LC-RNN: A Deep Learning Model for Traffic Speed Prediction

Zhongjian Lv[1], Jiajie Xu[1,2,3*], Kai Zheng[4*], Hongzhi Yin[5*], Pengpeng Zhao[1], Xiaofang Zhou[5,1]
[1] School of Computer Science and Technology, Soochow University, China
[2] Provincial Key Laboratory for Computer Information Processing Technology, Soochow University
[3] State Key Laboratory of Software Architecture (Neusoft Corporation), China
[4] University of Electronic Science and Technology, China
[5] The University of Queensland, Australia
zjlv@stu.suda.edu.cn, {xujj, ppzhao}@suda.edu.cn, zhengkai@uestc.edu.cn,
h.yin1@uq.edu.au, zxf@itee.uq.edu.au

Abstract

Traffic speed prediction is known as an important but challenging problem. In this paper, we propose a novel model, called LC-RNN, to achieve more accurate traffic speed prediction than existing solutions. It takes advantage of both RNN and CNN models by a rational integration of them, so as to learn more meaningful time-series patterns that can adapt to the traffic dynamics of surrounding areas. Furthermore, since traffic evolution is restricted by the underlying road network, a network embedded convolution structure is proposed to capture topology aware features. The fusion with other information, including periodicity and context factors, is also considered to further improve accuracy. Extensive experiments on two real datasets demonstrate that our proposed LC-RNN outperforms seven well-known existing methods.

1 Introduction

Traffic speed prediction (TSP) means to predict the future speed of each road segment based on historical observations. It has been recognized a vital technique for many traffic related applications, e.g. prospective traffic navigation to avoid potential jam in advance [Yuan *et al.*, 2010], just-in-time departure recommendation in trip schedule [Khetarpaul *et al.*, 2013], and the construction of intelligent transportation systems [Leontiadis *et al.*, 2011], etc. In recent years, trajectory data have been accumulated to an extremely large volume in enterprises. These data contain rich traffic information, which enables us to understand traffic dynamics in greater detail, and thus make accurate TSP possible. In this paper, we aim to investigate the problem of TSP using trajectory data.

For each road segment, the fluctuation of its speed usually follows some temporal patterns. For example, the speed in downtown area is relatively low during rush hours. Once a congestion occurs, it tends to affect the speed of that road in the near future. These patterns can be discovered by existing supervised learning models and time-series models, such as support vector regression (SVR) [Wu *et al.*, 2003] and hybrid auto-regressive integrated moving average (H-ARIMA) [Pan *et al.*, 2012]. Recently, recurrent neural network (RNN) has achieved great success in the problem of sequence learning [Sutskever *et al.*, 2014]. As a result, TSP can be naturally processed by RNN, and a solution based on long short-term memory (LSTM), a variant of RNN, has been proposed in [Ma *et al.*, 2015a]. However, this is insufficient for TSP problem, since the speed of different road segments have strong dependency with each other.

Therefore, the TSP model must deliberately incorporate spatial correlations among different road segments. Spatial correlation means that a congestion will affect surrounding areas. Conversely, speed on a road tends to slow down if congestions occur in its surrounding area. To this end, the convolutional neural network (CNN) has shown powerful ability to model similarity between pixels, as a kind of spatial relations, in image processing [Krizhevsky *et al.*, 2012]. This motivates some recent studies [Zhang *et al.*, 2017; Ma *et al.*, 2017; Wang *et al.*, 2017] to adopt CNN in modeling how traffic evolves in spatial. They have achieved significant success, even though the predictions are spatial cell based, or restricted by ring road only.

In despite of the advances of deep learning, previous methods have not fulfill their potential due to the following reasons. First, spatial correlation of traffic is strongly restricted by the underlying road network, but the integration between its topology and CNN model is untouched. This explains why these methods offer coarse-granularity speed prediction only. Secondly, accuracy can be further improved by taking advantage of different models, e.g. RNN is good at learning time series patterns with long time span, while CNN has superior performance in spatial correlation understanding. We thus aim to combine them together for improved time-series prediction using features related to surrounding areas. Last

* Corresponding author

but not the least, some context information like weather/peak periods/holiday will affect traffic speed, and periodical patterns for some roads in days/weeks also can help us predict future. Accordingly, extraction and fusion with these information play an important part too. It is thus crucial to design an improved model with all above aspects to be handled.

To address above issues, we propose a more effective model, called Look-up Convolution Recurrent Neural Network (LC-RNN), to support accurate traffic speed prediction in road segment granularity. The main contributions of this paper can be summarized as follows:

1. LC-RNN adopts a road network embedded convolution method to learn more meaningful spatial features. Based on a set of topology aware look-up operations, the convolution layer is able to capture complex traffic evolution patterns restricted by the underlying road network.
2. LC-RNN seamlessly integrates RNN(LSTM/GRU) and CNN in a rational way to make use of their advantages. The basic idea is to feed the features of surrounding areas extracted by CNN to RNN for learning time-series patterns, so as to achieve more accurate speed prediction by referencing neighboring area dynamics.
3. By taking periodicity and context factors into consideration, LC-RNN adaptively fuse the result with these information by a learnable parameter matrix to obtain a more accurate prediction.
4. We evaluate our model on a large road network using Beijing taxi trajectories and a small road network in Shanghai. The results demonstrate the advantages of our approach compared with seven benchmarks.

2 Problem Formulation

In this section, we briefly give out the formulation of network-based traffic speed prediction problem.

We denote a road network as a directed graph $G = (V, E)$, where each vertex $v \in V$ denotes an intersection or the segmentation point of a road, and each edge $r \in E$ is a road segment. We use $r.s$ and $r.e$ to represent the start and end vertexes of r. Figure 1(a) shows an abstract road network.

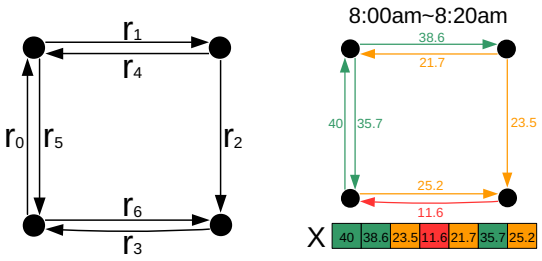
Given a time interval t, we use $x_t^{r_i}$ to represent the average traffic speed of a road segment $r_i \in E$. Towards the whole road network, we use a *speed vector*, denoted by $X_t = [x_t^{r_0}, x_t^{r_1}, ..., x_t^{r_{|E|-1}}]$, to represent the speed information of all road segments at the time interval t. Figure 1(b) shows a speed vector for the interval 8:00-8:20am.

Problem: Given the historical observations $\{X_i | i = 1, ..., t\}$, this paper aims to predict $Y_t = \{X_j | j = t+1, ..., t+z\}$, where z is the number of time intervals to be predicted.

3 Model Description

3.1 The Overview of LC-RNN

Figure 2 presents the architecture of LC-RNN, which is comprised of several look-up convolution layers, a recurrent layer, a periodicity extraction layer and a context extraction layer, modeling local spatial evolution, long temporal dependency, periodicity and context factors respectively.



(a) A road network (b) A speed vector at 8:00am - 8:20am

Figure 1: Road network and speed vector

For learning time-series patterns based on neighboring traffic evolution patterns, we take the speed vectors of intervals in the recent time and the topology of road network as inputs. By a set of topology aware convolution operations, where the topology of road network is embedded into convolution, look-up convolution (LC) can effectively capture the spatial traffic dynamics of the surrounding areas. Since the speed of one road will be affected by traffic condition from more distance areas, a stack of LC layers is used to understand such more distance spatial evolution. Meanwhile, we attempt Batch Normalization (BN) [Ioffe and Szegedy, 2015] after LC layer for faster training speed. After getting the complex traffic evolution patterns, we reshape these features in the way of time sequence to feed into RNN layer. As a varient of RNN, LSTM is good at learning long time-series patterns, so we predict speed of each road by using it, and concatenate all speeds to get Y_{ST}.

In addition to the explore of the spatio-temporal trend, we extract the other information including periodicity and context factors. Periodicity means that the current speed on certain road will be almost the same as days/weeks ago. Hence, we feed the speed vectors of corresponding time intervals several days/weeks ago into fully-connected (FC) layers, learning the daily/weekly periodicity to obtain Y_P. Concurrently, we extract context factors such as weather, holiday and so on by another two-layer FC neural network. The output of context extraction layer Y_C is integrated with Y_P to achieve Y_E. At last, we use a parameter-matrix-based method to fuse Y_{ST}

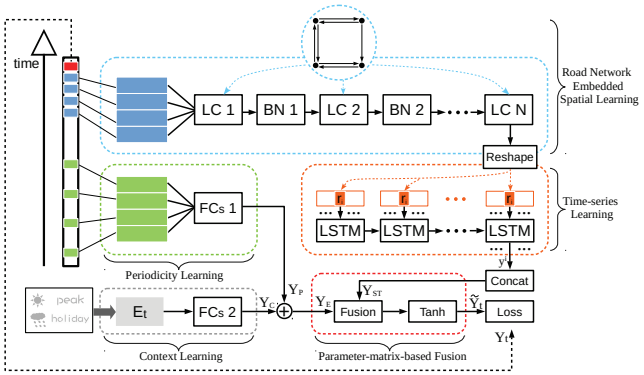


Figure 2: The architecture of LC-RNN. LC:Look-up Convolution; BN:Batch Normalization; FC:Fully-connected; LSTM:Long short-term memory

and Y_E. These techniques can give a significant boost to the accuracy of traffic speed prediction.

3.2 Look-up Convolution Layer

Traditional convolution neural network has been used in many fields such as image and textual data analysis [Lecun *et al.*, 2015]. It has shown good performance on capturing spatial features from adjacent pixels/grids of a tensor. However, since traffic evolution is restricted by the road network, the speed of a road is impacted by road segments adjacent in road network, but nonadjacent in the speed vectors. Embedded graph struture is an effective method for many applications such as [Wang *et al.*, 2016b]. Therefore, topology must be embedded into convolution for learning spatial correlations with road network constraints, which can be achieved by the look-up operations proposed in [Wu *et al.*, 2017]. We thus design a look-up convolution layer that embed the topology of road network into convolution to capture more meaning spatial features.

We use a structure called $adjacent\ road\ matrix$, denoted by M, to represent the road network topology. It records all adjacent roads for each road $r \in E$ by S_r. Concretely, $S_r = \{r, r' \in E \mid r'.s = r.e\ or\ r'.e = r.s\}$, denoting a set of roads directly connected to r and r itself. For all roads, we use $\mathbb{S}$ to denote all sets, namely $\mathbb{S} = \{S_r | r \in E\}$. Then M can be constructed by $\mathbb{S}$ which has the dimension of $A \times |E|$, where $A = max\{|S_r|\ |S_r \in \mathbb{S}\}$. Here, $M[:, i]$ records all roads in the S_{r_i}. For example, $M[:, 3] = [0, 2, 3, 6]^T$ in figure 1(a). Since the adjacent segment sets for different roads have different sizes, we pad each column with the road itself to A.

After expressing the road network topology by matrix M, we need to embed it into convolution to extract the information of adjacent roads. To this end, look-up operation is combined with convolution as shown in figure 3. For arbitrary l-th layer, it takes M and the previous output X^{l-1} as input. It is noted that the input of first layer is comprised of the previous p and current speed vectors (in figure 2), namely $X^0 = [X_t, X_{t-1}, ..., X_{t-p}]^T$. At l-th layer, we use k_l filters to convolve and concatenate all matrices to get X^l. The k-th matrix convolved by the k-th filter can be formulated as follows

$$X^{l,k} = [\mathbf{x}_0^{l,k}, ..., \mathbf{x}_i^{l,k}, ..., \mathbf{x}_{|E|-1}^{l,k}] \tag{1}$$

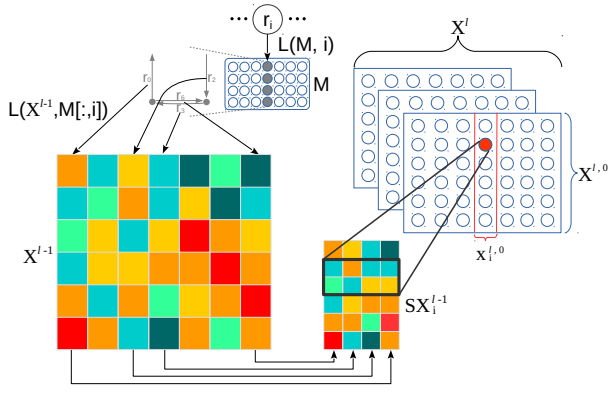


Figure 3: Look-up convolution at l layer

$$\mathbf{x}_i^{l,k} = relu(L(X^{l-1}, L(M, i)) * W^{l,k} + b^{l,k}) \tag{2}$$

where $L(P, Q) = P[:, Q]$ denotes a look-up operation. It returns a tensor by looking up the second dimension of P according to the elements in Q (as index) as well as puting these slices together. Hence, $L(M, i)$ returns a vector $M[:, i]$ in our model. Based on this vector, we further get a sub-matrix SX_i^{l-1} by look-up operation $L(X_{l-1}, M[:, i])$.

Here, $*$ denotes the convolution operation which uses the k-th filter $W^{l,k}$ being a $h \times A$ matrix to zigzag scan the sub-matrix and the result is a vector which can be denoted as

$$\mathbf{x}_i^{l,k} = [x_{0,i}^{l,k}, ..., x_{j,i}^{l,k}, ..., x_{p,i}^{l,k}]^T \tag{3}$$

$$x_{j,i}^{l,k} = relu(\sum_{m=0}^{h}\sum_{n=0}^{A} w_{m,n}^{l,k} sx_{j+m,n}^{l-1} + b^{l,k}) \tag{4}$$

where $b^{l,k}$ is a bias for the k-th filter, $w_{m,n}^{l,k}$ is the (m, n) element of $W^{l,k}$, $sx_{j+m,n}^{l-1}$ is the $(m, j+n)$ element of the sub-matrix SX_i^{l-1}. More details about the implementation of convolution layer could be found in [Bouvrie, 2006]. Here we use rectifier linear unit ($relu$) [Krizhevsky *et al.*, 2012] as activation function, namely $relu(x) = max(0, x)$, for reducing the problem of gradient vanishing.

In the process of matrix computation, the look-up operation will be carried out for all roads at the same time which means that it will not cost too much time. In addition, the size h of filter is set a small number such as one or two, where $h = 1$ means we just explore the local spatial evolution at the same time and $h = 2$ means we explore the evolution at the nearby time intervals. In fact, We are going to use these two kinds of filter at a layer for obtaining more descriptive and diversified spatial characteristics.

3.3 Recurrent Layer

On top of the look-up convolution, we further use the RNN model to learn the long-term temporal patterns that can reference surrounding area traffic dynamics. After N look-up convolution layers, the last output tensor $X^N \in \mathbb{R}^{|p+1| \times |E| \times |k_N|}$ is thus feed to RNN, where k_N is the number of the convolution filters at the last LC layer. Before using recurrent layer, we need to transform X^N to the spatial evolution time sequence $[V_S^{i,0}, ..., V_S^{i,t}, ..., V_S^{i,p}]$ for each road r_i. Each vector $V_S^{i,t}$ can be obtained by

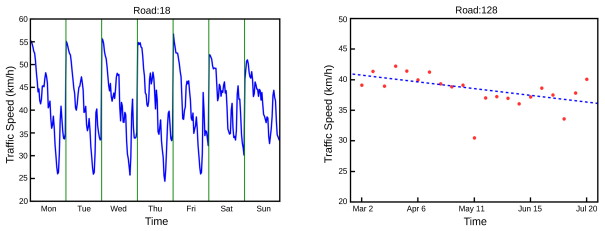
$$V_S^{i,t} = X^N[t, i, :] \tag{5}$$

Next, we can feed the sequence to the RNN layer. According to [Bengio *et al.*, 2002], the standard RNN is inferior at modeling long-term sequence information. On the other hand, it is hard to train due to the problem of vanishing gradient. Fortunately, this drawback can be handled by LSTM cell whose details can be found in [Graves, 2012]. Adopting LSTM, we can get the hidden state sequence $[h_0, ..., h_t, ..., h_p]$ generated iteratively by the following equations

$$h_t = LSTM(V_S^{i,t}, h_{t-1}) \tag{6}$$

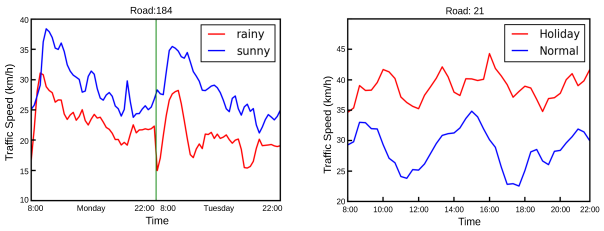
After getting the last hidden state h_p, the output can be computed by

$$y^{i,t+j} = \phi(U_y^j h_k + b_y^j) \tag{7}$$



(a) Daily Period (b) Weekly Trend

Figure 4: Periodicity Information



(a) Weather (b) Holiday

Figure 5: Context Information

where we use z units with learnable parameters to predict z traffic speeds, thus $y^i = [y^{i,t+1}, y^{i,t+2}, ..., y^{i,t+z}]$ can be got. ϕ is $relu$ activation if we fuse other information, otherwise, $tanh$. At last, we concatenate the predict values of all road segments to obtain $Y_{ST} = [y^0, y^1, ..., y^{|E|-1}]^T$.

3.4 Periodicity and Context Extraction Layer

For obtaining a more accurate prediction, we can extract other information including periodicity and context information.

Periodicity Information Extraction. Traffic speed usually changes periodically which means that the traffic speed for an road segment at a certain time interval is similar with the same time interval of the previous day or previous week in the absence of special condition. In general, daily periodicity and weekly trend are two main kinds of periodicity information. Figure 4(a) depicts the average traffic speed of a main road segment from 8:00am to 22:00pm in one week. We can see the obvious daily periodicity, especially at weekdays. Figure 4(b) describes the traffic speed at a certain time interval (17:00pm-17:20pm) of Wednesday from March 2016 and July 2016. As time goes by, the traffic speed decreases slightly. We also find the same properties in part of other road segments.

For extracting the periodicity information, we can get the daily periodicity and weekly trend information for each road segments at the time intervals to be predicted. For each road, a FC layer is used for extracting the average speed of several days ago and another FC layer is used to extract the trend of weekly changes. Then we integrate the two results and concatenate all to get Y_P that has the same shape with Y_{ST}.

Context Information Extraction. In addition to the periodicity information, context factors is another important information to extract such as weather and holiday event. Figure 5(a) shows that the driving in the heavy rain (Jun 13-14) is slower than that in sunny days (May 30-31) because of safe driving. On the other hand, figure 5(b) shows that the traffic speed during holidays (Qingming Festival) can be different from during normal days (the previous week of Qingming Festival).

In our experiment, we mainly consider holiday, weather and metadata (i.e. Weekday/Weekend, DayOfWeek, HourOfDay). To predict future traffic speeds, the holiday and metadata can be directly obtained, however, the weather is unknown. Instead, we can use forecasting weather information. Formally, feature learning algorithm [Wang *et al.*, 2016c] or a fully connected layer is adopted to extract context information and another fully connected layer is used to map low to high dimensions to get Y_C. Then we integrate it with Y_P from periodicity learning to obtain Y_E.

3.5 Fusion

Usually, the predicted speed is closer with the current time, the result is more accurate by spatio-temporal learning. And the faraway speed need to be corrected more by periodicity. On the other hand, different context factors affect speed in different degrees. Hence, we decide to use a parameter-matrix-based fusion matrix to fuse the two results by

$$\widetilde{Y}_t = tanh(W_{ST} \circ Y_{ST} + W_E \circ Y_E) \tag{8}$$

where $\circ$ is element-wise multiplication, W_{ST} and W_E are the learnable parameters.

Our LC-RNN can be trained via backpropagation to predict Y_t by minimizing mean squared error between the predicted speed vectors and the true vectors:

$$L(\theta) = ||Y_t - \widetilde{Y}_t||_2^2 \tag{9}$$

where θ are all learnable parameters in the LC-RNN.

4 Experiments

In this section, we mainly conducted experiments on two size road networks in Beijing and Shanghai respectively to evaluate the effectiveness of LC-RNN.

4.1 Data Preparation

In our experiment, we used two datasets from Beijing and Shanghai to test our LC-RNN model detailed as follows.

- **Beijing**: The trajectory data and context factors was collected from 1st Mar. to 31st Jul. in 2016. We mainly extracted a graph more than 10 thousand main road segments. There are more than 2 million trajectories covering the road network every day. The weather condition can be divided into 10 types (e.g. Sunny, Rainy), temperature ranges from -4 °C to 36 °C and wind speed is divided into 4 levels. There are about 12 important holidays and 24 weekends among the dataset. We use the previous 10 time intervals as input, namely $p = 10$. Furthermore we use the corresponding intervals of previous 5 days and 3 weeks to extract periodicity information. The data of the first 4 months were used as the training set, and the remaining 1 month as the test set.
- **Shanghai**: Trajectory data was collected from 1st Mar. to 31st Apr. in 2015. We just extracted a small road network about 1.5 thousand main roads. The weather condition is divided into 6 types and there are 8 weekends.

We use the previous 10 intervals and corresponding intervals of previous 5 days. Among the data, the last 15 days are test set and the others are training set.

In the preprocessing stage, we calculate the mean velocity of each road at each time interval. Data sparsity is also a challenging problem [Yin *et al.*, 2016], we use the data from 6:00am to 22:00pm and infer the missing value by spatial-temporal filling method [Yi *et al.*, 2016] to alleviate it. In the output of the LC-RNN, we use $tanh$ as our final activation, whose range is between -1 and 1. Hence, we use the Min-Max Normalization method to scale the traffic speed into [-1,1]. In the evaluation, we re-scale the predicted value back to the normal values. We train our network with the following hyper-parameters setting: mini-batch size (48), learning rate (0.0002) with adam optimizer, $1 \times A$ filters (32) and $2 \times A$ filters (16) in each LC layer. We select 90% of the training data for training model, and the remaining 10% is chosen as the validation set with 3 early stopping.

4.2 Benchmarks

We compare our model with the following baseline methods.

- **SVR** [Wu *et al.*, 2003]: Support Vector Regression (SVR) is an powerful regression method which has greater generalization ability.
- **H-ARIMA** [Pan *et al.*, 2012]: H-ARIMA is a method combined of ARIMA and HA to predict future values.
- **SAE** [Lv *et al.*, 2015]: Stacked Auto Encoders (SAE) is a deep learning model to learn generic traffic features and predict future values.
- **LSTM** [Ma *et al.*, 2015a]: As a representative of RNN, LSTM can effectively handle long temporal dependence and reduce gradient vanishing which is appropriate for speed prediction.
- **GC** [Defferrard *et al.*, 2016]: Graph Convolution (GC), pooling and fully-connected is used to forecast future speed.
- **DCNN** [Ma *et al.*, 2017]: Deep Convolution Neural Network (DCNN) with convolution, pooling and fully-connected is used for speed prediction.
- **ST-ResNet** [Zhang *et al.*, 2017]: Spatio-Temporal Residual Network (ST-ResNet) uses residual network to model three temporal properties to do prediction.

4.3 Performance Comparisons

We measure our methods and other benchmarks by Root Mean Square Error (RMSE).

Results on Beijing: We first give the comparison with 7 other benchmarks on the Beijing dataset, as shown in Table 1, where time interval is 10min and the number of intervals to be predicted z is 1. Meanwhile, we compare all 7 variants of LC-RNN with different layers and components. Taking LC-3-RNN-E-BN for example, it uses 3 lookup convolution layers with BN layers, recurrent layer and fuses with external factors. We observe that all of these 7 models are better than 7 benchmarks. And we can find that **LC-3-RNN-E-BN** reduces error to 5.274, which significantly improves accuracy.

Then we analyze the effects of different components:

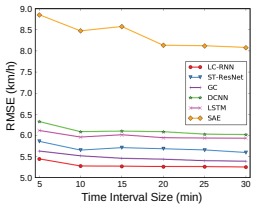
Methods	RMSE
SVR	10.245
H-ARIMA	11.867
SAE	8.471
LSTM	5.958
DCNN	6.085
GC	5.514
ST-ResNet	5.749
LC-3	5.437
LC-3-RNN	5.328
LC-3-RNN-E	5.296
LC-3-RNN-E-noFusion	5.392
LC-3-RNN-E-BN	**5.274**
LC-2-RNN-E-BN	5.319
LC-4-RNN-E-BN	5.285

Table 1: Comparison among different methods on Beijing dataset

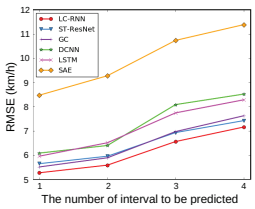
- The structure of look-up convolution layer: Results of LC-3, DCNN and ST-ResNet shows that look-up convolution achieves a good result compared with general convolution. That is to say, the topology of road network can be captured by our look-up convolution operation. On the other hand, using 3 LC layers obtains the best result, compared with 2/4 LC layers, because it can get a high-level features with suitable parameters. At last, we observe that BN between LC layers gains a little improvement which demonstrate the effectiveness of it.
- The structure of recurrent layer: Taking temporal dependency into consideration, LC-3-RNN feed the features from local spatial evolution into recurrent layer (LSTM in our experiment) and LC-3 do not. The result indicates that LC-3-RNN is further promoted which demonstrates the effectiveness of recurrent layer.
- Periodicity and context extraction layer: LC-3-RNN-E considers the other information including periodicity information and context factors. If not, the model is degraded as LC-3-RNN. It shows that the result of LC-3-RNN-E is better than LC-3-RNN, pointing out that external extraction layer is beneficial for speed prediction.
- Fusion: Different from LC-3-RNN-E, LC-3-RNN-E-noFusion use a straight-forward method, $i.e. Y_{ST} + Y_E$, instead of parameter-matrix-based fusion (Eq.8). It shows the error greatly increases, demonstrating the effectiveness of parameter-matrix fusion method.

In addition, we predict the traffic speed of the road network with the size of time interval varying from 5 to 30 minutes and the number of intervals to be predicted varying from 1 to 4. Here time interval size with 10 minutes and the number of predicted intervals with 1 are default parameter. Figure 6 shows the comparative performances for LC-RNN and the benchmarks since 2015. We give the analysis about the two scenarios.

- Varying time interval size: In figure 6(a), we can find that the performance of LC-RNN is always better than the other benchmarks under any time interval size setting. And we can see that the prediction performance becomes a little better when the length of time interval



(a) Varying the size of time interval



(b) Varying the number of intervals to be predicted

Figure 6: Prediction performance on Beijing dataset

increases. Intuitively, this may be due to the fact that the traffic speed becomes smoother with increasing the size of time interval. The result indicates that our model can effectively predict the traffic speed on the road network.

- Varying the number of time intervals to be predicted: As shown in figure 6(b), LC-RNN achieves the best performance when varying the quantity of intervals needing to be predicted. Note that the performance becomes worse with the quantity increases since the correlation of traffic speed between time intervals being predicted and current moment decreases.

In summary, from the above experimental result, we can find that LC-RNN achieves the best performance compared with the state-of-the-arts on the Beijing. The fusion of spatial evolution on the road network, temporal dependency and other information are the key to success.

Results on Shanghai: Table 2 shows the results of our model and other benchmarks on Shanghai dataset. Here, we adopt LC-3-RNN-E-BN which achieves best performance on the Beijing. We can find our model has relatively from 4.8% up to 48.4% lower RMSE than these benchmarks on the Shanghai, demonstrating that our proposed model has good generalization performance on the road network with different scale. In addition, the result about varying the size of time interval and the number of predicted intervals is also similar with one on the Beijing and we can find conclusion among the previous analysis.

5 Related Work

5.1 Traffic Speed Prediction

Traffic speed prediction is an important problem that has been intensively studied in the last few decades. Classical methods include ARIMA and its variations including seasonal-ARIMA [Zhang *et al.*, 2005] and H-ARIMA [Pan *et al.*, 2012]. Some supervised learning methods, such as SVR [Wu *et al.*, 2003] and LR [Ristanoski *et al.*, 2013], are used to deal TSP as a regression problem for each road segments individually. The boltzmann machine [Ma *et al.*, 2015b] and bayesian networks [Wang *et al.*, 2016a] models can be used to portray spatial correlation, though they cannot be applied for TSP directly due to the nature of probability model. More recently, more advanced solutions are proposed to capture traffic evolution in spatial using deep learning models. [Wang *et al.*, 2017] proposed a CNN based model with error feedback to deal with TSP on the ring road (only), and [Ma *et al.*, 2017] use a deep CNN model for a similar problem of traffic flow prediction. Above two models have shown good performance, though they fail to consider the topology structure of road network, and the results can be further improved if different deep learning models can be integrated.

Methods	RMSE
SVR	9.083
H-ARIMA	9.317
SAE	7.421
LSTM	5.274
DCNN	5.269
GC	4.921
ST-ResNet	5.087
LC-3-RNN-E-BN [ours]	**4.686**

Table 2: Comparison among different methods on Shanghai dataset

5.2 Deep Learning

Deep learning techniques have been successfully applied to various applications [Sutskever *et al.*, 2014; Yin *et al.*, 2017]. As one of famous techniques, RNN achieves great success in sequence learning tasks. The incorporation of LSTM [Graves, 2012] enables RNN to learn long-term sequence information. Traffic speed of each road following some long-term temporal partterns can be naturally solved by LSTM [Ma *et al.*, 2015a]. Another well-known technique called CNN is good at capturing spatial dependency. It is widely used in the field of computer vision [Lecun *et al.*, 2015] and many other applications including traffic management [Zhang *et al.*, 2017; Wang *et al.*, 2017]. [Zhang *et al.*, 2017] applies CNN to deal with grid-based crowd flow prediction and [Wang *et al.*, 2017] predicts traffic speed on the ring road. Embedding is a novel technology frequently used in the field of natural language processing [Mnih and Kavukcuoglu, 2013] to extract latent features of words. In modeling trajectory, the look-up operations of embedding are adopted in [Wu *et al.*, 2017] to represent the road network constraints. Graph Convolution [Defferrard *et al.*, 2016] is a spectral approach that ensures strictly localized filter and low computational complexity as well.

Compared to existing TSP solutions, our model takes advantages of both RNN and CNN by an integration of them with look-up operations, so that more meaningful time-series patterns adaptive to surrounding area dynamics can be learned to improve accuracy.

6 Conclusion

In this paper, we propose a novel deep learning based model LC-RNN for forecasting traffic speed. Our model not only seamlessly integrates CNN and RNN to learn speed fluctuation patterns that can reference surrounding area dynamics, but also adopts a road network embedded convolution to represent the constraints of the underlying road network. We further improve accuracy by adaptively fusing with other information including periodicity and context factors. The experimental results on two real datasets demonstrate the effectiveness of LC-RNN model for the TSP problem.

Acknowledgments

The authors would like to thank the anonymous reviewers for their critical and constructive comments and suggestions. This work was supported by the National Natural Science Foundation of China under Grant Nos. 61402312, 61572335, 61502324, 61472263, 61532018, and 61772356, the Natural Science Foundation of Jiangsu Province of China under Grant No. BK20151223 and the Open Program of State Key Laboratory of Software Architecture under item number SKLSAOP1801.

References

[Bengio *et al.*, 2002] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 2002.

[Bouvrie, 2006] Jake Bouvrie. Notes on convolutional neural networks. *Neural Nets*, 2006.

[Defferrard *et al.*, 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, pages 3837–3845, 2016.

[Graves, 2012] Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer Berlin Heidelberg, 2012.

[Ioffe and Szegedy, 2015] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.

[Khetarpaul *et al.*, 2013] Sonia Khetarpaul, S. K. Gupta, and L. Venkata Subramaniam. Analyzing travel patterns for scheduling in a dynamic environment. In *CD-ARES*, pages 304–318, 2013.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

[Lecun *et al.*, 2015] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[Leontiadis *et al.*, 2011] Ilias Leontiadis, Gustavo Marfia, David Mack, Giovanni Pau, Cecilia Mascolo, and Mario Gerla. On the effectiveness of an opportunistic traffic management system for vehicular networks. *IEEE Trans. Intelligent Transportation Systems*, 12(4):1537–1548, 2011.

[Lv *et al.*, 2015] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei Yue Wang. Traffic flow prediction with big data: A deep learning approach. *IEEE Trans. Intelligent Transportation Systems*, 16(2):865–873, 2015.

[Ma *et al.*, 2015a] Xiaolei Ma, Zhimin Tao, Yinhai Wang, Haiyang Yu, and Yunpeng Wang. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C Emerging Technologies*, 54:187–197, 2015.

[Ma *et al.*, 2015b] Xiaolei Ma, Haiyang Yu, Yunpeng Wang, and Yinhai Wang. Large-scale transportation network congestion evolution prediction using deep learning theory. *Plos One*, 10(3):e0119044, 2015.

[Ma *et al.*, 2017] Xiaolei Ma, Zhuang Dai, Zhengbing He, Jihui Ma, Yong Wang, and Yunpeng Wang. Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. *Sensors*, 17(4):818, 2017.

[Mnih and Kavukcuoglu, 2013] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *NIPS*, pages 2265–2273, 2013.

[Pan *et al.*, 2012] Bei Pan, Ugur Demiryurek, and Cyrus Shahabi. Utilizing real-world transportation data for accurate traffic prediction. In *ICDM*, pages 595–604, 2012.

[Ristanoski *et al.*, 2013] Goce Ristanoski, Wei Liu, and James Bailey. Time series forecasting using distribution enhanced linear regression. In *PAKDD*, pages 484–495, 2013.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[Wang *et al.*, 2016a] Dong Wang, Wei Cao, Mengwen Xu, and Jian Li. ETCPS: an effective and scalable traffic condition prediction system. In *DASFAA*, pages 419–436, 2016.

[Wang *et al.*, 2016b] Sen Wang, Xiaojun Chang, Xue Li, Guodong Long, Lina Yao, and Quan Z. Sheng. Diagnosis code assignment using sparsity-based disease correlation embedding. *TKDE*, 28(12):3191–3202, 2016.

[Wang *et al.*, 2016c] Sen Wang, Feiping Nie, Xiaojun Chang, Xue Li, Quan Z. Sheng, and Lina Yao. Uncovering locally discriminative structure for feature analysis. In *ECML/PKDD*, pages 281–295, 2016.

[Wang *et al.*, 2017] Jingyuan Wang, Qian Gu, Junjie Wu, Guannan Liu, and Zhang Xiong. Traffic speed prediction and congestion source exploration: A deep learning method. In *ICDM*, pages 499–508, 2017.

[Wu *et al.*, 2003] Chun Hsin Wu, Chia Chen Wei, Da Chun Su, and Ming Hua Chang. Travel time prediction with support vector regression. In *IEEE Trans. Intelligent Transportation Systems*, volume 5, pages 1438–1442 vol.2, 2003.

[Wu *et al.*, 2017] Hao Wu, Ziyang Chen, Weiwei Sun, Baihua Zheng, and Wei Wang. Modeling trajectories with recurrent neural networks. In *IJCAI*, pages 3083–3090, 2017.

[Yi *et al.*, 2016] Xiuwen Yi, Yu Zheng, Junbo Zhang, and Tianrui Li. ST-MVL: filling missing values in geo-sensory time series data. In *IJCAI*, pages 2704–2710, 2016.

[Yin *et al.*, 2016] Hongzhi Yin, Xiaofang Zhou, Bin Cui, Hao Wang, Kai Zheng, and Nguyen Quoc Viet Hung. Adapting to user interest drift for POI recommendation. *TKDE*, 28(10):2566–2581, 2016.

[Yin *et al.*, 2017] Hongzhi Yin, Weiqing Wang, Hao Wang, Ling Chen, and Xiaofang Zhou. Spatial-aware hierarchical collaborative deep learning for POI recommendation. *TKDE*, 29(11):2537–2551, 2017.

[Yuan *et al.*, 2010] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. T-drive: driving directions based on taxi trajectories. In *GIS*, pages 99–108, 2010.

[Zhang *et al.*, 2005] H. Zhang, J. K. Liu, X. Y. Liu, and X. Z. Guo. Modeling and prediction of freeway traffic flow using seasonal arima models. *Journal of Tianjin University*, 88(10):1675 – 1679, 2005.

[Zhang *et al.*, 2017] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*, pages 1655–1661, 2017.