k-hop$ neighborhood as in AGC-Seq2Seq-Att. The above suggests that attention mechanism allows better predictions at longer prediction horizon. SGCN-SST achieves the most accurate predictions when T' is larger. We attribute this to its ability to consider multiple neighborhoods.

The larger `Loop-Seattle` dataset recorded more complex traffic patterns with higher volatility. For this dataset, DCRNN achieves the worst accuracy across all prediction horizons, suggesting that DCRNN did not manage to cope with the complexity of the traffic in `Loop-Seattle` even at short prediction horizon (i.e., $T' = 1$) when it performs the best in `Pems-Bay` dataset. For more challenged `Metr-La` with many missing data points, DCRNN is also worst when compared to AGC-Seq2Seq-Att, STANN and SAGCN-SST. At the other end of the spectrum, we see that SAGCN-SST consistently performs the best across different prediction horizons. This indicates that SAGCN-SST is able to handle data of different complexities and thus, offer robust prediction performance. Moreover, we also observe that both SAGCN-SST and STANN achieve better results on `Loop-Seattle` than on `Pems-Bay` while the opposite is true for DCRNN and AGC-Seq2Seq-Att. All these four models perform worse on `Metr-La` than the other two. We conjecture that this may be due to (1) DCRNN concentrates on spatial-temporal features and neglects the dynamic-spatial and/or dynamic-temporal features analysis

and (2) that dynamic-spatial feature analysis seems more important than dynamic-temporal analysis for traffic prediction on road network with more complex traffic patterns.

7.5.3. Transformer-based model

GMAN is a transformer-based model. Compared to GCN-based models that use GCN for the spatial or dynamic-spatial feature extraction, GMAN makes use of the attention mechanism to directly analyze dynamic-spatial features. For dynamic-spatial features, individual weights are assign to the different sensors on the road network for attributing the different contribution of each location to the targeted sensor. This incur higher costs in terms of computer memory and computation time. To reduce these costs, GMAN randomly partitions all sensors into several equal-size groups and use them for analyzing local spatial features instead of global spatial features. This is conceptually similar to the GCN-based models that define the size of neighborhood using k for local spatial feature analysis. However, since group membership is random, there is no specific focus on nodes' neighborhoods.

This could be the reason that GMAN perform worse than AGC-Sqe2Sqe-Att, STANN and SAGCN-SST, especially for long-term prediction. In addition, similar to DCRNN, the three errors achieved by GMAN (cf. TABLE 4) increase with longer prediction horizon. For instance, its MAPE increases from 2.85% for $T' = 1$ to 4.36% for $T' = 12$ on `Pems-Bay`. On `Loop-Seattle` and `Metr-La`, the MAPEs improve in the same trend but in larger magnitude.

7.6. Prediction Robustness

In this section, we focus on the robustness analysis of all representative models. According to [130], the robustness of deep learning model is evaluated by measuring how effective the model is when tested with new independent but similar datasets. Therefore, we analyze the robustness of all competing models by comparing their performances on different datasets. TABLE 5 presents the difference of MAPE between (1) `Pems-Bay` (top row) and `Loop-Seattle` (i.e., $|MAPE_{\texttt{Pems-Bay}} - MAPE_{\texttt{Loop-Seattle}}|$) and (2) `Metr-La` (bottom row) and `Loop-Seattle` for all prediction horizons. AGC-Seq2Seq-Att obtains the averagely smallest difference of MAPEs between `Pems-Bay` (`Metr-La`) and `Loop-Seattle` and the average of differences for all prediction horizons is only 0.561% (= (0.180% + 0.942%) / 2). It is followed by STANN (0.441% = (0.538% + 0.344%)) and SAGCN-SST (0.813% = (0.728% + 0.898 / 2)). This implies AGC-Seq2Seq-Att is the most robust among all models,

followed by STANN and SAGCN-SST. Although the differences of AGC-Seq2Seq-Att and STANN are smaller than SAGCN-SST, the three types of error achieved by SAGCN-SST are smaller than the other two. Considering the three types of errors, prediction accuracy for short- and long-term predictions and the differences of MAPEs between three datasets, our results suggest SAGCN-SST offers a balance trade-off among the models used in our comparative experiments. Finally, models that are developed based on GCN, attention mechanism and the Seq2Seq architecture (AGC-Seq2Seq-Att and SAGCN-SST) achieves more accurate predictions and more robust on different datasets.

Table 5: The differences of MAPE between `Pems-Bay`(top) / `Metr-La`(bottom) and `Loop-Seattle` for different prediction horizons.

Model Name	Data	CLTFP	SRCNs	T-GCN	TGC-LSTM	DCRNN	GLA	AGC-Seq2Seq-Att	STANN	GMAN	SAGCN-SST
5-min	Pems-Bay	3.28	2.51	3.68	3.06	3.47	2.93	**0.25**	0.96	3.30	0.67
	Metr-La	0.05	0.63	3.80	0.36	**0.04**	10.71	0.41	0.32	0.39	0.77
15-min	Pems-Bay	0.81	5.27	4.07	3.29	4.41	3.40	**0.20**	0.53	4.91	0.75
	Metr-La	0.12	1.95	3.82	0.69	0.32	7.89	0.64	0.38	**0.04**	0.87
30-min	Pems-Bay	**0.08**	3.00	4.40	3.66	5.52	3.60	0.09	0.42	5.62	0.76
	Metr-La	4.62	0.56	4.07	1.20	0.77	10.43	1.26	**0.33**	0.34	0.96
45-min	Pems-Bay	4.48	3.52	4.77	4.15	6.42	4.01	**0.32**	0.59	6.24	0.74
	Metr-La	0.79	2.06	4.20	1.50	1.12	10.65	1.57	**0.34**	0.57	0.92
60-min	Pems-Bay	5.84	20.49	5.26	4.73	7.20	3.73	**0.04**	0.19	6.80	0.72
	Metr-La	1.43	12.70	4.16	1.63	1.35	11.48	0.83	**0.35**	0.81	0.97
average	Pems-Bay	2.898	6.958	4.436	3.778	5.404	3.534	**0.180**	0.538	5.374	0.728
	Metr-La	1.402	2.324	4.010	1.076	0.704	10.232	0.942	0.344	**0.274**	0.898
variance	Pems-Bay	4.721	46.647	0.299	0.362	1.800	0.130	0.011	0.063	1.472	**0.001**
	Metr-La	3.550	35.760	0.035	0.290	0.323	1.871	0.220	**0.001**	0.218	0.006

8. Summary, Conclusions and Future Directions

In this work, we first reviewed recent traffic prediction solutions employing hybrid deep learning models. In our review, we first put into context the evolution of the traffic prediction solutions leading to these models before constructing a taxonomy of these hybrid models, broadly into three: 1) CNN-based models; 2) GCN-based models; and 3) transformer-based models; based on the methodologies used for the spatial feature extraction. The models in each of these category are further segregated and analyzed based on their spatial-temporal feature extraction methods (i.e., either fixed or dynamic as defined in Definition 1 and 2). We then highlight and discuss common architectural choices of the reviewed models as well as the constituent sub-models frequently exploited in these hybrid models. In addition, we also summarize commonly used public data sources to facilitate future researches in developing and evaluating new solutions.

We then chose ten representative models covering all types of models based on our taxonomy and conduct an extensive comparative study. Since in the literature, different datasets and settings are used in evaluation, it is then difficult to compare the performance of different models fairly. In our study, we reconstructed the chosen models and compare their performance under identical experiment setup and using the same datasets. We consider both short- and long-term predictions to offer better overview of the strength and capabilities of the different models. From our study, models that can achieve high prediction accuracy for traffic predictions on large-scale road networks have characteristics summarized as follows:

- Using GCN module to analyze spatial or dynamic-spatial feature dependencies.
- Building under the Seq2Seq framework that is effective for long-term temporal feature extraction.
- Using the attention mechanism for dynamic-spatial or/and temporal feature analysis. It is noted that applying the attention mechanism for dynamic-spatial feature analysis can obtain marginally better results compared to applying it for dynamic-temporal feature analysis.

We summarize the observations from our study on model architectures and constituent modules as follow:

- Between the paralleled and stacked architecture, the former achieves better prediction when same constituent modules are used to build the model.

- Considering larger neighborhood and/or more neighborhoods offers better performance when the models are built based on same modules and have the similar architecture.

- Models that include the Seq2Seq architecture achieve more accurate predictions, especially for long-term traffic prediction.

- Models that are built based on both attention mechanism and the Seq2Seq architecture obtain small differences of performances between short- and long-term traffic prediction.

- Models that are able to extract dynamic spatial features offer better performances than models that concentrate on dynamic temporal feature analysis.

Furthermore, we identify two directions that could potentially offer significant improvement in the prediction accuracy.

- For spatial feature analysis, the attention mechanism allocates different weights to sensors in the neighborhood of the targeted sensor for contributing to the targeted sensor. However, existing works mostly consider fixed neighborhood region (i.e., all the neighborhoods are of the same size). In real world, the neighborhood size having influence on the targeted sensor is dynamic along both the space and time domains. For example, at the same time step, the neighborhood size that impacts traffic state in the targeted sensor during period of serious traffic congestion is larger than under normal traffic situation. In addition, at different time steps, the neighborhood size for the same targeted sensor should be treated differently because traffic state constantly evolve over time. Therefore, how to define a dynamic neighborhood for each sensor over time is one challenging aspect for building a deep learning model to analyze real-world spatial dependencies.

- For the temporal feature analysis, future traffic state relies more on previous time steps under the abnormal traffic situation such as serious traffic congestion than when under normal traffic situation. Therefore,

the length of historical traffic data used to predict future traffic data should be dynamic rather than fixed for each sensor along the time domain.

References

[1] UN, 68% of the world population projected to live in urban areas by 2050, says united nations (2018).
URL https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html

[2] INRIX, Inrix 2020 global traffic scorecard (2020).
URL https://inrix.com/scorecard/

[3] T. Vaa, M. Penttinen, I. Spyropoulou, Intelligent transport systems and effects on road traffic accidents: state of the art, IET Intelligent Transport Systems 1 (2) (2007) 81–88.

[4] M. S. Ahmed, A. R. Cook, Analysis of freeway traffic time-series data by using Box-Jenkins techniques, no. 722, 1979.

[5] Z. B. Xian, L. Qiang, Arma-based traffic prediction and overload detection of network [j], Journal of Computer Research and Development 12 (2002).

[6] B. M. Williams, L. A. Hoel, Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results, Journal of transportation engineering 129 (6) (2003) 664–672.

[7] S. M. Weiss, I. Kapouleas, An empirical comparison of pattern recognition, neural nets, and machine learning classification methods., in: IJCAI, Vol. 89, 1989, pp. 781–787.

[8] C. M. Bishop, Pattern recognition and machine learning, springer, 2006.

[9] J. M. Peña, P. A. Gutiérrez, C. Hervás-Martínez, J. Six, R. E. Plant, F. López-Granados, Object-based image classification of summer crops with machine learning methods, Remote Sensing 6 (6) (2014) 5019–5041.

[10] S. Loussaief, A. Abdelkrim, Machine learning framework for image classification, in: 2016 7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), IEEE, 2016, pp. 58–61.

[11] D. M. Powers, C. C. Turk, Machine learning of natural language, Springer Science & Business Media, 2012.

[12] S. Sagiroglu, U. Yavanoglu, E. N. Guven, Web based machine learning for language identification and translation, in: Sixth International Conference on Machine Learning and Applications (ICMLA 2007), IEEE, 2007, pp. 280–285.

[13] M. Castro-Neto, Y.-S. Jeong, M.-K. Jeong, L. D. Han, Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions, Expert systems with applications 36 (3) (2009) 6164–6173.

[14] L. Zhang, Q. Liu, W. Yang, N. Wei, D. Dong, An improved k-nearest neighbor model for short-term traffic flow prediction, Procedia-Social and Behavioral Sciences 96 (2013) 653–662.

[15] A. Csikós, Z. J. Viharos, K. B. Kis, T. Tettamanti, I. Varga, Traffic speed prediction method for urban networks—an ann approach, in: 2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), IEEE, 2015, pp. 102–108.

[16] S. Sun, C. Zhang, G. Yu, A bayesian network approach to traffic flow forecasting, IEEE Transactions on intelligent transportation systems 7 (1) (2006) 124–132.

[17] H.-S. Zhang, Y. Zhang, Z.-H. Li, D.-C. Hu, Spatial-temporal traffic data analysis based on global data management using mas, IEEE Transactions on Intelligent Transportation Systems 5 (4) (2004) 267–275.

[18] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, B. Yin, A comprehensive survey on traffic prediction, arXiv preprint arXiv:2004.08555 (2020).

[19] R. Fu, Z. Zhang, L. Li, Using lstm and gru neural network methods for traffic flow prediction, in: 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), IEEE, 2016, pp. 324–328.

[20] R. Toncharoen, M. Piantanakulchai, Traffic state prediction using convolutional neural network, in: 2018 15th International Joint Conference on Computer Science and Software Engineering (JCSSE), IEEE, 2018, pp. 1–6.

[21] F. Diehl, T. Brunner, M. T. Le, A. Knoll, Graph neural networks for modelling traffic participant interaction, in: 2019 IEEE Intelligent Vehicles Symposium (IV), IEEE, 2019, pp. 695–701.

[22] J. Barros, M. Araujo, R. J. Rossetti, Short-term real-time traffic prediction methods: A survey, in: 2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), IEEE, 2015, pp. 132–139.

[23] L. N. Do, N. Taherifar, H. L. Vu, Survey of neural network-based models for short-term traffic state prediction, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 9 (1) (2019) e1285.

[24] K. Lee, M. Eo, E. Jung, Y. Yoon, W. Rhee, Short-term traffic prediction with deep neural networks: A survey, IEEE Access 9 (2021) 54739–54756.

[25] K. Irawan, R. Yusuf, A. S. Prihatmanto, A survey on traffic flow prediction methods, in: 2020 6th International Conference on Interactive Digital Media (ICIDM), IEEE, 2020, pp. 1–4.

[26] B. Alsolami, R. Mehmood, A. Albeshri, Hybrid statistical and machine learning methods for road traffic prediction: a review and tutorial, Smart Infrastructure and Applications (2020) 115–133.

[27] A. Boukerche, Y. Tao, P. Sun, Artificial intelligence-based vehicular traffic flow prediction methods for supporting intelligent transportation systems, Computer Networks 182 (2020) 107484.

[28] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. Choudhury, A. Qin, A survey on modern deep neural network for traffic prediction: Trends, methods and challenges, IEEE Transactions on Knowledge and Data Engineering (2020).

[29] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, B. Yin, Deep learning on traffic prediction: Methods, analysis and future directions, IEEE Transactions on Intelligent Transportation Systems (2021).

[30] A. M. Nagy, V. Simon, Survey on traffic prediction in smart cities, Pervasive and Mobile Computing 50 (2018) 148–163.

[31] W. Jiang, J. Luo, Graph neural network for traffic forecasting: A survey, arXiv preprint arXiv:2101.11174 (2021).

[32] R. Jiang, D. Yin, Z. Wang, Y. Wang, J. Deng, H. Liu, Z. Cai, J. Deng, X. Song, R. Shibasaki, Dl-traff: Survey and benchmark of deep learning models for urban traffic prediction, in: Proceedings of the 30th ACM international conference on information & knowledge management, 2021, pp. 4515–4525.

[33] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, arXiv preprint arXiv:1707.01926 (2017).

[34] Z. Zhang, M. Li, X. Lin, Y. Wang, F. He, Multistep speed prediction on traffic networks: A deep learning approach considering spatio-temporal dependencies, Transportation research part C: emerging technologies 105 (2019) 297–322.

[35] Z. Cui, K. Henrickson, R. Ke, 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 (2019).

[36] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-gcn: A temporal graph convolutional network for traffic prediction, IEEE Transactions on Intelligent Transportation Systems (2019).

[37] P. J. Bickel, C. Chen, J. Kwon, J. Rice, E. Van Zwet, P. Varaiya, Measuring traffic, Statistical Science (2007) 581–597.

[38] W. Min, L. Wynter, Real-time road traffic prediction with spatio-temporal correlations, Transportation Research Part C: Emerging Technologies 19 (4) (2011) 606–616.

[39] Y. Shi, H. Feng, X. Geng, X. Tang, Y. Wang, A survey of hybrid deep learning methods for traffic flow prediction, in: Proceedings of the 2019 3rd International Conference on Advances in Image Processing, 2019, pp. 133–138.

[40] T. Seo, A. M. Bayen, T. Kusakabe, Y. Asakura, Traffic state estimation on highway: A comprehensive survey, Annual reviews in control 43 (2017) 128–151.

[41] L. Zhu, F. R. Yu, Y. Wang, B. Ning, T. Tang, Big data analytics in intelligent transportation systems: A survey, IEEE Transactions on Intelligent Transportation Systems 20 (1) (2018) 383–398.

[42] Y. Wu, H. Tan, Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework, arXiv preprint arXiv:1612.01022 (2016).

[43] Y. Liu, H. Zheng, X. Feng, Z. Chen, Short-term traffic flow prediction with conv-lstm, in: 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), IEEE, 2017, pp. 1–6.

[44] H. Yu, Z. Wu, S. Wang, Y. Wang, X. Ma, Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks, Sensors 17 (7) (2017) 1501.

[45] G. Yang, Y. Wang, H. Yu, Y. Ren, J. Xie, Short-term traffic state prediction based on the spatiotemporal features of critical road sections, Sensors 18 (7) (2018) 2287.

[46] Y. Wu, H. Tan, L. Qin, B. Ran, Z. Jiang, A hybrid deep learning based traffic flow prediction method and its understanding, Transportation Research Part C: Emerging Technologies 90 (2018) 166–180.

[47] Z. Duan, Y. Yang, K. Zhang, Y. Ni, S. Bajgain, Improved deep hybrid networks for urban traffic flow prediction using trajectory data, IEEE Access 6 (2018) 31820–31827.

[48] W. Jin, Y. Lin, Z. Wu, H. Wan, Spatio-temporal recurrent convolutional networks for citywide short-term crowd flows prediction, in: Proceedings of the 2nd International Conference on Compute and Data Analysis, 2018, pp. 28–35.

[49] D. Zang, J. Ling, Z. Wei, K. Tang, J. Cheng, Long-term traffic speed prediction based on multiscale spatio-temporal feature learning network, IEEE Transactions on Intelligent Transportation Systems 20 (10) (2018) 3700–3709.

[50] S. Zhao, S. Lin, J. Xu, Time series traffic prediction via hybrid neural networks, in: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE, 2019, pp. 1671–1676.

[51] J. Zhao, H. Qu, J. Zhao, D. Jiang, Spatiotemporal traffic matrix prediction: A deep learning approach with wavelet multiscale analysis, Transactions on Emerging Telecommunications Technologies 30 (12) (2019) e3640.

[52] Z. Zheng, Y. Yang, J. Liu, H.-N. Dai, Y. Zhang, Deep and embedded learning approach for traffic flow prediction in urban informatics, IEEE Transactions on Intelligent Transportation Systems 20 (10) (2019) 3927–3939.

[53] A. E. Essien, I. Petrounias, P. Sampaio, S. Sampaio, Deep-presimm: Integrating deep learning with microsimulation for traffic prediction, in: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), IEEE, 2019, pp. 4257–4262.

[54] Z. He, C.-Y. Chow, J.-D. Zhang, Stcnn: A spatio-temporal convolutional neural network for long-term traffic prediction, in: 2019 20th IEEE International Conference on Mobile Data Management (MDM), IEEE, 2019, pp. 226–233.

[55] H. Yao, X. Tang, H. Wei, G. Zheng, Z. Li, Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 5668–5675.

[56] P. Le Nguyen, Y. Ji, et al., Deep convolutional lstm network-based traffic matrix prediction with partial information, in: 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), IEEE, 2019, pp. 261–269.

[57] K. Niu, H. Zhang, T. Zhou, C. Cheng, C. Wang, A novel spatio-temporal model for city-scale traffic speed prediction, IEEE Access 7 (2019) 30050–30057.

[58] X. Ma, H. Zhong, Y. Li, J. Ma, Z. Cui, Y. Wang, Forecasting transportation network speed using deep capsule networks with nested lstm models, IEEE Transactions on Intelligent Transportation Systems (2020).

[59] B. Liao, J. Zhang, C. Wu, D. McIlwraith, T. Chen, S. Yang, Y. Guo, F. Wu, Deep sequence learning with auxiliary information for traffic prediction, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 537–546.

[60] Z. Xie, W. Lv, S. Huang, Z. Lu, B. Du, R. Huang, Sequential graph neural network for urban road traffic speed prediction, IEEE Access (2019).

[61] Y. Kim, P. Wang, L. Mihaylova, Scalable learning with a structural recurrent neural network for short-term traffic prediction, IEEE Sensors Journal 19 (23) (2019) 11359–11366.

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

[63] Z. Li, G. Xiong, Y. Chen, Y. Lv, B. Hu, F. Zhu, F.-Y. Wang, A hybrid deep learning approach with gcn and lstm for traffic flow prediction, in: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE, 2019, pp. 1929–1933.

[64] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, J. Zhang, Urban traffic prediction from spatio-temporal data using deep meta learning, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 1720–1730.

[65] C. Chen, K. Li, S. G. Teo, X. Zou, K. Wang, J. Wang, Z. Zeng, Gated residual recurrent graph neural networks for traffic prediction, in: Pro-

ceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 485–492.

[66] N. Zhang, X. Guan, J. Cao, X. Wang, H. Wu, Wavelet-hst: A wavelet-based higher-order spatio-temporal framework for urban traffic speed prediction, IEEE Access 7 (2019) 118446–118458.

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

[68] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, B. Yin, Multi-stage attention spatial-temporal graph networks for traffic prediction, Neurocomputing 428 (2021) 42–53.

[69] G. Zheng, W. K. Chai, V. Katos, A dynamic spatial-temporal deep learning framework for traffic speed prediction on large-scale road networks, Expert Systems with Applications (2022).

[70] C. Zheng, X. Fan, C. Wang, J. Qi, Gman: A graph multi-attention network for traffic prediction, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 1234–1241.

[71] M. Xu, W. Dai, C. Liu, X. Gao, W. Lin, G.-J. Qi, H. Xiong, Spatial-temporal transformer networks for traffic flow forecasting, arXiv preprint arXiv:2001.02908 (2020).

[72] Y. Kim, Convolutional neural networks for sentence classification, arXiv preprint arXiv:1408.5882 (2014).

[73] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[74] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei-Fei, Large-scale video classification with convolutional neural networks, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2014, pp. 1725–1732.

[75] S. Ji, W. Xu, M. Yang, K. Yu, 3d convolutional neural networks for human action recognition, IEEE transactions on pattern analysis and machine intelligence 35 (1) (2012) 221–231.

[76] S. Lawrence, C. L. Giles, A. C. Tsoi, A. D. Back, Face recognition: A convolutional neural-network approach, IEEE transactions on neural networks 8 (1) (1997) 98–113.

[77] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, Y. Wang, Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction, Sensors 17 (4) (2017) 818.

[78] Z. Ge, C. Wei Koong, K. Vasilis, W. Michael, A joint temporal-spatial ensemble model for short-term traffic prediction, Neurocomputing 457 (2021) 26–39.

[79] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.

[80] B. Vijayalakshmi, K. Ramar, N. Jhanjhi, S. Verma, M. Kaliappan, K. Vijayalakshmi, S. Vimal, U. Ghosh, An attention-based deep learning model for traffic flow prediction using spatiotemporal features towards sustainable smart city, International Journal of Communication Systems 34 (3) (2021) e4609.

[81] H. Zheng, F. Lin, X. Feng, Y. Chen, A hybrid deep learning model with attention-based conv-lstm networks for short-term traffic flow prediction, IEEE Transactions on Intelligent Transportation Systems (2020).

[82] R. de Medrano, J. L. Aznarte, A spatio-temporal attention-based spot-forecasting framework for urban traffic prediction, Applied Soft Computing 96 (2020) 106615.

[83] E. Ahmed, A. Saint, A. E. R. Shabayek, K. Cherenkova, R. Das, G. Gusev, D. Aouada, B. Ottersten, A survey on deep learning advances on different 3d data representations, arXiv preprint arXiv:1808.01462 (2018).

[84] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: Advances in neural information processing systems, 2014, pp. 3104–3112.

[85] F. Li, J. Feng, H. Yan, G. Jin, D. Jin, Y. Li, Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution, arXiv preprint arXiv:2104.14917 (2021).

[86] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, in: The 28th International Joint Conference on Artificial Intelligence (IJCAI), International Joint Conferences on Artificial Intelligence Organization, 2019.

[87] F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv preprint arXiv:1511.07122 (2015).

[88] K. He, X. Chen, Q. Wu, S. Yu, Z. Zhou, Graph attention spatial-temporal network with collaborative global-local learning for citywide mobile traffic prediction, IEEE Transactions on Mobile Computing (2020).

[89] E. J. Keogh, M. J. Pazzani, Derivative dynamic time warping, in: Proceedings of the 2001 SIAM international conference on data mining, SIAM, 2001, pp. 1–11.

[90] X. Shi, H. Qi, Y. Shen, G. Wu, B. Yin, A spatial-temporal attention approach for traffic prediction, IEEE Transactions on Intelligent Transportation Systems (2020).

[91] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.

[92] Y. LeCun, Y. Bengio, et al., Convolutional networks for images, speech, and time series, The handbook of brain theory and neural networks 3361 (10) (1995) 1995.

[93] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907 (2016).

[94] J. Bastings, I. Titov, W. Aziz, D. Marcheggiani, K. Sima'an, Graph convolutional encoders for syntax-aware neural machine translation, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 1957–1967.

[95] L. Zhao, X. Peng, Y. Tian, M. Kapadia, D. N. Metaxas, Semantic graph convolutional networks for 3d human pose regression, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3425–3435.

[96] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 3634–3640.

[97] J. B. Estrach, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and deep locally connected networks on graphs, in: 2nd International Conference on Learning Representations, ICLR 2014, 2014.

[98] Y. Li, Y. Yuan, Convergence analysis of two-layer neural networks with relu activation, in: Advances in neural information processing systems, 2017, pp. 597–607.

[99] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780. doi:http://www.bioinf.jku.at/publications/older/2604.pdf.

[100] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, in: NIPS 2014 Workshop on Deep Learning, December 2014, 2014.

[101] N. Kalchbrenner, P. Blunsom, Recurrent continuous translation models, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013, pp. 1700–1709.

[102] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473 (2014).

[103] C. Yan, Y. Tu, X. Wang, Y. Zhang, X. Hao, Y. Zhang, Q. Dai, Stat: spatial-temporal attention mechanism for video captioning, IEEE transactions on multimedia 22 (1) (2019) 229–241.

[104] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, X. Tang, Residual attention network for image classification, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 3156–3164.

[105] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, Y. Bengio, Attention-based models for speech recognition, in: Advances in neural information processing systems, 2015, pp. 577–585.

[106] S. Guo, Y. Lin, N. Feng, C. Song, H. Wan, Attention based spatial-temporal graph convolutional networks for traffic flow forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 922–929.

[107] M. Li, Z. Zhu, Spatial-temporal fusion graph neural networks for traffic flow forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 4189–4196.

[108] K. Guo, Y. Hu, Z. Qian, H. Liu, K. Zhang, Y. Sun, J. Gao, B. Yin, Optimized graph convolution recurrent neural network for traffic prediction, IEEE Transactions on Intelligent Transportation Systems 22 (2) (2020) 1138–1149.

[109] Z. Pan, W. Zhang, Y. Liang, W. Zhang, Y. Yu, J. Zhang, Y. Zheng, Spatio-temporal meta learning for urban traffic prediction, IEEE Transactions on Knowledge and Data Engineering (2020).

[110] M. Lu, K. Zhang, H. Liu, N. Xiong, Graph hierarchical convolutional recurrent neural network (ghcrnn) for vehicle condition prediction, arXiv preprint arXiv:1903.06261 (2019).

[111] T. Mallick, P. Balaprakash, E. Rask, J. Macfarlane, Graph-partitioning-based diffusion convolutional recurrent neural network for large-scale traffic forecasting, Transportation Research Record 2674 (9) (2020) 473–488.

[112] Y. Zhang, T. Cheng, Y. Ren, K. Xie, A novel residual graph convolution deep learning model for short-term network-based traffic forecasting, International Journal of Geographical Information Science 34 (5) (2020) 969–995.

[113] Z. U. Abideen, H. Sun, Z. Yang, R. Z. Ahmad, A. Iftekhar, A. Ali, Deep wide spatial-temporal based transformer networks modeling for the next destination according to the taxi driver behavior prediction, Applied Sciences 11 (1) (2021) 17.

[114] H. Yan, X. Ma, Learning dynamic and hierarchical traffic spatiotemporal features with transformer, arXiv preprint arXiv:2104.05163 (2021).

[115] J. Zhao, X. Li, Q. Xue, W. Zhang, Spatial-channel transformer network for trajectory prediction on the traffic scenes, arXiv preprint arXiv:2101.11472 (2021).

[116] C. Yang, Spatio-temporal transformer with tcn for pedestrian trajectory prediction.

[117] C. D. of Transportation, Caltrans performance measurement system (pems).
URL https://pems.dot.ca.gov/

[118] C. Song, Y. Lin, S. Guo, H. Wan, 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, 2020, pp. 914–921.

[119] M. Cuturi, Fast global alignment kernels, in: Proceedings of the 28th international conference on machine learning (ICML-11), 2011, pp. 929–936.

[120] WebTRIS, Highways england.
URL https://webtris.highwaysengland.co.uk/

[121] B. Liao, J. Zhang, C. Wu, D. McIlwraith, T. Chen, S. Yang, Y. Guo, F. Wu, Deep sequence learning with auxiliary information for traffic prediction, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2018.

[122] N. Government, Tlc trip record data.
URL https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[123] P. Michal, S.-D. Natasa, G. Matthias, The epfl/mobility dataset (v. 2009-02-24).
URL https://crawdad.org/~crawdad/epfl/mobility/20090224/

[124] T. M. webmaster, Gateway traveler information system.
URL https://www.travelmidwest.com/lmiga/home.jsp

[125] Z. Cui, R. Ke, Z. Pu, Y. Wang, Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction, arXiv preprint arXiv:1801.02143 (2018).

[126] C. Chen, Freeway performance measurement system (pems), Public Roads 57 (3) (1994) 8–14.

[127] Y. Lv, et al., Traffic flow prediction with big data: A deep learning approach., IEEE Trans. Intell. Transp. Syst. 16 (2) (2015) 865–873.

[128] M.-C. Tan, S. C. Wong, J.-M. Xu, Z.-R. Guan, P. Zhang, An aggregation approach to short-term traffic flow prediction, IEEE Transactions on Intelligent Transportation Systems 10 (1) (2009) 60–69.

[129] W. Huang, G. Song, H. Hong, K. Xie, Deep architecture for traffic flow prediction: deep belief networks with multitask learning, IEEE Transactions on Intelligent Transportation Systems 15 (5) (2014) 2191–2201.

[130] H. Xu, C. Caramanis, S. Mannor, Sparse algorithms are not stable: A no-free-lunch theorem, IEEE transactions on pattern analysis and machine intelligence 34 (1) (2011) 187–193.