Scalable Learning With a Structural Recurrent Neural Network for Short-Term Traffic Prediction

Youngjoo Kim, Peng Wang, and Lyudmila Mihaylova, *Senior Member, IEEE*

Abstract—This paper presents a scalable deep learning approach for short-term traffic prediction based on historical traffic data in a vehicular road network. Capturing the spatio-temporal relationship of the big data often requires a significant amount of computational burden or an ad-hoc design aiming for a specific type of road network. To tackle the problem, we combine a road network graph with recurrent neural networks (RNNs) to construct a structural RNN (SRNN). The SRNN employs a spatio-temporal graph to infer the interaction between adjacent road segments as well as the temporal dynamics of the time series data. The model is scalable thanks to two key aspects. First, the proposed SRNN architecture is built by using the semantic similarity of the spatio-temporal dynamic interactions of all segments. Second, we design the architecture to deal with fixed-length tensors regardless of the graph topology. With the real traffic speed data measured in the city of Santander, we demonstrate the proposed SRNN outperforms the image-based approaches using the capsule network (CapsNet) by 14.1% and the convolutional neural network (CNN) by 5.87%, respectively, in terms of root mean squared error (RMSE). Moreover, we show that the proposed model is scalable. The SRNN model trained with data of a road network is able to predict traffic speed of different road networks, with the fixed number of parameters to train.

Index Terms—Graph theory, intelligent transportation systems, machine learning, scalability, time series analysis.

I. INTRODUCTION

MAKING accurate predictions of traffic data in a road network of interest is one of the important tasks for building intelligent transportation systems. The traffic data are usually obtained by magnetic induction loop detectors installed on road segments. The traffic data include traffic speed and flow, where the term traffic flow is used interchangeably with the terms traffic count or traffic volume. The sequence of traffic data on each road segment is essentially a time series, which is also spatially related to traffic in different road segments. Capturing the spatio-temporal features of the traffic data has been of great interest of researchers.

Deep learning approaches have recently been applied to traffic prediction tasks, given the advancement of technologies

for obtaining and managing massive volume of traffic data. Early deep learning approaches to traffic prediction are based on convolutional neural networks (CNNs) [1], [2]. They have been demonstrated to be effective in exploring spatial features. Studies incorporating recurrent neural networks (RNNs) have also been reported, considering the traffic prediction as a time series forecasting. Different gating mechanisms like long short-term memories (LSTMs) [2], [3] and gated recurrent unit (GRU) [4] have been employed in various architectures. A novel approach has been introduced in [5] where the spatio-temporal traffic data are converted into images with two axes representing time and space and the images are fed into a CNN. This method enables the CNN model capture the spatio-temporal characteristics of the traffic data by learning the images. Recently, it has been demonstrated that a capsule network (CapsNet) architecture proposed in [6] outperforms the CNN-based method in large, complex road networks. The dynamic routing algorithm of the CapsNet replaces the max pooling operation of the CNN, resulting in more accurate predictions but more parameters to train. Gaussian process (GP) is another data-driven approach, also known as a kernel-based learning algorithm. GPs have been demonstrated to be powerful in understanding the implicit relationship between data to give estimates for unseen points. Comparative studies [7], [8] have shown that GPs are effective in short-term traffic prediction. However, they still suffer from cubic time complexity in the size of training data [9].

The main contributions of this work are as follows: i a structural RNN (SRNN) approach for traffic prediction that incorporates the topological information of the road network is proposed; ii) it is shown that the SRNN represents well both the spatial and temporal dynamics of the traffic; iii) the scalability of the proposed SRNN is demonstrated and validated with real data. Our work is mainly inspired by ideas from [10], [11]. We build a spatio-temporal graph representation of the road network by considering each road segment as a node. All the nodes and edges are associated with RNNs that are jointly trained. It has been demonstrated in our preliminary work [12] that the SRNN is more computationally efficient than the image-based state-of-the-art. In this paper, we introduce an advanced SRNN architecture with more details and provide a comprehensive demonstration for comparison with the image-based methods. More importantly, we show the proposed model is scalable; the SRNN model trained with a road network is capable of predicting traffic states in other road networks that have different network topologies (the number of

road segments and how they are connected with one another). The proposed SRNN model is evaluated with the real dataset from the case studies of the SETA EU project [13].

The rest of this paper is organized as follows. It starts with discussing the related work regarding deep learning methods for traffic prediction and SRNNs in Section II. We present the proposed architecture of the SRNN for traffic prediction in Section III. The proposed approach and results of the performance evaluation with a real dataset are given in Section IV. Finally, Section V discusses the results.

II. RELATED WORK

In this section, we give a detailed overview of the relevant literature about previous works on deep learning for traffic prediction and SRNN. This will help readers figure out the background and differentiate our work from the existing works.

A. Deep Learning for Traffic Prediction

Recently, the notion of big data has been introduced to transportation research [14] and deep learning approaches have been actively used to address traffic prediction problems. One of the early approaches is based on a stacked auto-encoder model [1]. Whereas this has a fully-connected structure, a method based on a CNN has been proposed [2]. However, this method treats the time dimension of the traffic flow as a channel of image data and therefore the temporal features of the traffic flow are ignored. The spatio-temporal features of the traffic have been of great interest of researchers.

One recent approach [5] converts the traffic data into an image with two axes representing time and space and applies a CNN to capture the features of the spatio-temporal images. A CapsNet has also been used to improve the performance [6] by effectively capturing the relationship between distant local features. A major drawback of these image-based methods is that the computational complexity increases as the number of road segments increases. Besides, a combination of a CNN, RNN, and attention model has been studied [4] for mining spatial features, temporal features, and periodic features separately. However, this model aims for a simple network topology that consists of road segments along a straight path without intersections.

In this paper we develop a compact neural network model that is capable of learning spatio-temporal relationship regardless of the type of network topology. To broaden the applicability, we assume we are using only the historical traffic data and the topological information expressed as an adjacency matrix. The traffic data and the topological information are generally available for any road networks of interest.

B. Structural Recurrent Neural Network

The SRNN has been proposed in [10] with the application in human motion forecasting, human activity anticipation, and driver maneuver anticipation where the connected components in a system of interest are represented as nodes in a graph. The SRNN is based on the spatio-temporal graphs which are usually used to model spatial and temporal reasoning [15]. The spatial and temporal interactions between nodes are

parametrized with a factor graph [16]. As the principles described in [10] implies, the SRNN is applicable to any systems that can be expressed as spatio-temporal graphs.

An attention model has recently been applied to the SRNN [11] to find subsets of human crowds within which humans interact with each other, instead of using a fixed graph to represent the connection among the humans. This approach deals with only one kind of nodes, humans, to predict human trajectories, which resembles the problem of short-term traffic prediction.

Inspired by these two works [10], [11] on the SRNN, we apply the SRNN to the short-term traffic prediction problem. We consider road segments as nodes that are semantically the same but use a road network topology to construct the spatio-temporal graph. Moreover, we verify the scalability of the SRNN by using training and evaluation datasets that have different network topologies.

III. STRUCTURAL RECURRENT NEURAL NETWORK FOR SCALABLE TRAFFIC PREDICTION

A. Problem Definition

In this study, we address the problem of short-term traffic speed prediction based on historical traffic speed data and a road network graph. Suppose we deal with N road segments where the loop detectors are installed. Let x_u^t represent the traffic speed on road segment u at time step t. Given a sequence of traffic speed data $\{x_u^t\}$ for road segments $u = 1, 2, \ldots, N$ at time steps $t = t_c - l + 1, \ldots, t_c$, we predict the future traffic speed $x_u^{t_c+1}$ on each road segment where t_c denotes the current time step and l denotes the length of a historical data sequence under consideration. The road network graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{G}$ denotes the graph, $\mathcal{V}$ denotes the set of nodes, and $\mathcal{E}$ denotes the set of edges connecting two nodes in $\mathcal{V}$. In this study, the nodes in the graph correspond to road segments of interest. Thus, $|\mathcal{V}| = N$. We use a directed adjacency matrix $\mathcal{A}$ to represent the edges in the graph. For example, suppose the traffic flow comes from node u to node v. This means there is an edge e linking nodes u and v. Thus, $\mathcal{A}(u, v) = 1$ and $e = (u, v) \in \mathcal{E}$. We say two nodes u and v are connected if either $(u, v) \in \mathcal{E}$ or $(v, u) \in \mathcal{E}$. Note that we use e to denote an edge in general or the form of (u, v) with parentheses to reveal the nodes connected by the edge.

B. Spatio-Temporal Graph Representation

The SRNN is constructed based on a spatio-temporal graph that is obtained by unrolling the spatial graph $\mathcal{G}$ over time. We use a spatio-temporal graph representation $\mathcal{G}_{ST} = (\mathcal{V}, \mathcal{E}_S, \mathcal{E}_T)$. Let $\mathcal{G}_{ST}$ denote the spatio-temporal graph. $\mathcal{V}$, $\mathcal{E}_S$, and $\mathcal{E}_T$ denote the set of nodes, the set of spatial edges, and the set of temporal edges, respectively. Note that $\mathcal{E} = \mathcal{E}_S$.

The spatial edges in $\mathcal{E}_S$ represent the dynamics of traffic interaction between two adjacent road segments, and the temporal edges in $\mathcal{E}_T$ represent the dynamics of the temporal evolution of the traffic speed in road segments. Fig. 1(a) shows an example spatio-temporal graph. Nodes $u, v, w \in \mathcal{V}$ represent road segments. The connections between the road
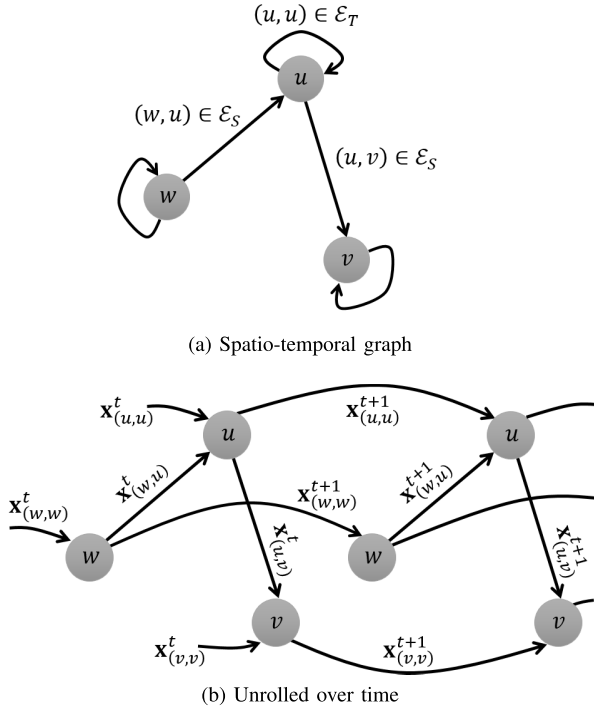
Fig. 1. An example spatio-temporal graph. (a) Nodes represent road segments and the nodes are linked by edges in $\mathcal{E}_S$ and temporal edges in $\mathcal{E}_T$. (b) The spatio-temporal graph that is unrolled over time through the temporal edges in $\mathcal{E}_T$. The edges are labelled with the corresponding feature vectors.

Fig. 2. Architecture of the SRNN in perspective of node u drawn with the spatio-temporal graph.

segments are represented by spatial edges in $\mathcal{E}_S$. Note that our approach differs from [11] in that the spatial edges are established if the two road segments are connected, whereas [11] employs an attention model on a fully-connected graph. Besides, a temporal edge originated from node u is pointing to node u. The spatial graph $(\mathcal{V}, \mathcal{E}_S)$ is unrolled over time using temporal edges in $\mathcal{E}_T$ to form $\mathcal{G}_{\mathcal{ST}}$ as depicted in Fig. 1(b) where the edges are labelled with the corresponding feature vectors.

The feature of node $u \in \mathcal{V}$ at time step t is x_u^t, denoting the traffic speed on the road segment. The feature vector of spatial edge $(u, v) \in \mathcal{E}_S$ at time step t is $\mathbf{x}_{(u,v)}^t = [x_u^t, x_v^t]$, which is obtained by concatenating the features of nodes u and v. As explained above, the spatial edge (u, v) means that the traffic flow comes from node u to node v. If the traffic flow goes in the opposite direction, the edge will be denoted as (v, u) and associated with the different feature vector, $\mathbf{x}_{(v,u)}^t = [x_v^t, x_u^t]$. Note that we employ one spatial edge for each link between two nodes whose feature vector suffices to represent the directionality of the interaction between the nodes, as opposite to our preliminary work [12] using two spatial edges for each link. The feature vector of temporal edge $(u, u) \in \mathcal{E}_T$ at time step t is $\mathbf{x}_{(u,u)}^t = [x_u^{t-1}, x_u^t]$, which is obtained by concatenating the features of node v at the previous time step and the current time step.

C. Model Architecture

In our SRNN architecture, the sets of nodes $\mathcal{V}$, spatial edges $\mathcal{E}_S$, and temporal edges $\mathcal{E}_T$ are associated with RNNs denoted as nodeRNN, $\mathbf{R}_\mathcal{V}$, spatial edgeRNN, $\mathbf{R}_{\mathcal{E}_S}$, and
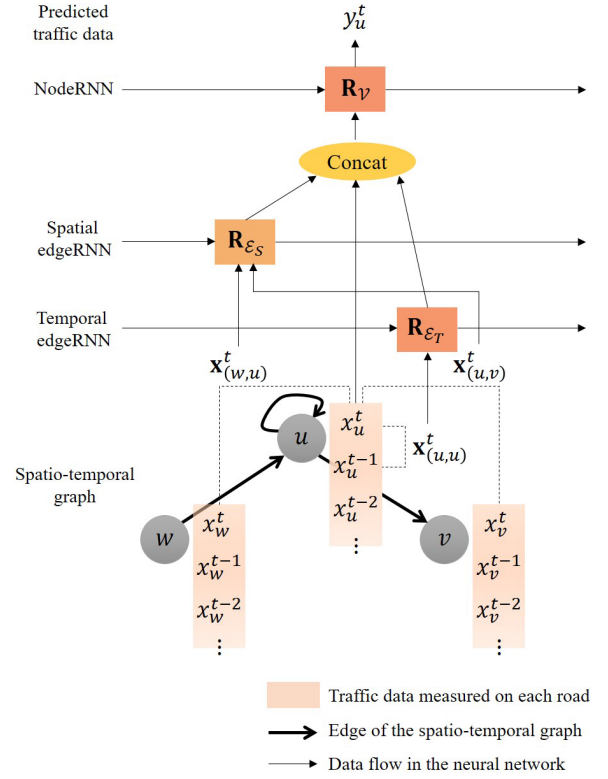
temporal edgeRNN, $\mathbf{R}_{\mathcal{E}_T}$, respectively. The SRNN is derived from the factor graph representation [10]. Our architecture is the simplest case where the nodes, spatial edges, and temporal edges are sharing the same factors, respectively. This means we assume the dynamics of the spatio-temporal interactions is semantically same for all road segments, which keeps the overall parametrization compact and makes the architecture scalable and independent of the road network topology. Readers interested in the factor graph representation can refer to [16].

Fig. 2 visualizes the overall architecture. For each node u, a sequence of node features $\{x_u^t\}_{t=t_c-l+1}^{t_c}$ is fed into the architecture. Every time each node feature enters, the SRNN is supposed to predict the node label y_u^t, which corresponds to the traffic speed at the next time step, x_u^{t+1}. The input into the edgeRNNs is the edge feature $\mathbf{x}_e^t$ of edge $e \in \mathcal{E}_S \cup \mathcal{E}_T$, where the edge is incident to node u in the spatio-temporal graph. The node feature x_u^t is concatenated with the outputs of the edgeRNNs to be fed into the nodeRNN. We use LSTMs for the RNNs.

D. Forward Path Algorithm

In this subsection, we provide a detailed explanation on how our SRNN algorithm works in a forward path. Before we run the forward path, we assure the adjacency matrix, $\mathcal{A}$, describing the road network topology is read to form the set of spatial edges, $\mathcal{E}_S$. The set of spatial edges connected to node u is then obtained as $\mathcal{C}(u) = \{(u, v) \in \mathcal{E}_S \text{ or } (v, u) \in \mathcal{E}_S, \forall v \in \mathcal{V}\}$.

For each time step t, the node feature, x_u^t for $u \in \mathcal{V}$, the spatial edge feature, $\mathbf{x}_e^t$ for $e \in \mathcal{E}_S$, the temporal edge

feature, $\mathbf{x}_e^t$ for $e \in \mathcal{E}_T$, are fed into the architecture. The inputs to the RNNs are converted into fixed-length vectors by an embedding function, denoted as $\phi(\cdot)$. The embedding function applies a linear transformation to the input with a rectified linear unit (ReLU) activation and dropout. For each node u, the following RNNs are executed to predict the node label y_u^t.

1) Spatial EdgeRNN: The spatial edgeRNN, $\mathbf{R}_{\mathcal{E}_S}$, takes the spatial edge features, $\{\mathbf{x}_e^t\}_{e \in \mathcal{E}_S}$, and the hidden state, h_S^{t-1}, as inputs. The hidden state, h_S^{t-1}, has a dimension of $|\mathcal{E}_S| \times \lambda_{\mathcal{E}_S}$ where $\lambda_{\mathcal{E}_S}$ is the size of the spatial edgeRNN. The spatial edge features are fed into a LSTM cell after converted into a fixed-length vector $\mathbf{a}_S^t$ by an embedding function as:

$$\mathbf{a}_S^t = \phi(\{\mathbf{x}_e^t\}_{e \in \mathcal{E}_S}; W_S^E) \tag{1}$$

$$h_S^t = \text{LSTM}(\mathbf{a}_S^t, h_S^{t-1}; W_S^L) \tag{2}$$

where $\phi(\cdot)$ denotes the embedding function, W_S^E denotes the weight associated with the embedding function, and W_S^L denotes the weight associated with the LSTM cell.

2) Temporal EdgeRNN: The temporal edgeRNN, $\mathbf{R}_{\mathcal{E}_T}$, takes the temporal edge features, $\{\mathbf{x}_e^t\}_{e_T \in \mathcal{E}_T}$, and the hidden state, h_T^{t-1}, as inputs. The hidden state, h_T^{t-1}, has a dimension of $|\mathcal{E}_T| \times \lambda_{\mathcal{E}_T}$ where $\lambda_{\mathcal{E}_T}$ is the size of the temporal edgeRNN. Similarly to the spatial edgeRNN, the temporal edge features go through the embedding function and a LSTM cell as:

$$\mathbf{a}_T^t = \phi(\{\mathbf{x}_e^t\}_{e \in \mathcal{E}_T}; W_T^E) \tag{3}$$

$$h_T^t = \text{LSTM}(\mathbf{a}_T^t, h_T^{t-1}; W_T^L) \tag{4}$$

where $\phi(\cdot)$ denotes the embedding function, W_T^E denotes the weight associated with the embedding function, and W_T^L denotes the weight associated with the LSTM cell.

3) NodeRNN: The nodeRNN, $\mathbf{R}_{\mathcal{V}}$, takes the node features, $\{x_u^t\}_{u \in \mathcal{V}}$, the hidden state, h^{t-1}, and the outputs of the spatial edgeRNN and the temporal edgeRNN as inputs.

For each node $u \in \mathcal{V}$, rows of the hidden state of the spatial edgeRNN associated with the spatial edges connected to node u are selected as:

$$h_{\mathcal{C}(u)}^t = h_S^t(\mathcal{C}(u)) \tag{5}$$

where $h_{\mathcal{C}(u)}^t$ has a dimension of $|\mathcal{C}(u)| \times \lambda_{\mathcal{E}_S}$. We add up the row vectors of $h_{\mathcal{C}(u)}^t$ to get the spatial edges' contribution to the nodeRNN as:

$$h_{S_u}^t = \text{sum}(h_{\mathcal{C}(u)}^t) \tag{6}$$

where $h_{S_u}^t$, encapsulating the average spatial influence to node u, has a dimension of $1 \times \lambda_{\mathcal{E}_S}$. We employ summation, rather than concatenation, here to realize a fixed architecture regardless of the graph topology. This is one of key efforts to control the number of parameters by representing the variable context with a fixed-length vector. A row vector of the hidden state of the temporal edgeRNN associated with node u, denoted as $h_{T_u}^t$, is concatenated with $h_{S_u}^t$ to form

$$H_u^t = \text{concat}(h_{T_u}^t, h_{S_u}^t) \tag{7}$$

where H_u^t has a dimension of $1 \times (\lambda_{\mathcal{E}_T} + \lambda_{\mathcal{E}_S})$. Vertically concatenating these row vectors for all nodes in $\mathcal{V}$ provides an input to the nodeRNN, denoted as H^t. Each row of H^t represents the influence of the spatio-temporal edges to the corresponding node feature.

The node features, $\{x_u^t\}_{u \in \mathcal{V}}$, and the concatenated hidden state, H^t, are fed into a LSTM cell after converted into fixed-length vectors by embedding functions as:

$$\mathbf{a}^t = \phi(\{x_u^t\}_{u \in \mathcal{V}}; W^E) \tag{8}$$

$$\mathbf{a}_H^t = \phi(H^t; W_H^E) \tag{9}$$

$$h^t = \text{LSTM}(\text{concat}(\mathbf{a}^t, \mathbf{a}_H^t), h^{t-1}; W^L) \tag{10}$$

$$\{y_u^t\} = W^O h^t \tag{11}$$

where $\phi(\cdot)$ denotes the embedding function, W^E denotes the weight associated with the embedding function for node features, W_H^E denotes the weight associated with the embedding function for the concatenated hidden state, and W^L denotes the weight associated with the LSTM cell. The output hidden state of the nodeRNN is passed through a linear layer with W^O to provide the output node features $\{y_u^t\}$ that represent the predicted labels for all nodes.

E. Training the Structural Recurrent Neural Network

In order to train the SRNN model, the traffic speed measurements for every time step t are used to comprise the edge features, $\{\mathbf{x}_e^t\}_{e \in \mathcal{E}_S}$ and $\{\mathbf{x}_e^t\}_{e \in \mathcal{E}_T}$, and the node features $\{x_u^t\}_{u \in \mathcal{V}}$, as described in Section III-B. The directional adjacency matrix, $\mathcal{A}$, is read to form $\mathcal{C}(u)$ for $u \in \mathcal{V}$.

After the forward path addressed in Section III-D is executed, the prediction error, $\{x_u^{t+1}\} - \{y_u^t\}$, is jointly back-propagated through the nodeRNN, the spatial edgeRNN, and the temporal edgeRNN involved in the forward path. We employ mean squared error (MSE) as a loss function.

Note that the trainable parameters of this SRNN are $\{W_S^E, W_S^L, W_T^E, W_T^L, W^E, W_H^E, W^L, W^O\}$ whose size is independent of the size of the spatio-temporal graph. It only depends on the sizes of the RNNs.

IV. VALIDATION WITH REAL DATA

This section presents the proposed approach and results of performance validation. The purpose of the experiment is to demonstrate: 1) the SRNN outperforms the image-based state-of-the-art methods, CapsNet [6] and CNN [5], in learning the spatially-related time series data; and 2) the SRNN is scalable regardless of the network topology. The detailed methods for preparing the dataset and the neural networks are described in Section IV-A.

A. Implementation Details

In this subsection, we present how datasets are set up and how the different neural networks are implemented for the performance validation.

1) Datasets: We use the traffic speed data that had been gathered in the city of Santander, Spain during the year of 2016, which are available from the case studies of the SETA EU project [13]. Traffic speed measurements taken by magnetic loop detectors on road segments are aggregated every 15 minutes and each sparsely missing measurement is masked with an average of the speed data recorded at the

Road subset 1 (R1)
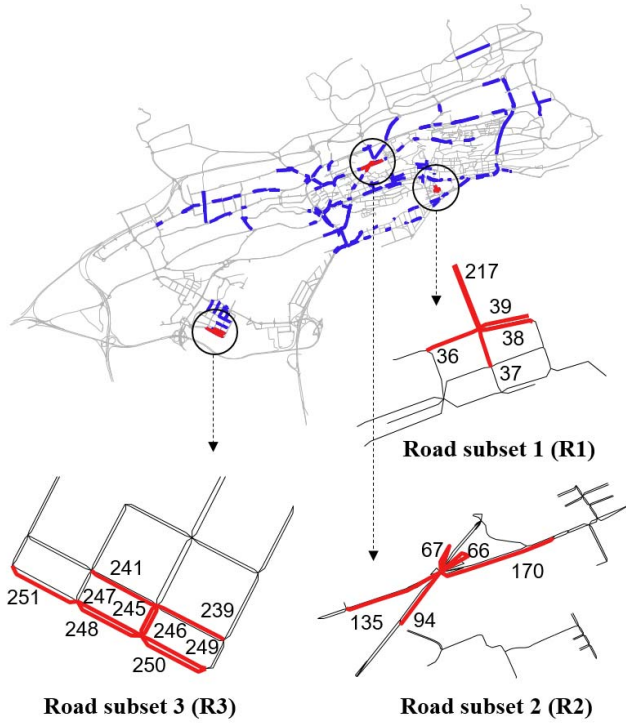
Road subset 3 (R3) **Road subset 2 (R2)**

Fig. 3. Road segments marked in blue and red in the road network are the road segments of interest in which the sensors are installed and the traffic data are available. Road segments marked in red are belong to road subsets used in the performance validation. The road networks around the three road subsets are magnified below. Next to each road segment is a unique ID number for the segment.

(a) Road subset 1 (R1) (b) Road subset 2 (R2)

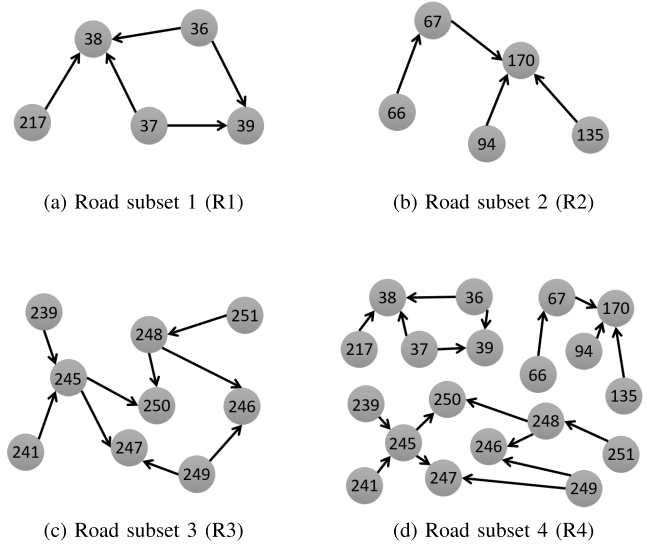(c) Road subset 3 (R3) (d) Road subset 4 (R4)

Fig. 4. Graph representation of four road subsets used in the performance validation. Nodes in the shape of grey circle, with the corresponding ID numbers, denotes road segments. The arrows denote spatial edges that represent directional connection between two adjacent road segments. R4 is the sum of sets R1, R2, and R3. ID numbers for the road segments are presented in the nodes.

same time in the other days. The overall dataset contains 35,040 speed readings per road segment. We use data of the first 9 months as the training set and the remaining data of the last 3 months as the evaluation set. We scale the traffic speed data into the range [0,1] before feeding into the neural networks. In addition, the SRNN uses the adjacency matrix that represents the directional connections between the road network. The adjacency matrix for the road segments with sensors installed, which are of our interest, is extracted from the adjacency matrix for the whole network by taking only the rows and columns associated with the road segments with sensors.

On top of Fig. 3 is the road network of the city of Santander. The blue lines denote the road segments of interest where the traffic data are available. Since such road segments are sparsely located, the road network of interest is a disconnected graph where there is at least one unreachable node, or road segment, starting from a different node in the graph. The road network consists of many small connected graphs whose size varies from 1 to 9. We deal with these connected road subsets because we believe traffic data from the adjacent road segments are correlated, which is referred to as their spatial relationship to be learnt by the SRNN. We use three largest connected sets for validation and denote them as Road subset 1 (R1), Road subset 2 (R2), and Road subset 3 (R3), as shown in Fig. 3. R1 and R2 have 5 road segments and R3 consists of 9 road segments. Next to each road segment is a unique ID number for the segment.

In the experiment, we use traffic speed data and adjacency matrices for 4 road subsets including Road subset 4 (R4) that is the sum of sets R1, R2, and R3. Their graph representation illustrated in Fig. 4 clarifies the spatial connections between the nodes.

2) Neural Networks: We compare the performance of the traffic speed prediction provided by the 3 different neural networks:

- Proposed SRNN
- CapsNet [6]
- CNN [5]

Here we provide a detailed description of how these neural networks are implemented. All the neural networks employ MSE as a loss function and Adam optimizer [17] to minimize the sum of the MSE.

The architecture of the proposed SRNN can be referred to Section III. The SRNN shows its best performance with the following hyperparameters. The hidden state of the nodeRNN has a size of 64, and that of the edgeRNNs has a size of 64 as well. We employ embedding layers in the network that convert the inputs into a 32-dimensional vectors with a dropout rate of 0.5. SRNN is trained with the starting learning rate of 0.0005 and the exponential decay rate of 0.99. The major hyperparameters of the SRNN are summarized in Table I. The SRNN is built based on the Pytorch implementation of [11]. The computational complexity of the SRNN is $\mathcal{O}(W)$ since the SRNN consists of 3 RNNs (a nodeRNN, a spatial edgeRNN, and a temporal edgeRNN executed sequentially) and the computational complexity of a RNN with a LSTM is $\mathcal{O}(W)$ [18] where W is the number of trainable parameters. Note that the number of trainable parameters is independent of the size of the road network as discussed in Section III-E.

The CapsNet and CNN are built based on the Tensorflow implementation of [6]. In summary, they convert the

TABLE I
HYPERPARAMETERS OF THE PROPOSED SRNN

Hyperparameter	Value
NodeRNN size, λ	64
Spatial edgeRNN size, $\lambda_{\mathcal{E}_S}$	64
Temporal edgeRNN size, $\lambda_{\mathcal{E}_T}$	64
Embedding size	32
Learning rate	0.0005
Decay rate	0.99
Dropout rate	0.5

TABLE II
PREDICTION PERFORMANCE IN RMSE (UNIT: km/h)

	R1	R2	R3	R4
SRNN	6.877	8.537	9.419	9.322
CapsNet	9.416	9.931	10.52	9.910
CNN	7.329	9.075	10.54	10.36

spatio-temporal traffic data into an image with two axes representing space and time and apply the deep learning methods for capturing the relationship between the spatio-temporal features in the image. Only one max pooling operation is used in the CNN implementation because the spatio-temporal images from the datasets are small (e.g., 5 segments for R1 and R2). Both of the networks show their best performance with the common starting learning rate of 0.0005 and the exponential decay rate of 0.9999. The computational complexity of the CNN is proportional to the spatial size of the output feature map [19], which is proportional to the number of road segments in our problem, given the number of filters and the filter size are constant. Thus, the CNN has $\mathcal{O}(N)$. On the other hand, the computational complexity of the CapsNet will be proportional to $\mathcal{O}(N^2)$ since it runs routing-by-agreement between capsule features in two convolutional layers [20].

3) Task: Given the datasets, the neural networks are supposed to give 15-min prediction of traffic speed based on 150-min data, which corresponds to a short-term prediction for the next time step based on data of previous 10 time steps ($l = 10$).

4) Performance Metrics: We use the root mean squared error (RMSE) as a performance metric for assessing prediction accuracy as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{I}(y^{(i)} - \hat{y}^{(i)})^2}{I}} \qquad (12)$$

where $\hat{y}^{(i)}$ and $y^{(i)}$ denote the i-th speed prediction and its true value, respectively. Here, I represents the number of the speed data in the evaluation set. Although there are other metrics such as mean relative error (MRE) and mean absolute error (MAE), we believe the RMSE suffices to show the performance difference between the neural networks employing MSE as a loss function. The MRE and MAE show a similar tendency as the RMSE in this experiment [12].

Besides the accuracy, we compare the number of trainable parameters of the neural networks. Since computation time depends on the type of software platform, computing machine, code optimization, and the usage of graphics processing unit (GPU), we see the number of trainable parameters as an empirical measure of the computational complexity. The number of trainable parameters is obtained by counting the number of variables that are optimized by back-propagation, such as the weights and biases in the neural networks.

B. Learning Spatio-Temporal Relationship

Here we verify the SRNN is capable of learning the spatio-temporal relationship accurately, requiring fewer parameters to train. Whereas the RNN learns the time series data on each road as if data on different roads are independent, the SRNN takes the road graph as well as the historical traffic data and learns the spatial relationship between data from different road segments and temporal relationship of time series data on each road.

The three neural networks are trained and evaluated with the datasets R1, R2, R3, and R4. Each dataset contains the 1-year traffic speed data, where the data of the first 9 months constitute the training set and the remaining data of the last 3 months are taken as the evaluation set. We run 10 epochs of training and evaluation and take the average performance as the result. Table II shows the average prediction performance of the methods on each dataset. One can observe that the CNN performs better than the CapsNet for datasets R1 and R2, but worse for datasets R3 and R4. This observation confirms the result reported in [6] that the CapsNet is more effective when the image size (the number of road segments) is larger. Meanwhile, the SRNN outperforms both of the image-based methods. In average, the SRNN is better than the CapsNet and the CNN by 14.1 % and 5.87 %, respectively, in RMSE. Note that the performance is realized with the smaller number of parameters to train. The number of trainable parameter of the SRNN is 8.79×10^4 whereas that of the CapsNet ranges from 5.59×10^5 (R1 and R2) to 7.44×10^6 (R4) and that of the CNN ranges from 3.75×10^5 (R1 and R2) to 4.26×10^5 (R4).

C. Scalability to Network Topology

In order to demonstrate the scalability, we evaluate the three neural networks by using datasets different from the dataset used in training. For example, the neural networks trained with the training set (data of the first 9 months) of R1 are evaluated with the evaluation set (data of the last 3 months) of the other datasets: R2, R3, and R4. This is repeated for evaluations with training sets of R2, R3, and R4, respectively.

Table III summarizes the result. The number in i-th row and j-th column denotes the RMSE obtained with the training set of Ri and the evaluation set of Rj. The diagonal terms are equal to the result in Table II. The SRNN shows consistent performance for all the combinations of training sets and evaluation sets. The evaluation performance tends to depend on the dataset used in evaluation, regardless of the dataset used in training.

On the other hand, the image-based methods are not able to predict traffic data for datasets having the different number of

TABLE III

PREDICTION PERFORMANCE IN RMSE, TRAINED AND EVALUATED WITH
DIFFERENT ROAD NETWORKS (UNIT: km/h)

(a) SRNN

Train \ Eval	R1	R2	R3	R4
R1	6.877	8.697	9.504	9.455
R2	6.990	8.537	9.653	9.444
R3	7.141	9.450	9.419	10.06
R4	6.871	8.714	9.400	9.322

(b) CapsNet

Train \ Eval	R1	R2	R3	R4
R1	9.416	23.46	-	-
R2	21.26	9.931	-	-
R3	-	-	10.52	-
R4	-	-	-	9.910

(c) CNN

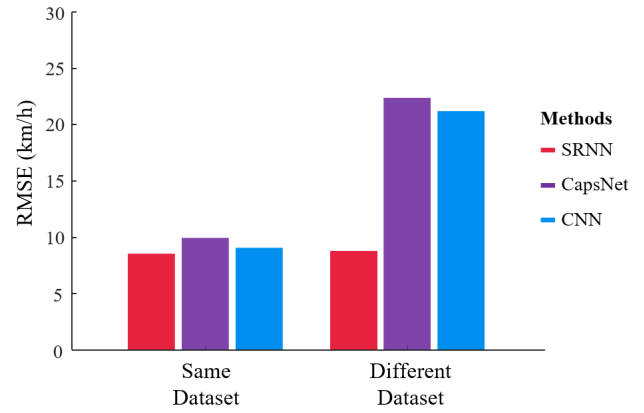| Train \ Eval	R1	R2	R3	R4
R1	7.329	23.57	-	-
R2	18.77	9.075	-	-
R3	-	-	10.54	-
R4	-	-	-	10.36

Fig. 5. Average performance of methods in RMSE, evaluated with the same dataset and the different dataset with the training dataset.

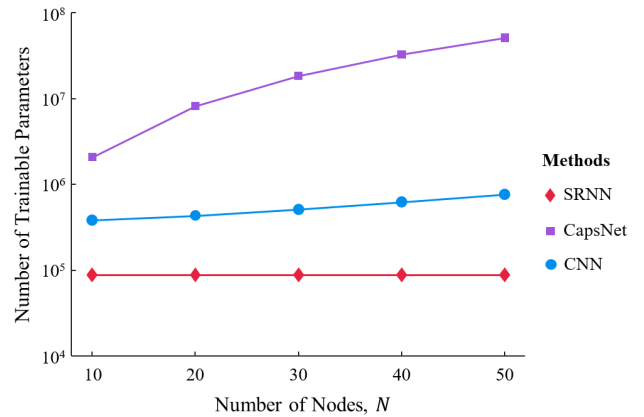Fig. 6. Number of trainable parameters of the SRNN, CapsNet, and CNN.

road segments. Moreover, even if the number of road segments is equal for a training set and an evaluation set extracted from different datasets, it is obvious that the image-based methods will fail to provide accurate predictions because the image-based methods learn the specific features of the spatio-temporal images that are unique for each dataset. Table III shows that the predictions of CapsNet and CNN on different datasets are erroneous. Fig. 5 shows the average RMSEs of the diagonal elements and off-diagonal elements for the three methods in Table III. The SRNN shows similar performance on the same dataset and the different datasets whereas the CapsNet and CNN show a large difference in the performance between the two cases.

We compare the number of trainable parameters in Fig. 6. As the number of nodes, N, increases, the number of trainable parameters of the SRNN remains the same while those of the CapsNet and the CNN keep increasing. When $N = 50$, the number of trainable parameters of the CapsNet is over 100 times larger than that of the SRNN.

Therefore, the SRNN shows better scalability than the image-based methods in terms of the number of road segments and the applicability to other road networks that have the different number of road segments and different topologies.

D. Discussion

As one can observe in Section IV-B and Section IV-C, the proposed SRNN outperforms the image-based methods and is applicable to different datasets with different network topologies. This is contributed by the network design where the dynamics of the spatial and temporal edge features learned by the SRNN can be generalized. The result implies that the SRNN is scalable between disjoint road networks (e.g., R1 and R2 where $R1 \cap R2 = \emptyset$) and also between a subset and a superset (e.g., R1 and R4 where $R1 \subset R4$). Besides, the evaluations on road network R4 verify that the SRNN is able to predict traffic data in a road network that is not a connected set.

Moreover, the number of trainable parameters of the SRNN is independent of the number of the road segments. The size of each RNN, not the number of nodes, determines the number of trainable parameters of the SRNN. The SRNN is able to provide accurate predictions with the smaller number of parameters to train, which will become more advantageous for larger networks.

V. CONCLUSION

In this paper We present an SRNN architecture that learns the spatial and temporal relationship between the traffic

data represented as a spatio-temporal graph. To broaden the applicability, we assume that we are using only historical data and the topological information expressed as an adjacency matrix. With the real traffic speed data measured by induction loop detectors in the city of Santander, we demonstrate that the proposed SRNN is computationally more efficient than image-based state-of-the-art approaches. More importantly, we show that the proposed approach is scalable. When the proposed SRNN model is trained with historic road traffic data, the proposed SRNN model shows reliable and accurate performance also over different road networks. The proposed model can be integrated into the traffic management systems or route planning systems to provide accurate predictions on future traffic states by an efficient and scalable manner.

REFERENCES

[1] 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.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

[2] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. AAAI Conf. Artif. Intell.*, Feb. 2017, pp. 1655–1661.

[3] X. Ma, H. Yu, Y. Wang, and Y. Wang, "Large-scale transportation network congestion evolution prediction using deep learning theory," *PLoS ONE*, vol. 10, no. 3, pp. 1–17, Mar. 2015.

[4] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transp. Res. C, Emerg. Technol.*, vol. 90, pp. 166–180, May 2018.

[5] 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, Apr. 2017.

[6] Y. Kim, P. Wang, Y. Zhu, and L. Mihaylova, "A capsule network for traffic speed prediction in complex road networks," in *Proc. Symp. Sensor Data Fusion, Trends, Solutions, Appl.*, Oct. 2018, pp. 1–6.

[7] Y. Xie, K. Zhao, Y. Sun, and D. Chen, "Gaussian processes for short-term traffic volume forecasting," *Transp. Res. Rec.*, vol. 2165, no. 1, pp. 69–78, Jan. 2010.

[8] J. Chen, K. H. Low, Y. Yao, and P. Jaillet, "Gaussian Process Decentralized Data Fusion and Active Sensing for Spatiotemporal Traffic Modeling and Prediction in Mobility-on-Demand Systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 901–921, Jul. 2015.

[9] P. Wang, Y. Kim, L. Vaci, H. Yang, and L. Mihaylova, "Short-term traffic prediction with vicinity Gaussian process in the presence of missing data," in *Proc. Symp. Sensor Data Fusion, Trends, Solutions, Appl.*, Oct. 2018, pp. 1–6.

[10] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-RNN: Deep learning on Spatio- temporal graphs," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5308–5317.

[11] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," in *Proc. Int. Conf. Robot. Autom.*, May 2018, pp. 1–7.

[12] Y. Kim, P. Wang, and L. Mihaylova, "Structural recurrent neural network for traffic speed prediction," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, May 2019, pp. 5207–5211.

[13] SETA EU Project. (2016). *A Ubiquitous Data and Service Ecosystem for Better Metropolitan Mobility, Horizon 2020 Programme*. Available:. [Online]. Available: http://setamobility.weebly.com/

[14] X. Zheng *et al.*, "Big data for social transportation," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 3, pp. 620–630, Mar. 2015.

[15] W. Brendel and S. Todorovic, "Learning spatiotemporal graphs of human activities," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Nov. 2011, pp. 778–785.

[16] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[17] D. P. Kingma and J. Ba, "Adam: A method for stochasticoptimization," Dec. 2014, *arXiv:1412.6980*. [Online]. Available:https://arxiv.org/abs/1412.6980

[18] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," Feb. 2014, *arXiv:1402.1128*. [Online]. Available: https://arxiv.org/abs/1402.1128

[19] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5353–5360.

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

Youngjoo Kim, photograph and biography not available at the time of publication.

Peng Wang, photograph and biography not available at the time of publication.

Lyudmila Mihaylova, photograph and biography not available at the time of publication.