Forecasting Transportation Network Speed Using Deep Capsule Networks With Nested LSTM Models

Xiaolei Ma, Houyue Zhong, Yi Li, Junyan Ma, Zhiyong Cui, and Yinhai Wang

***Abstract*— Accurate and reliable traffic forecasting for complicated transportation networks is of vital importance to modern transportation management. The complicated spatial dependencies of roadway links and the dynamic temporal patterns of traffic states make it particularly challenging. To address these challenges, we propose a new capsule network (CapsNet) to extract the spatial features of traffic networks and utilize a nested LSTM (NLSTM) structure to capture the hierarchical temporal dependencies in traffic sequence data. A framework for network-level traffic forecasting is also proposed by sequentially connecting CapsNet and NLSTM. On the basis of literature review, our study is the first to adopt CapsNet and NLSTM in the field of traffic forecasting. An experiment on a Beijing transportation network with 278 links shows that the proposed framework with the capability of capturing complicated spatiotemporal traffic patterns outperforms multiple state-of-the-art traffic forecasting baseline models. The superiority and feasibility of CapsNet and NLSTM are also demonstrated, respectively, by visualizing and quantitatively evaluating the experimental results.**

***Index Terms*— Capsule network, LSTM, traffic prediction, spatial and temporal dependency, transportation network.**

I. Introduction

As a challenging component of intelligent transportation systems, traffic prediction has become crucial for individuals and public agencies due to the requirements of accurate travel time estimation and dynamic transportation management. Traffic prediction aims to forecast the future traffic states of connected roadway segments on the basis of historical traffic data within an underlying roadway network structure.

Manuscript received January 21, 2019; revised August 16, 2019, January 8, 2020, and March 10, 2020; accepted March 23, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1601600 and in part by the National Natural Science Foundation of China under Grant 61773036. The Associate Editor for this article was Y. Wang. *(Corresponding author: Zhiyong Cui.)*

Xiaolei Ma and Houyue Zhong are with the Beijing Key Laboratory for Cooperative Vehicle Infrastructure System and Safety Control, School of Transportation Science and Engineering, Beihang University, Beijing 100191, China, and also with the Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing 100191, China (e-mail: xiaolei@buaa.edu.cn; houyue.zhong@buaa.edu.cn).

Yi Li is with AutoNavi Software Company, Beijing 100102, China (e-mail: yier.ly@autonavi.com).

Junyan Ma is with the School of Information Engineering, Chang'an University, Xi'an 710064, China (e-mail: majy@chd.edu.cn).

Zhiyong Cui and Yinhai Wang are with the Department of Civil and Environmental Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: zhiyongc@uw.edu; yinhai@uw.edu).

Digital Object Identifier 10.1109/TITS.2020.2984813

In the early stage, the majority of traffic prediction studies that focus on small-scale roadway networks are normally fulfilled based on statistical models with limited transportation data. In recent years, advanced data-driven machine learning methods have been widely adopted for network-wide traffic state prediction with the rapid development in traffic sensing technologies and computational power. Machine learning models have outperformed classical statistical models due to their capabilities of handling high-dimensional and complicated spatiotemporal data. However, the potential of machine learning models for traffic prediction has not been fully utilized until the rise of deep neural network (NN) models (also referred to as deep learning models) [1].

Deep learning models have achieved superior performance in traffic forecasting tasks compared with conventional machine learning models. With the utilization of fully connected NNs [2] for traffic prediction, many advanced and powerful deep learning models, such as deep belief networks (DBNs) [3], convolutional NNs (CNNs) [4], and recurrent NNs (RNNs) [5], have been applied to extract high-dimensional features of traffic states and have achieved good prediction performance. However, these models should be improved in terms of capturing the spatial and temporal dependencies in high-dimensional traffic data. Most of the existing studies on traffic prediction have modeled spatial dependencies with CNNs [4], [6] and captured temporal dependencies via RNNs [1], [7]. They [8]–[10] have also proposed models by combining CNNs and RNNs to fulfill this task. However, conventional CNNs and RNNs have their limitations when handling network-wide traffic data.

Conventional CNNs are appropriate for capturing spatial relationships in Euclidean space that are represented by two-dimensional (2D) matrices or images. On this basis, spatiotemporal traffic data learning using CNNs can be roughly categorized into two strategies. The first strategy [4] uses CNNs to learn spatiotemporal traffic data as a 2D matrix, in which the spatial and temporal dimensions are separately distributed in two directions. However, the actual structure of a complicated roadway network cannot be properly represented by a 2D matrix, and CNNs inevitably capture a certain amount of spurious spatial relationships. The second strategy [6], [8] employs CNNs to capture the spatial dependencies by projecting various traffic states to their corresponding physical roadway links using different colors and by processing a traffic network map as an image. In this way, the actual

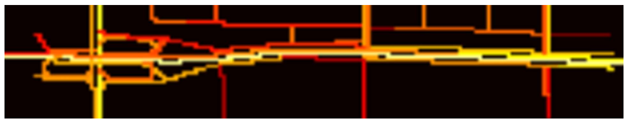


Fig. 1. Viaducts, intersections, and side roads in a traffic network.

spatial features of the traffic network are learned. However, CNN-based feature extraction models still face challenges. First, the pooling operations in CNNs proactively discards substantial information; thus, critical correlations of traffic states between links may be lost. Then, the neurons in conventional CNNs are unsuitable in representing the various properties of a particular entity [11], such as pose, deformation, albedo, hue, and texture. Given that the structure of a traffic network is fixed in a map-based image, the various colors of pixels located on roadways that represent traffic states can be considered as the textures or poses of the traffic network viewed from different perspectives. Thus, the CNN approach is insufficient to capture the relative spatial dependencies between colored pixels when these colors gradually change in a sequence of map-based images. In addition, the interdependencies between roadway links cannot be captured by CNNs for several specific complicated road network structures that contain viaducts, intersections, and side roads, as shown in **Fig. 1**.

RNN and its variants, such as long short-term memory (LSTM) network [12], are effective for capturing the temporal features of traffic states. Existing studies have used stacked LSTMs to enhance the short-term traffic prediction performance [7]. Generally, traffic conditions are not only influenced by short-term historical information that is directly relevant to current traffic states but also by upper-level long-term traffic patterns with strong periodicity and regularity. However, long-term temporal dependencies under severe weather or disasters do not contribute considerable short-term information to traffic prediction and should be selectively captured. According to [13], a stacked LSTM or a single-layer LSTM cannot comprehensively characterize a temporal hierarchy.

To overcome the drawbacks of conventional CNNs and LSTMs, we propose a new capsule network (CapsNet) [11] to extract the spatial features of traffic networks and utilize a newly proposed nested LSTM (NLSTM) [13] structure to improve the performance of time-series learning. The CapsNet utilizes capsules in vector form rather than in scalar form as neurons in the NN. The direction and length of a capsule vector encode the state of high-level features and the detection probability of a feature, respectively. With the aid of capsules and a dynamic routing algorithm between them, the CapsNet considers slightly active features and largely addresses the existing problems in conventional CNNs. The NLSTM with the capability to access inner memories selectively in constructing temporal hierarchies is utilized to capture the hierarchical temporal dependencies in traffic data dynamically. The evaluation results show that the proposed framework with the combination of CapsNet and NLSTM outperforms multiple state-of-the-art traffic forecasting baseline models.

In summary, our main contributions are as follows:

1. We learn the traffic network as an image and propose a new framework for network-level traffic prediction by *combining CapsNet and NLSTM. On the basis of* literature review, our study is the first to adopt CapsNet and NLSTM in traffic forecasting;
2. We propose a new CapsNet to extract the high-level features and capture the spatial dependencies between the roadway links from the traffic state images. The learnt features are characterized by the use of "capsules" in a vector form rather than traditional scalar forms of neurons;
3. We utilize an NLSTM structure to dynamically capture the hierarchical temporal dependencies in the spatial features learnt by CapsNet;
4. The superiority and feasibility of CapsNet and NLSTM for network-wide traffic forecasting are quantitatively evaluated by comparing with baseline models.

The remainder of this paper is organized as follows. Sections 2 and 3 presents the related works and our methodology, respectively. Section 4 shows the experimental data and results. Finally, Section 5 concludes the paper and open questions for future research.

II. Literature Review

The approaches in the existing literature on short-term traffic prediction can be divided into two families, namely, statistical methods and artificial intelligence. Statistical methods, such as autoregressive integrated moving average (ARIMA) [14], ARIMA variants [15]–[17], Kalman filter [18], and exponential smoothing [19], [20], have been investigated and applied to predict traffic flow parameters. In comparison with parametric statistical models, non-parametric machine learning models have more portability, higher accuracy, and are free of assumptions on data distribution [21]. For example, k-nearest neighbors [22] and support vector machines [23]–[25], [31], which are popular in the field of prediction, have been widely utilized to predict traffic speed and travel time.

However, as a component of artificial intelligence, machine learning methods may fail when addressing complicated high-dimensional data. In the early stage, several NN approaches, such as artificial NN [26] and radial basis function NN [27], [28], were applied to predict traffic states. In recent years, considerable advanced and powerful deep learning models, such as DBNs [3], stacked autoencoders [30], CNNs [4], and RNNs [1], [5], [7], [29], [34], have been adopted in traffic forecasting. As a representative variant of RNNs, LSTM was first introduced in traffic prediction task and showed promising performance [1]. On this basis, stacked bidirectional LSTMs are also adopted to enhance the short-term traffic prediction [7]. Existing studies [4], [8] have used CNNs in expanding the study areas to large-scale traffic networks, which are proven effective in computer vision and disease diagnosis areas [32], [33], to extract spatial dependencies from traffic data to facilitate prediction performance. For example, a one-dimensional CNN has been used to capture the spatial features of traffic flow [35].

However, a common means of adopting CNNs for traffic forecasting is by processing 2D spatial–temporal data as images and by learning spatial–temporal dependencies from

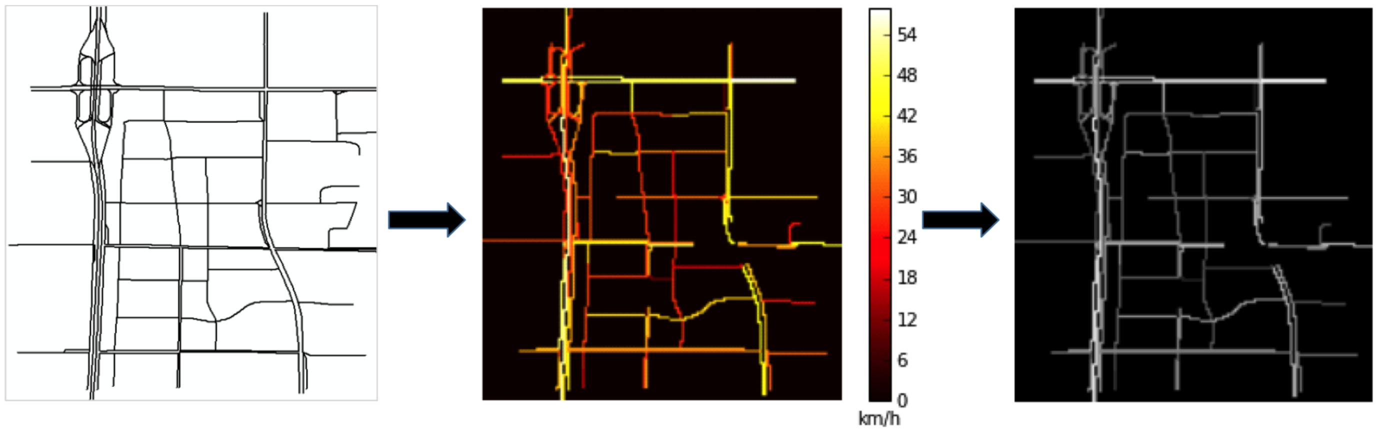


Fig. 2. Transportation network representation.

these images [4]. To model the spatial–temporal relationships of traffic states effectively, hybrid CNN approaches that incorporate LSTM [8] and residual unit [6] have been proposed for traffic prediction by learning the spatial features from the images converted from grid-or pixel-based traffic network maps. However, these approaches that adopt conventional CNNs still cannot handle the overlapping roadways caused by low-resolution images, separate spatially interlaced links in viaducts in 2D space, and accommodate the physical specialties of traffic networks.

We construct a new CapsNet to solve the limitations of CNN approach in extracting the spatial feature of network-level traffic states and utilize a nested LSTM structure to improve the performance of time-series learning. The two methods are sequentially connected to build a deep learning architecture for the traffic prediction problem.

III. Methodology

A. Network Representation

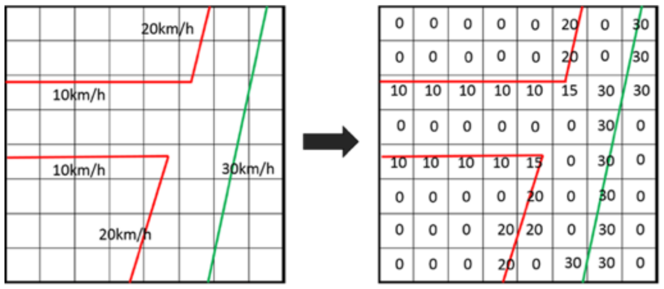
The traffic state of a roadway link in a road network is defined by the average speed of vehicles that travel on that link. The average speed of a link a in period t is calculated as follows:

$$V_{a,t} = \frac{\sum_{i=1}^{k} V_{a,i,t}}{k} \tag{1}$$

where a $\in (1, 2, \cdots, \text{n})$, k is the number of vehicles passing through the link during the time interval, and $V_{a,i,t}$ represents the average speed of each vehicle.

To learn the traffic as an image, the average speed of each link is projected in the road network combined with a GIS map to establish the spatial correspondence between the links and traffic states. As shown in **Fig. 2**, we initially visualize the speed values on the links with different colors and convert the road network with speed values to a single-channel grayscale image for further processing.

The traffic states of the road network are characterized by matrix images through a gridding process. The road network is divided into multiple grids with a certain spatial latitude and longitude range. The schematic of processing is shown in **Fig. 3**, in which a small part of the road network is used as an example. First, the road network is segmented by grids with a size of $0.0001° \times 0.0001°$ (latitude and longitude), which guarantees that the two links on a road with opposite directions can be separated into different grids in the studied area. Subsequently, the value of each grid is determined on the basis of the speed of links using the following criteria: if no link passes through the grid area, then the value is zero; if only one link passes through the grid area, then the value is the speed of this link; if multiple links pass through the same grid area, then we assign their average speed to the grid.



0	0	0	0	0	20	0	30
0	0	0	0	0	20	0	30
10	10	10	10	10	15	30	30
0	0	0	0	0	0	30	0
10	10	10	10	15	0	30	0
0	0	0	0	20	0	30	0
0	0	0	20	20	0	30	0
0	0	0	20	0	30	30	0

Fig. 3. Schematic of the gridding process.

On the basis of the above process, each grid is taken as a pixel with one channel, in which its value is the projected velocity value. Sequences of images are generated as data samples, and the time interval in these sequences is 2 min. These images not only represent the traffic state but also contain the spatial structure of the road network and the relative topology among different links.

B. Spatial Features Captured by CapsNet

In the aforementioned review, the CNN approach has shown promising results in capturing the spatial relationships among the links in urban road networks, where the congestion in one link not only affects its most adjacent links but may also propagate to other far-side regions. However, this method has several important drawbacks. First, the CNN approach extracts the spatial dependencies on many distant links by using successive convolutional layers or max pooling, where valuable information is lost. Second, the CNN approach cannot effectively distinguish between two links that are not spatially

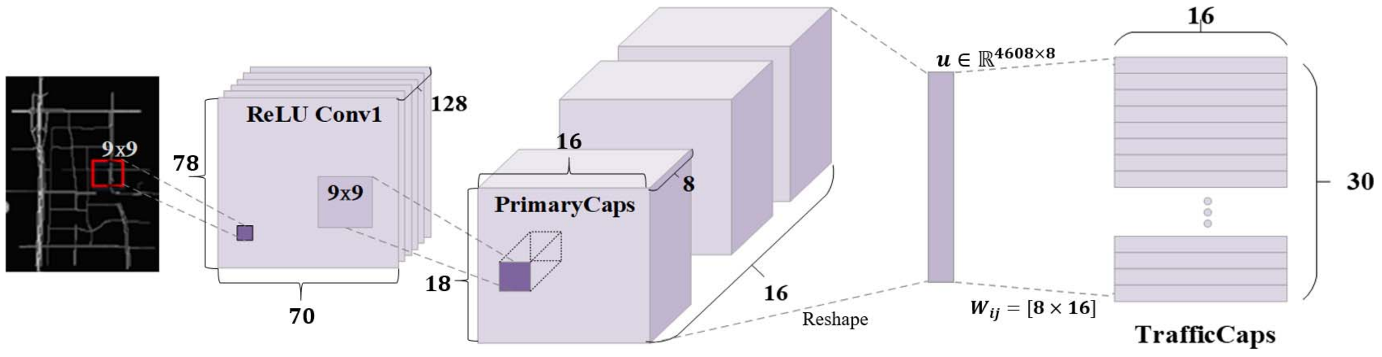


Fig. 4. Layers of CapsNet. The computational process in capsule networks is shown by taking the traffic state image as the input.

connected for several complicated road structures, such as viaducts. Third, the CNN approach cannot sufficiently handle the overlapping areas due to the low resolution in the grid-based image process. Thus, in this study, a CapsNet is utilized to solve the limitations of the CNN approach in extracting the spatial features of network-level traffic states.

A CapsNet is a new type of NN structure that is characterized by the use of "capsules" in a vector form rather than traditional scalar forms of neurons. Since the colors on traffic states images representing the speed information changes over time, the capsules may be able to capture the color changes and encode the color features in the vector form with higher dimensions. Particularly, the length of an output vector encodes the detection probability of a feature. The direction of the vector encodes the state of the features, such as rotation angle, direction, and size. The CapsNet inherently can detect multiple objects. Therefore, the CapsNet can effectively distinguish the spatial interlaced links and overlapping regions on several complicated road structures and low-resolution problem in traffic images by using such state of the features implied in the output vectors. In addition, the CapsNet can retain all the extracted local features by replacing the pooling operation with a dynamic routing operation between the capsules and thus avoid the problem of missing several spatial relationships among the links.

In this study, the CapsNet model is composed of two convolutional layers and a fully connected layer (called TrafficCaps), as shown in **Fig. 4**. For the input images that represent the traffic state of the road network, the first convolutional layer is used to extract the spatial relations between the adjacent links, that is, the local features of traffic states. The second convolution layer is then utilized in the primary capsule layer (named PrimaryCaps), and the "neurons" that use a single scalar output are converted to primary capsules in the vector form with a dimension of 8. Finally, the TrafficCaps is employed to capture the spatial relationship between the local features implied in all primary capsules and to output the features to a set of advanced capsules with a dimension of 16. The details of each part are subsequently explained.

In CNNs, high-level neurons receive input scalars from low-level neurons through weighting operations and activation functions, and the weights are learned by backpropagation [36]. By contrast, the weighting operations, activation functions, and learning method of weights between primary and advanced capsules are different because the capsules are in the vector form in the CapsNet.

The first convolution layer is the same as the convolutional layer in CNN by using ReLU as the activation function. The latter two layers use a novel nonlinear "squashing" activation function for the vector form of capsules, as shown as follows:

$$v_j = \frac{\|s_j\|^2}{1+\|s_j\|^2}\frac{s_j}{\|s_j\|} \tag{2}$$

where v_j is the output vector of capsule j, and s_j is the input vector. The squashing operation ensures that the short vectors shrink to approximately zero length and long vectors shrink to a length slightly below 1. Thus, the length of the output vector of a capsule can represent the probability of the existence of the extracted local features.

Taking the traffic image as an input, the computational process in capsule networks is shown **Fig. 4**. The first convolution layer, i.e. the "Conv1" layer, extracts low level features from the traffic image. Then the PrimaryCaps layer applies convolutional operations on these features and a 3-dimentional (3D) matrix with the dimension of 18*16*128 is obtained. This matrix is grouped into 16 capsules (groups), each of which is a matrix with the dimension of 18*16*8. By applying the squashing activation function on the capsules and by flattening/stacking the capsules, 4608 capsules are obtained. Each capsule, i.e. the extracted feature u_i, is in the vector form having 8 dimensions. The parameters of each layer are introduced in the experiments section.

To obtain the spatial relationship between the local features of network-level traffic state extracted by the PrimaryCaps layer and advanced features, an affine transformation is performed by multiplying the local features with a weight matrix W_{ij}.

$$\hat{u}_{j|i} = W_{ij}u_i \tag{3}$$

where u_i is the local features extracted by a primary capsule i, and $\hat{u}_{j|i}$ is the input vector associated with an advanced capsule j.

For the TrafficCaps, input s_j to an advanced capsule j is the weighted sum over all input vectors $\hat{u}_{j|i}$ from the primary capsules in the layer.

$$s_j = \sum_i c_{ij}\hat{u}_{j|i} \tag{4}$$

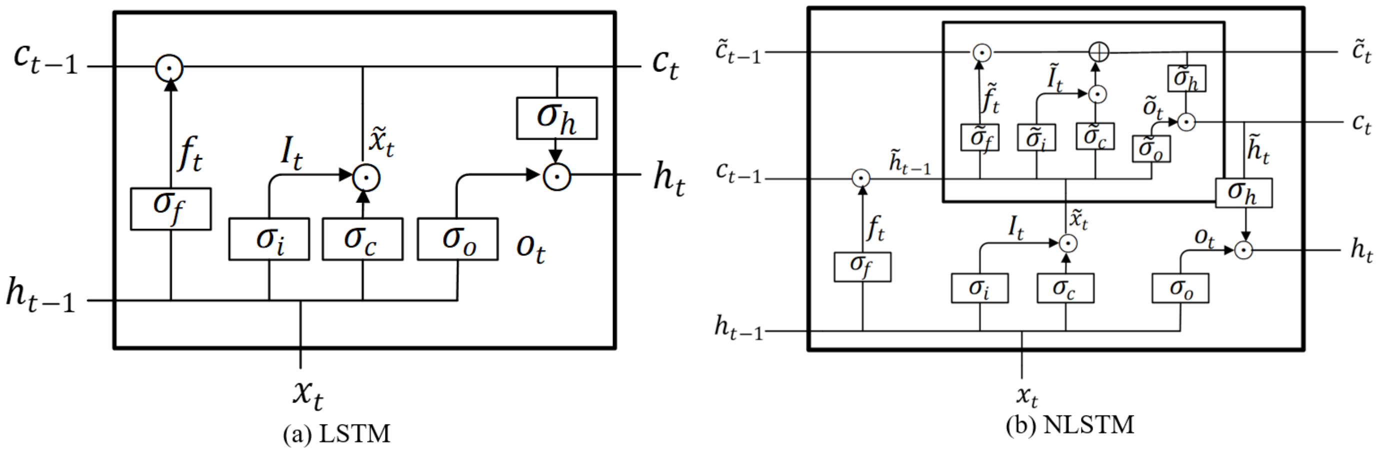


Fig. 5. LSTM and NLSTM architectures.

where weights c_{ij} are the coupling coefficients that determined by an iterative dynamic routing algorithm [11]. The essence of the dynamic routing algorithm is to find a part of primary capsules that is highly correlated to the advanced capsules, that is, to determine the local features with high probability to be associated with the high-level feature. This process represents the capability of the model to explore the spatial relationships among the distant links. For example, the dynamic routing algorithm will associate advanced capsule j with a set of primary capsules that contains the local features that affect congestion when it represents a severe congestion at a viaduct. The specific process of the dynamic routing algorithm is described as follows.

1) For each primary capsule i in the PrimaryCaps layer, the coupling coefficients c_{ij} with all the advanced capsules j are summed to 1 by using a SoftMax function:

$$c_{ij} = \frac{\exp\left(b_{ij}\right)}{\sum_k \exp\left(b_{ik}\right)} \tag{5}$$

where routing logit b_{ij} is the log prior probability that capsule i should be coupled to capsule j, and output c_{ij} represents the normalized probability that primary capsule i is associated with advanced capsule j. In the first iteration, the initial value of routing logit b_{ij} is set to zero in which the probabilities of the primary capsule accepted by each advanced capsule are equal.

2) After all the weights c_{ij} are calculated for all the primary capsules, each advanced capsule j of the TrafficCaps is weighted by using Equation (4).

3) In this step, all the capsules from the last step are activated by the squashing nonlinear function, as shown in Equation (2). In this process, the direction of the vector is preserved in output v_j and its length is enforced to be less than 1, which corresponds to the detection probability of high-level features.

4) In the iteration process, the initial coupling coefficients are iteratively refined by updating b_{ij} on the basis of the following rule:

$$b_{ij} = b_{ij} + \hat{u}_{j|i} \cdot v_j \tag{6}$$

Routing logit b_{ij} is updated by using the dot product of the input to capsule j and its output. In the field of mathematics, the dot product becomes large for similar vectors. Therefore, the corresponding routing logit increases when the input and output are similar; thus, the primary capsule is coupled to the advanced capsule with a similar output. This process represents the association of local features with the high-level feature.

5) The algorithms from Steps 1–4 are repeated several times to obtain the optimal routing weights. The dynamic routing algorithm is easy to be optimized, and experiments show that the CapsNet model can be optimized by iterating three times on an MNIST dataset [37].

Overall, a set of vectors is generated to express the spatial features of network-level traffic state by applying the CapsNet model on the input traffic images for subsequent operations in the next step.

C. Long Short-Term Temporal Features Captured by NLSTM

The traffic state normally has strong time evolution patterns and long-term dependencies, and a congestion state may last for several hours. LSTMs, whose architecture is shown **Fig. 5**(a), introduce memory units to learn whether to forget previous hidden states and update hidden states and achieve promising learning capability of long-term time series [12]. In this study, a novel nested architecture of LSTMs, which is evaluated to outperform stacked and single-layer LSTMs on various character-level language modeling tasks [13], is used to capture the temporal features of traffic state. In stacked LSTMs [7], all the information extracted from a low LSTM must be inputted to the subsequent high-level LSTM layer and must be filtered again.

NLSTMs add depth to LSTMs by nesting rather than stacking. As shown in **Fig. 5**(b), the value of a memory cell in an NLSTM is computed by using an LSTM structure, which acts as an internal unit that has its own inner memory cells. The long-term information learned by the internal unit can be selectively read and written by using the standard LSTM gates. This process enables the inner memories to remember and process traffic events on long time scales, especially when these events are irrelevant to the immediate present. Such selective access to inner memories in NLSTM exhibits a

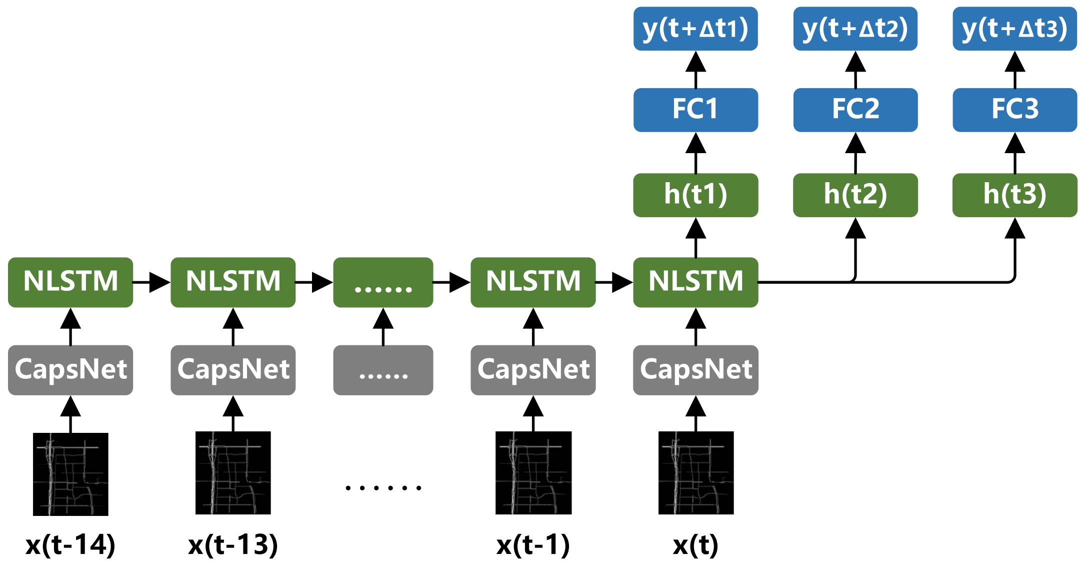


Fig. 6. Architecture of the prediction model (FC = Fully connected layers).

stable and efficient performance in capturing the long-term dependencies of traffic states.

The equations that update the cell state and gates in an internal LSTM unit are similar to the standard LSTM unit, as shown as follows (the parameters with superscript ~ belong to the internal LSTM unit):

$$\widetilde{I}_t = \widetilde{\sigma}_i\left(\widetilde{x}_t \widetilde{W}_{xi} + \widetilde{h}_{t-1}\widetilde{W}_{hi} + \widetilde{b}_i\right) \tag{7}$$
$$\widetilde{f}_t = \widetilde{\sigma}_f\left(\widetilde{x}_t \widetilde{W}_{xf} + \widetilde{h}_{t-1}\widetilde{W}_{hf} + \widetilde{b}_f\right) \tag{8}$$
$$\widetilde{c}_t = \widetilde{f}_t \odot \widetilde{c}_{t-1} + \widetilde{I}_t \odot \widetilde{\sigma}_c\left(\widetilde{x}_t \widetilde{W}_{xc} + \widetilde{h}_{t-1}\widetilde{W}_{hc} + \widetilde{b}_c\right) \tag{9}$$
$$\widetilde{o}_t = \widetilde{\sigma}_o\left(\widetilde{x}_t \widetilde{W}_{xo} + \widetilde{h}_{t-1}\widetilde{W}_{ho} + \widetilde{b}_o\right) \tag{10}$$
$$\widetilde{h}_t = \widetilde{o}_t \odot \widetilde{\sigma}_h\left(\widetilde{c}_t\right) \tag{11}$$

where $\widetilde{x}_t$ and $\widetilde{h}_{t-1}$ are the inputs of the internal LSTM unit and are calculated based on the parameters of the external unit, as shown as follows:

$$\widetilde{x}_t = I_t \odot \sigma_c\left(x_t W_{xc} + h_{t-1} W_{hc} + b_c\right) \tag{12}$$
$$\widetilde{h}_{t-1} = f_t \odot c_{t-1} \tag{13}$$

where $\widetilde{I}_t$, $\widetilde{f}_t$, and $\widetilde{o}_t$ are the three states of the gates; $\widetilde{c}_t$ is the cell input state; $\widetilde{W}_{xi}$, $\widetilde{W}_{xf}$, $\widetilde{W}_{xo}$, and $\widetilde{W}_{xc}$ are the weight matrices that connect $\widetilde{x}_t$ to the three gates and cell input; $\widetilde{W}_{hi}$, $\widetilde{W}_{hf}$, $\widetilde{W}_{ho}$, and $\widetilde{W}_{hc}$ are the weight matrices that connect $\widetilde{h}_{t-1}$ to the three gates and cell input; $\widetilde{b}_i$, $\widetilde{b}_f$, $\widetilde{b}_o$, and $\widetilde{b}_c$ are the biases of the three gates and cell input; σ represents the sigmoid function; and $\odot$ represents the scalar product of two vectors.

For the external LSTM unit, only the cell state update rule is changed to the output of the internal LSTM.

$$c_t = \widetilde{h}_t \tag{14}$$

In this study, the temporal features of the traffic state are iteratively calculated by using the NLSTM model for traffic prediction.

D. Framework

The spatiotemporal features of the traffic state can be learned by the CapsNet and the NLSTM. We sequentially integrate CapsNet and NLSTM to forecast the future traffic states. The outputs of the CapsNet are spread in one vector and are passed to the NLSTM as the input, as shown as follows:

$$x_t = \left\{v_j^t\right\}_{j=1}^{p} \tag{15}$$

where v_j^t is the output vector of advanced capsule j at timestamp t, and p is the number of advanced capsules. At the end of the model, a fully connected layer is added after the NLSTM model to obtain the predictions of the traffic states of all links. The predicted speed is calculated as follows:

$$y^{t+1} = w \times h_t + b \tag{16}$$

where h_t is the output of the NLSTM; and w and b represent the weight and bias between the hidden layer and the fully connected layer, respectively. y^{t+1} is the final output vector with the size of the number of links.

The entire prediction model of the network-level traffic state is shown in **Fig. 6**. The model is trained from end to end, and multi-step predictions are conducted based on the historical data of several steps.

IV. Experiments and Comparisons

A. Data Description

The traffic data used in the experiment is aggregated roadway link-based traffic speed data provided by Gaode Maps.[1] The raw speed data were collected from the GPS devices mounted on floating vehicles. The time interval of data uploading approximately ranged from 10 s to 1 min, which depends on the sampling resolutions of GPS devices. Generally, narrow intervals may generate invalid data with average speed of zero

[1]https://gaode.com/

TABLE I
MODEL STRUCTURE OF CAPSNET+NLSTMS

Name of layers	Parameters	Output	Parameter scale
Input		164 × 148 × 1	0

Convolution	Kernel size = 9 × 9
	Channels = 128
	Stride = 2
		78 × 70 × 128	10,496

PrimaryCaps (Convolution)	Kernel size = 9 × 9
	Channels = 128
	Stride = 4
		18 × 16 × 128	1,327,232

Reshape	Capsule dimension = 8	4,608 × 8	0
TrafficCaps (Fully connected)	Advanced capsule = 30 Capsule dimension = 16	30 × 16	17,694,720
(Flattened)		480	0
NLSTM	Hidden unit = 800	800	9,222,400
Dropout	0.2	800	0
Fully connected		278	222,678
Total parameters			28,477,526

TABLE II
MODEL STRUCTURE OF CNN+LSTMS

Name of layers	Parameters	Output	Parameter scale
Input		164 × 148 × 1	0
Convolution1 Pooling1	Filter (3 × 3 × 16) Pooling (2 × 2)	82 × 74 × 16	160
Convolution2 Pooling2	Filter (3 × 3 × 32) Pooling (2 × 2)	41 × 37 × 32	4,640
Convolution3 Pooling3	Filter (3 × 3 × 64) Pooling (2 × 2)	21 × 19 × 64	18,496
Convolution4 Pooling4	Filter (3 × 3 × 128) Pooling (2 × 2)	11 × 10 × 128	73,856
Flattened		14,080	0
LSTM1	Hidden unit = 800	800	47,619,200
LSTM2	Hidden unit = 800	800	5,123,200
Fully connected		278	222,678
Total parameters			53,062,230

and affect the traffic prediction performance. Thus, the raw data were processed by Gaode Maps and the time interval was aggregated to 2 min to capture the traffic state variations of the road network accurately.

The evaluated roadway network in this study encompasses 278 links, which include arterial roads, interchanges, and intersections, that are located between the Second and Third Ring Roads in Beijing. After the gridding process, the traffic states of the network are represented by an image with a size of 164 × 148 for 2 min. The dataset used in this study was divided into two subsets for training and testing to validate the effectiveness of the proposed prediction model. The training set was collected from June 1 to July 1, 2015, and the test set was collected from August 1–14, 2015.

In the experiment, the time lag of the input sequence was set to 15, which indicated that the traffic states of the previous 30 min were used as the input of the proposed model. The 30 min historical traffic speeds were used to predict the following 2, 10, and 20 min traffic speeds, which corresponded to the number of the time lags $(\Delta t_1, \Delta t_2, \Delta t_3) = (1, 5, 10)$ in Fig. 6.

B. Implementation

1) Hardware: The deep learning model was implemented by using Python Keras [38] and was executed on a server with 8 NVIDIA GeForce Titan X GPUs (12 GB RAM).

2) Baseline Models: We compared the proposed model with nine baseline deep NN models, namely, LSTM [12], NLSTM [13], CNN [4], 3D CNN [41], CapsNet [11], CNN+LSTM [8], CapsNet+LSTM, CNN+NLSTM, and 3D CNN+NLSTM, to evaluate its prediction performance. The parameters of the proposed CapsNet+NLSTM model are shown in **Table I**. The details of the CNN+LSTM model are shown in **Table II**, whose total parameters are approximately half of the CapsNet+NLSTM model. For the LSTM, NLSTM, CNN, and CapsNet models, their structures were the same as

TABLE III
COMPARISON AMONG DIFFERENT METHODS

Prediction horizon / Algorithms	2 min		10 min		20 min	
	MSE	MAPE	MSE	MAPE	MSE	MAPE
LSTM	41.67	0.2158	44.67	0.2255	48.11	0.2273
NLSTM	39.55	0.2067	44.49	0.2229	47.32	0.2246
CNN	42.94	0.2131	47.14	0.2367	51.38	0.2384
3D CNN	37.70	0.2211	43.95	0.2371	46.50	0.2525
CapsNet	35.80	0.1891	42.53	0.2205	47.08	0.2308
CNN+LSTM	36.57	0.2051	43.10	0.2181	45.90	0.2258
CapsNet+LSTM	38.87	0.2227	43.79	0.2443	46.35	0.2493
CNN+NLSTM	33.21	0.2036	41.15	0.2342	43.99	0.2453
3D CNN+NLSTM	38.25	0.2184	45.37	0.2525	46.49	0.2533
CapsNet+NLSTM	**31.04**	**0.1757**	**39.29**	**0.2071**	**42.88**	**0.2183**

the part of the two combined models. The LSTM model was constructed by stacking two standard LSTMs with 800 hidden units. The NLSTM model was a nested structure with the same 800 hidden units. For the single model of CNN and CapsNet, a flattened layer was added on the fully connected layer to integrate the outputs of 15 time-steps into one vector for prediction.

3) Model Parameters and Hyper-Parameters: The model input has three dimensions, where the first two dimensions represent the resolution of the input image, and the last dimension indicates the amount of channel of the input image. The model was trained using the optimizer RMSprop [39], and the batch size was set to 32. For the CNN-based baseline models, the initial learning rate was set as 0.001 and reduced to half every 20 epochs. The initial learning rate of 3D CNN-based models was set to 0.001 and reduced to half every 40 epochs. As for CapsNet-based models, the initial learning rate was 0.001 and reduced to half very 25 epochs. A dropout layer and the early stopping mechanism were applied to prevent the problem of overfitting [40]. In addition, a fivefold cross-validation was used to determine the parameters of our deep learning model. In the cross-validation, the train set was divided into five subsets. Four subsets were used for training, and the remaining subset was used for validation. The optimal model has the lowest average prediction error in all validation datasets.

4) Evaluation Metrics: The deep learning models in this study were evaluated by using two commonly used metrics in traffic forecasting, namely, mean squared error (MSE) and mean absolute percentage error (MAPE), which can be expressed as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{N}\left(\hat{y}_i - y_i\right)^2 \tag{17}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{N}\left(\frac{\hat{y}_i - y_i}{y_i}\right) \tag{18}$$

where $\hat{y}_i$ is the prediction result of sample i, and y_i is the ground truth of the corresponding traffic speed.

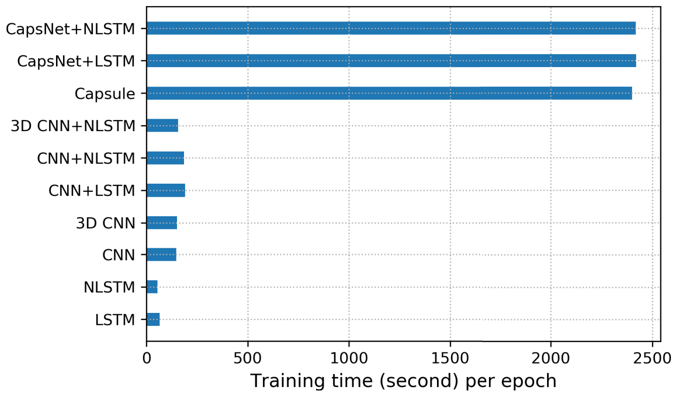
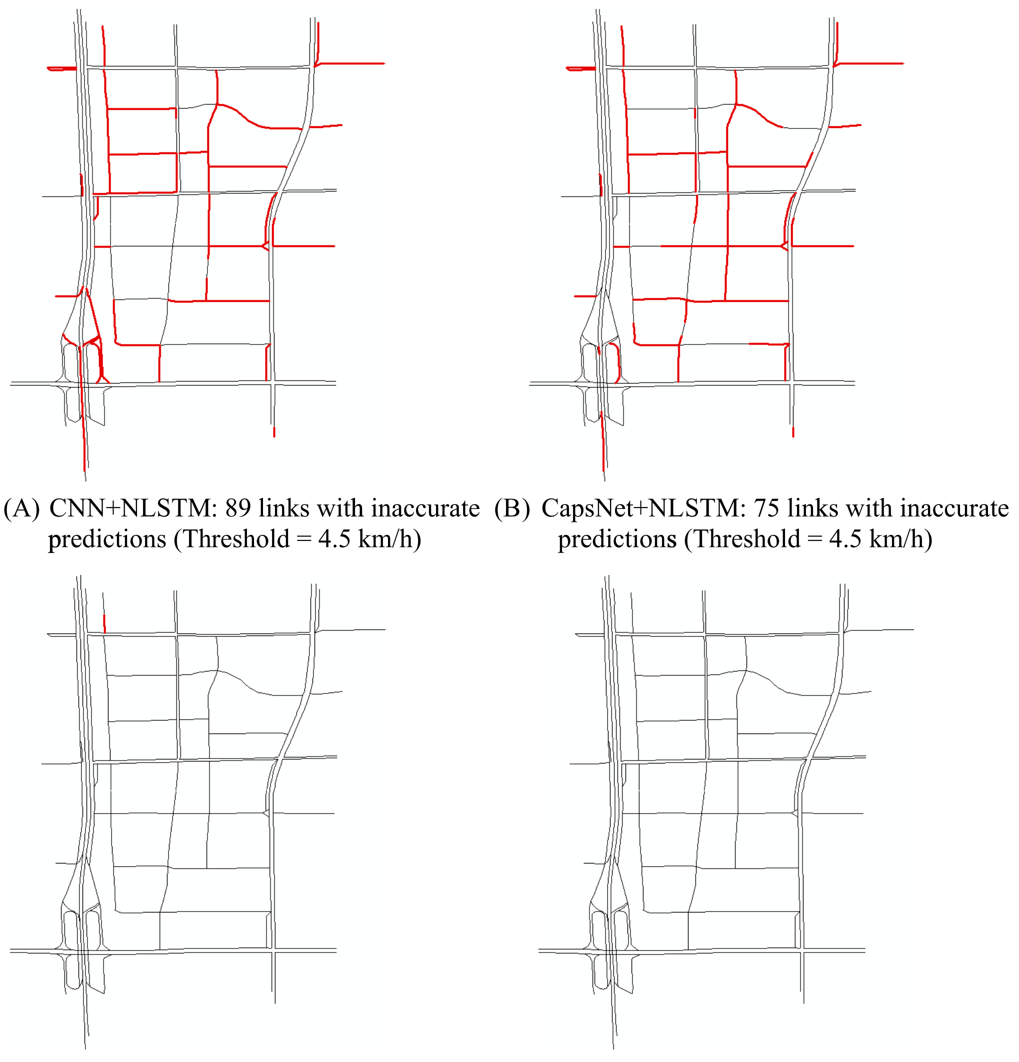


Fig. 7. Training time comparison.

C. Experimental Results

1) Comparison: In this section, we compared our CapsNet+NLSTM model with the other nine baseline models and evaluated the prediction results by using the MSE and MAPE metrics. **Table III** shows the comparison of different models for 1, 5, and 10 step-ahead predictions. Among the three prediction steps, our CapsNet+NLSTM model yields the most accurate results in terms of MSE and MAPE. The average MSE values for CNN+LSTM decrease by 15%, 8.8% and 6.6%. The CapsNet model performs better than the CNN model with 16.6%, 9.8%, and 8.4% lower MSE. Further, the MSE and MAPE of CapsNet+NLSTM are smaller than those of other NLSTM-based models, i.e. the CNN+NLSTM and the 3D CNN+NLSTM. This finding indicates that CapsNet shows stronger capability compared with CNNs in terms of the extraction of spatial features. For long-term temporal features, the prediction error increases with the prediction horizon, and the gap between the NLSTM and CapsNet+NLSTM models becomes small. This phenomenon indicates that the temporal features play an important role for the traffic prediction with the increase of prediction step size. Notably, the proposed model utilizes less parameters compared with the baseline models and achieves more accurate results, which indicates that the CapsNet+NLSTM model achieves superior



(A) CNN+NLSTM: 89 links with inaccurate predictions (Threshold = 4.5 km/h)

(B) CapsNet+NLSTM: 75 links with inaccurate predictions (Threshold = 4.5 km/h)

(C) CNN+NLSTM: 2 links with inaccurate predictions (Threshold = 9 km/h)

(**D**) CapsNet+NLSTM: 0 links with inaccurate predictions (Threshold = 9 km/h)

Fig. 8. Visualization of prediction results. (The links with inaccurate predictions whose mean absolute errors are more than the threshold are marked in red.)

performance in predicting traffic states and shows a promising potential to be utilized.

2) Evaluation of Training Time: In this section, we compared the training time of CapsNet+NLSTM and all other baseline models. A bar chart of the training time of all models is shown in **Fig. 7**. Although the NLSTM and the LSTM have the same amount of hidden units, NLSTM-based models run slightly faster than LSTM-based models. Since the dynamic routing operations in the CapsNet are computationally expensive, the CapsNet-based models cost more training time than the CNN-based models, as shown in **Fig. 7**.

3) Evaluation of CapsNet: We visualized and compared the prediction results of the CapsNet+NLSTM and CNN+NLSTM models to evaluate the superior capability of CapsNet in extracting the spatial features of traffic states implied in the traffic images of complicated road networks. As shown in **Fig. 8**, we highlighted the links with a mean absolute error of speed more than the threshold, which were considered inaccurate predictions. When the threshold is set as 4.5 km/h, in comparison with the CNN+NLSTM prediction results in **Fig. 8** (A) with 89 inaccurate links, the accuracy of CapsNet+NLSTM model exhibits an outstanding improvement with only 75 links highlighted in **Fig. 8** (B). When the threshold increases to 9 km/h, the prediction results of CNN+NLSTM and CapsNet+NLSTM only have two and zero links with inaccurate predictions, as shown in Fig. 8 (C) and (D), respectively. Furthermore, the inaccurate prediction results in **Fig. 8** (A) are mainly concentrated in the viaducts and low-resolution areas with links that are tightly arranged. This condition verifies the proposed models' capability in handling complicated road networks.

In order to validate the CapsNet+NLSTM's superiority of dealing with interlaced and compact links, 38 road links in the viaduct area are selected for testing the prediction accuracy, as shown in **Fig. 9** (A). To avoid the road links forwarding opposite directions fall in the same grid, the road network is re-segmented by grids with the size of $0.00005^\circ \times 0.00005^\circ$ (latitude and longitude). When the threshold of

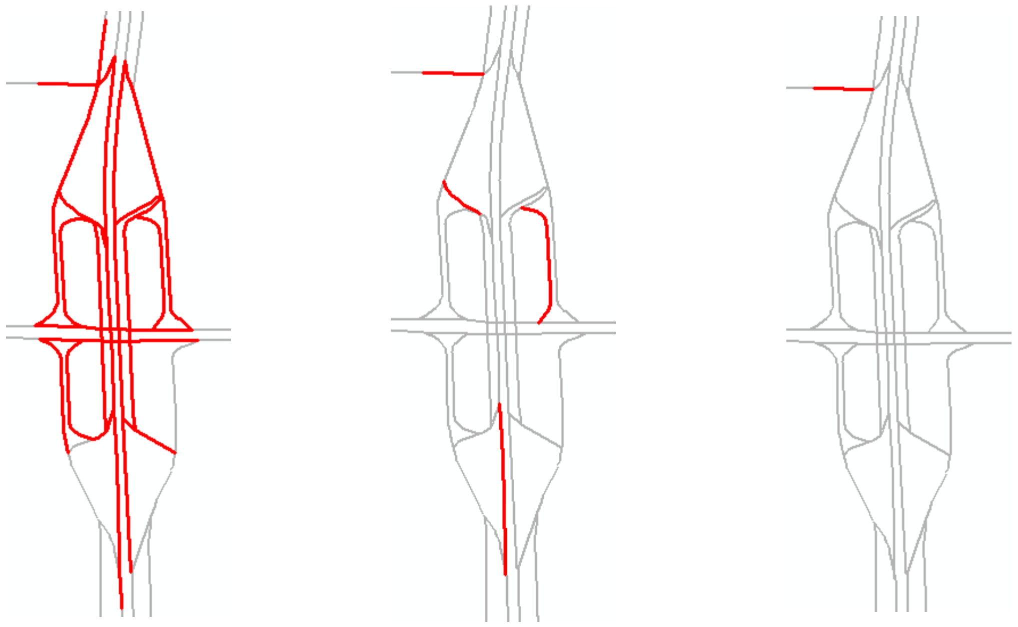

(A) 38 road links with red color in the viaduct are selected for the experiment.
(B) CNN+NLSTM: four links with inaccurate predictions (Threshold = 4.5 km/h)
(C) CapsNet+NLSTM: one links with inaccurate predictions (Threshold = 4.5 km/h)

Fig. 9. Visualization of prediction results in the viaduct area. (The links with inaccurate predictions whose mean absolute errors are more than the threshold are marked in red.)

TABLE IV
COMPARISON OF CNN+NLSTM AND CAPSNET+NLSTM

Prediction horizon / Algorithms	2 min		10 min		20 min	
	MSE	MAPE	MSE	MAPE	MSE	MAPE
CNN+NLSTM	23.21	0.1673	33.18	0.1987	41.76	0.2385
CapsNet+NLSTM	19.84	0.1372	32.11	0.1828	40.19	0.2111

TABLE V
COMPARISON OF NLSTMS AND LSTMS

Metrics	Prediction horizons	2 min		10 min		20 min	
	Time lags	30 min	40 min	30 min	40 min	30 min	40 min
MSE	LSTMs	41.67	43.09	44.67	45.12	48.11	47.32
	NLSTMs	39.55	39.41	44.49	44.65	47.32	49.96
Efficiency (s)	LSTMs	42	47	34	46	34	45
	NLSTMs	19	23	18	23	20	22

inaccurate prediction is set as 4.5 km/h, the road links with inaccurate predictions generated by CNN+NLSTM and CapsNet+NLSTM are visualized by **Fig. 9** (B) and **Fig. 9** (C), respectively. The CapsNet+NLSTM obviously outperforms the CNN+NSTLM that the results of CapsNet+NLSTM only have one inaccurate predicted link. The prediction results under different prediction horizons, as shown in **Table IV**, also indicate that the CapsNet+NLSTM achieves superior prediction performance, especially when the prediction horizon is set as 2 minutes. Thus, the comparison of the prediction results indicates that the CapsNet-based model can achieve better prediction accuracy for traffic networks with specific complicated road network structures.

4) Evaluation of NLSTM: Moreover, we compared the NLSTM with LSTM in a long time lag with 40-min historical traffic speeds as inputs to evaluate the performance of NLSTM in learning long-term features. The results are shown in **Table V**.

In comparison with LSTMs, the MSE values of NLSTM fluctuate more slightly when using long-term historical data in predicting multistep traffic speeds, especially in the 2-min interval short-term prediction. In terms of the number of parameters, the stacked and nested structures have the same scale of 8.8 M, and the NLSTM algorithm consumes less time in performing the prediction. These comparison results indicate that the NLSTM exhibits a stable and efficient performance in learning a long time series, which is the same as we expected.

Overall, the CapsNet outperforms the CNNs in capturing the spatial features of traffic states, and the NLSTM performs better in terms of stability and efficiency compared with the stacked LSTMs. These results verify the superiority and feasibility of our CapsNet+NLSTM model, which shows promising potential in forecasting the traffic states of large-scale urban road networks.

V. Conclusion

Traffic prediction remarkably influences the overall performance of traffic management and control systems. In this study, a CapsNet+NLSTM approach is presented to address the important drawbacks of statistical models and machine learning methods in handling the complex spatial relationships among the links when performing network-level traffic state prediction. We use the traffic roadway network as an image to capture the spatial structure of the road network and the relative topology among the different links. Many spatial relationships among the links are preserved, and considerable spatial features of the network are encapsulated in the vector form of capsules, such as position, direction, length, and travel speed of the road segment, by using the new CapsNet rather than conventional CNNs. The incorporated NLSTM model can achieve a stable performance in time-series prediction compared with the traditional stacked structure of LSTM. The experimental results indicate that the CapsNet+NLSTM model outperforms other baseline models.

The major contributions of this study are summarized as follows. (1) A new CapsNet is developed to extract the comprehensive spatial features of roadway networks. (2) An NLSTM model is sequentially incorporated to capture the hierarchical temporal dependencies of traffic states. (3) The proposed model with the capability of capturing complicated spatiotemporal traffic patterns achieves the best prediction performance compared with the baseline models. (4) The visualized prediction results display the proposed model's promising capability in handling complicated road networks that contain interlaced and compact links, such as viaducts and side roads.

It is admitted that training CapsNet may be more time consuming than training CNN. Several endeavors are conducting to release the computational burden of CapsNet: (1) Modify the dynamic routing mechanism by introducing a novel 3D convolutional operation (Rajasegaran *et al.*, 2019); (2) Accelerate training of CapsNet via lightweight software-level optimizations (Marchisio *et al.*, 2019). With the advent of more advanced computing hardware and software techniques, we believe that the proposed CapsNet and NLSTM framework is potential to be applied in large-scale complicated transportation systems in the near future.

Several potential extensions in this research are considered. For example, the dynamic routing algorithm between capsules will be improved. Specifically, the prediction accuracy and model efficiency should be increased because the dynamic routing algorithm is the core component of the CapsNet. Furthermore, the interpretation of the learned spatiotemporal features captured by the CapsNet and NLSTM will be investigated in the future.

References

[1] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 187–197, May 2015.

[2] D. Park and L. R. Rilett, "Forecasting freeway link travel times with a multilayer feedforward neural network," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 14, no. 5, pp. 357–367, Sep. 1999.

[3] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2191–2201, Oct. 2014.

[4] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[5] J. W. C. V. Lint, S. P. Hooqendoorn, and H. J. V. Zuvlen, "Freeway travel time prediction with state-space neural networks: Modeling state-space dynamics with recurrent neural networks," *Transp. Res. Rec. J. Transp. Res. Board*, vol. 1811, no. 1, pp. 347–369, 2002.

[6] J. Zhang, Y. Zheng, and D. Qi, "Deep Spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. AAAI*, 2016, pp. 1655–1661.

[7] Z. Cui, R. Ke, and Y. Wang, "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction," in *Proc. 6th Int. Workshop Urban Comput.*, 2017, pp. 1–8.

[8] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, p. 1501, 2017.

[9] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–16.

[10] Y. Zhang, Y. Zhang, and A. Haghani, "A hybrid short-term traffic flow forecasting method based on spectral analysis and statistical volatility model," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 65–78, Jun. 2014.

[11] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3856–3866.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] J. R. A. Moniz and D. Krueger, "Nested LSTMs," in *Proc. Mach. Learn. Res.*, 2018, pp. 530–544.

[14] M. M. Hamed, H. R. Al-Masaeid, and Z. M. B. Said, "Short-term prediction of traffic volume in urban arterials," *J. Transp. Eng.*, vol. 121, no. 3, pp. 249–254, 1995.

[15] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining kohonen maps with arima time series models to forecast traffic flow," *Transp. Res. C, Emerg. Technol.*, vol. 4, no. 5, pp. 307–318, Oct. 1996.

[16] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov. 2003.

[17] B. M. Williams, "Multivariate vehicular traffic flow prediction: Evaluation of ARIMAX modeling," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1776, no. 1, pp. 194–200, Jan. 2001.

[18] J. Guo, W. Huang, and B. M. Williams, "Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 50–64, Jun. 2014.

[19] B. M. Williams, P. K. Durvasula, and D. E. Brown, "Urban freeway traffic flow prediction: Application of seasonal autoregressive integrated moving average and exponential smoothing models," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1644, no. 1, pp. 132–141, Jan. 1998.

[20] M.-C. Tan, S. C. Wong, J.-M. Xu, Z.-R. Guan, and P. Zhang, "An aggregation approach to short-term traffic flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 1, pp. 60–69, Mar. 2009.

[21] G. A. Davis and N. L. Nihan, "Nonparametric regression and short term freeway traffic forecasting," *J. Transp. Eng.*, vol. 117, no. 2, pp. 178–188, Mar. 1991.

[22] P. Cai, Y. Wang, G. Lu, P. Chen, C. Ding, and J. Sun, "A spatiotemporal correlative k-nearest neighbor model for short-term traffic multistep forecasting," *Transp. Res. C, Emerg. Technol.*, vol. 62, pp. 21–34, Jan. 2016.

[23] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statist. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004.

[24] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 276–281, Dec. 2004.

[25] M.-L. Huang, "Intersection traffic flow forecasting based on V-GSVR with a new hybrid evolutionary algorithm," *Neurocomputing*, vol. 147, pp. 343–349, Jan. 2015.

[26] S. H. Huang and B. Ran, "An application of neural network on traffic speed prediction under adverse weather condition," in *Proc. Transp. Res. Board 82nd Annu. Meeting*, Washington, DC, USA, Jan. 2003.

[27] C. Messer and I. I. Thomas Urbanik, "Short-term freeway traffic vol. forecasting, using radial basis function neural network," *Transp. Res. Rec. J. Transp. Res. Board*, vol. 1651, no. 1 pp. 39–47, 1998.

[28] J. Z. Zhu, J. X. Cao, and Y. Zhu, "Traffic volume forecasting based on radial basis function neural network with the consideration of traffic flows at the adjacent intersections," *Transp. Res. C, Emerg. Technol.*, vol. 47, pp. 139–154, Oct. 2014.

[29] B. Yang, S. Sun, J. Li, X. Lin, and Y. Tian, "Traffic flow prediction using LSTM with feature enhancement," *Neurocomputing*, vol. 332, pp. 320–327, Mar. 2019, doi: 10.1016/j.neucom.2018.12.016.

[30] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, pp. 1–9, 2014.

[31] T. Zhou *et al.*, "σ-agree AdaBoost stacked autoencoder for short-term traffic flow forecasting," *Neurocomputing*, vol. 247, pp. 31–38, Jul. 2017.

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 60, 2012, pp. 1097–1105.

[33] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proc. Comput. Vis. Pattern Recognit.*, 2014, pp. 1717–1724.

[34] Y. Tian, K. Zhang, J. Li, X. Lin, and B. Yang, "LSTM-based traffic flow prediction with missing data," *Neurocomputing*, vol. 318, pp. 297–305, Nov. 2018.

[35] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016, *arXiv:1612.01022*. [Online]. Available: http://arxiv.org/abs/1612.01022

[36] Y. Lecun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," *Adv. Neural Inf. Process. Syst.*, vol. 2, no. 2, pp. 396–404, 1990.

[37] Y. Lecun and C. Cortes. (2010). *The Mnist Database of Handwritten Digits*. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[38] F. Chollet. (2018). *Keras Documentation*. [Online]. Available: https://keras.io/

[39] T. Tieleman and G. Hinton, "Lecture 6.5-RM Sprop: Divide the gradient by a running average of its recent magnitude," *COURSERA, Neural Netw. Mach. Learn.*, vol. 4, pp. 26–31, Jun. 2012.

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[41] S. Guo, Y. Lin, S. Li, Z. Chen, and H. Wan, "Deep spatial–temporal 3D convolutional neural networks for traffic data forecasting," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3913–3929, Oct. 2019.

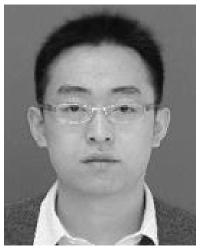
[42] J. Rajasegaran, V. Jayasundara, S. Jeyasekara, N. Jeyasekara, S. Seneviratne, and R. Rodrigo, "DeepCaps: Going deeper with capsule networks,"in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10725–10733.

[43] A. Marchisio *et al.*, "X-TrainCaps: Accelerated training of capsule nets through lightweight software optimizations," 2019, *arXiv:1905.10142*. [Online]. Available: http://arxiv.org/abs/1905.10142

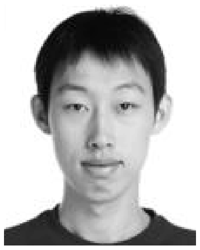
Xiaolei Ma is currently an Associate Professor with the Beijing Key Laboratory for Cooperative Vehicle Infrastructure Systems and Safety Control, School of Transportation Science and Engineering, Beihang University, Beijing, China. His research interests include urban transit optimization, data-driven transportation networks, and transportation data mining.

Houyue Zhong is currently pursuing the M.S. degree with the Beijing Key Laboratory for Cooperative Vehicle Infrastructure Systems and Safety Control, School of Transportation Science and Engineering, Beihang University, Beijing, China. His research interests include traffic data mining and public transit network optimization.

Yi Li is currently working as a Traffic Engineer with AutoNavi Software Company, Beijing, China. His main research interest is network-level traffic prediction.



Junyan Ma is currently an Associate Professor with the School of Information Engineering, Chang'an University, Xi'an, China. His interests include cyber-physical systems and connected autonomous vehicles with emphasis on using data analytics for system testing and reliability.



Zhiyong Cui is currently pursuing the Ph.D. degree in civil and environmental engineering with the University of Washington, Seattle, WA, USA. His research interests include deep learning in transportation, traffic data mining, and intelligent transportation systems.

Yinhai Wang received the master's degree in computer science from the University of Washington (UW) and the Ph.D. degree in transportation engineering from the University of Tokyo in 1998. He is currently a Professor in transportation engineering and the Founding Director of the Smart Transportation Applications and Research Laboratory (STAR Lab), UW. He also serves as the Director for Pacific Northwest Transportation Consortium (PacTrans), USDOT University Transportation Center for Federal Region 10. His active research fields include traffic sensing, e-science of transportation, and transportation safety.