This is a repository copy of *A capsule network for traffic speed prediction in complex road networks*.

White Rose Research Online URL for this paper:
http://eprints.whiterose.ac.uk/134023/

Version: Accepted Version

A Capsule Network for Traffic Speed Prediction in Complex Road Networks

Youngjoo Kim, Peng Wang, Yifei Zhu, and Lyudmila Mihaylova

Department of Automatic Control and Systems Engineering

The University of Sheffield

Sheffield, United Kingdom

{youngjoo.kim, peng.wang, yzhu42, l.s.mihaylova}@sheffield.ac.uk

Abstract—This paper proposes a deep learning approach for traffic flow prediction in complex road networks. Traffic flow data from induction loop sensors are essentially a time series, which is also spatially related to traffic in different road segments. The spatio-temporal traffic data can be converted into an image where the traffic data are expressed in a 3D space with respect to space and time axes. Although convolutional neural networks (CNNs) have been showing surprising performance in understanding images, they have a major drawback. In the max pooling operation, CNNs are losing important information by locally taking the highest activation values. The inter-relationship in traffic data measured by sparsely located sensors in different time intervals should not be neglected in order to obtain accurate predictions. Thus, we propose a neural network with capsules that replaces max pooling by dynamic routing. This is the first approach that employs the capsule network on a time series forecasting problem, to our best knowledge. Moreover, an experiment on real traffic speed data measured in the Santander city of Spain demonstrates the proposed method outperforms the state-of-the-art method based on a CNN by 13.1% in terms of root mean squared error.

Index Terms—traffic speed prediction, capsule network (CapsNet), convolutional neural network (CNN)

Fig. 1: Road network of central Santander city. Red lines denote road segments where the speed sensors are located.

I. INTRODUCTION

Traffic prediction is one of the central tasks for building intelligent transportation management systems in metropolitan areas. Traffic congestion causes delays and costs millions to the economy worldwide, which is worse in urban centres. Predicting the traffic flow will provide the stakeholders with tools for modelling and decision support.

Early approaches to traffic flow prediction are statistical techniques including support vector machines (SVM) [1] and the autoregressive integrated moving average (ARIMA) model [2]. These statistical approaches have been demonstrated to be effective as regression techniques for time series data. However, they do not address the spatio-temporal relationship of transportation networks and cannot be applied to a large-scale road network. Recently, machine learning technologies [3]–[7] have been actively applied given that traffic prediction is essentially to make estimations of future state based on big data. The spatio-temporal features of the traffic has been of great interest of researchers. Understanding the spatial evolution of traffic for the entire road network rather than for a small part of the network is necessary at both stages of off-line planning and on-line traffic management. Convolutional neural
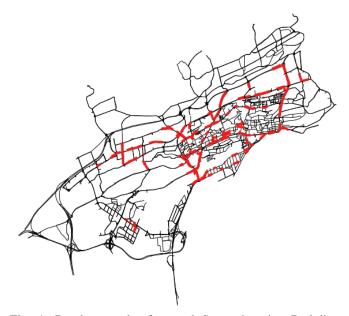
networks (CNNs) have been successful in dealing with spatial features of road networks [3], [4]. Besides, recurrent neural networks (RNNs) with long short-term memories (LSTM) [4], [5] and gated recurrent unit (GRU) [6] have been incorporated, considering the traffic flow prediction as a time series forecasting.

A novel approach has been proposed in [7] that converts the traffic speed into images where the traffic speed data of each road segment at each time step is expressed in the third dimension. A CNN is used to capture spatio-temporal features in the images. This method has been demonstrated to outperform other state-of-the-art methods. This approach differs from others in that other approaches simply treat the time dimension of the traffic flow as a channel of image data and therefore the temporal features of traffic flow are ignored [6]. However, this work was demonstrated on rectangular sub-networks of a metropolitan road network, which have a relatively simple topology.

The goal of this study is to devise a traffic speed prediction method for complex road networks. We are dealing with traffic
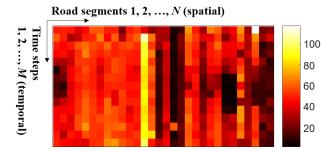
Fig. 2: Spatio-temporal image representation of traffic speed data (unit: km/h).

TABLE I: Layer parameters of CNN.

Layer	Parameter	Activation
Convolution1	(256, 3, 3)	ReLu
Pooling1	(2, 2)	-
Convolution2	(128, 3, 3)	ReLu
Pooling2	(2, 2)	-
Convolution3	(64, 3, 3)	ReLu
Pooling3	(2, 2)	-
Flattening	-	-
Fully-connected	-	-

data gathered in a metropolitan area of Santander city of Spain, whose road network is depicted in Fig. 1. The red lines denote road segments where induction loop detectors are installed to measure vehicle speeds. The speed sensors are sparsely located in a complex network. Adjacency in the spatio-temporal image does not necessarily mean adjacency in the road network. In this case, some spatial features would be disregarded by a CNN because it uses the max pooling operation to construct higher order features locally. Therefore, we utilize a capsule network (CapsNet) [8], [9] that replaces the pooling operation with dynamic routing, which enables to take into account important spatial hierarchies between simple and complex objects. We propose a CapsNet architecture designed to be suitable for traffic speed prediction and demonstrate its effectiveness by comparing it with the CNN-based method in [7]. To our best knowledge, this is the first application of the CapsNet to a time series forecasting problem.

The rest of this paper is organized as follows. It starts with addressing the method of converting traffic data into images in Section II. After that, an existing approach to traffic speed prediction based on a CNN is introduced. We then present the proposed architecture of the CapsNet designed for traffic speed prediction. The methods and results of the performance evaluation with a real dataset are given in Section III. Finally, Section IV presents a summary and conclusion.

II. TRAFFIC SPEED PREDICTION

A. Traffic Speed Data as an Image

Each induction loop sensor records time history of traffic speed on different road segments. In order to consider the spatio-temporal relationship, the traffic data are converted to an image with two axes representing time and space. As a result, we have an image as an $M \times N$ matrix where M and N denote the number of time steps and the number of sensors, respectively. The matrix is then represented as:

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1N} \\ \vdots & \ddots & \vdots \\ x_{M1} & \cdots & x_{MN} \end{bmatrix} \tag{1}$$

where $x_{mn}(m = 1, ..., M, \ n = 1, ..., N)$ denotes the traffic speed at m-th time step in n-th road segment. For example,

Fig. 2 depicts the spatio-temporal image representation of traffic speed data.

Suppose we have traffic speed data from N sensors and we are going to predict the traffic speed in L time steps ahead based on data from previous M time steps. Given the overall time history of traffic speed, a strip of $M \times N$ matrix becomes an input and the data in the next time steps act as labels in training neural networks. The output can be an array with a size of LN, which can be obtained by reshaping a strip of $L \times N$ matrix to an array as:

$$Y = \begin{bmatrix} y_1 & \cdots & y_N & y_{N+1} & \cdots & y_{LN} \end{bmatrix} \tag{2}$$

B. CNN for Traffic Speed Prediction

The CNN has been demonstrated to be significantly effective in understanding images by using max pooling and successive convolutional layers that reduce the spatial size of the data flowing through the network. These procedures increase the field of view of high-level layers and allow them to capture high-order features of the input image.

We use the CNN architecture proposed in [7] as the baseline. This consists of three pairs of a convolutional layer and a pooling layer followed by a flattening operation and a fully-connected layer. Fig. 3 depicts the architecture of the CNN for traffic speed prediction. The three convolutional layers have 256, 128, and 64 channels, respectively, of size 3×3. Each convolution layer involves a rectified linear unit (ReLu) activation function to give nonlinearity to the network.

Pooling layers have filters of size 2×2 applied with a stride of 2. This downsamples every depth slice in the input by 2 and reduces the redundancy of representation by removing 75% of the activations. The output of each max pooling filter is determined by taking the maximum over 4 numbers in a 2×2

TABLE II: Layer parameters of CapsNet.

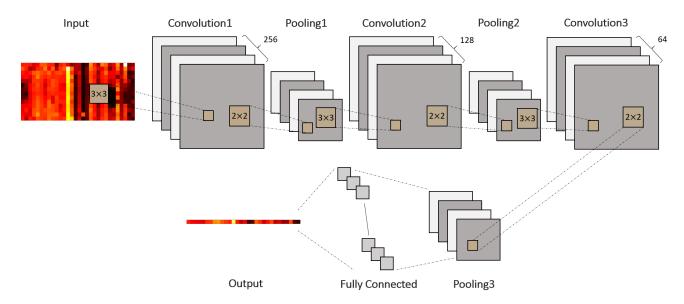| Layer	Parameter	Activation
Convolution1	(32, 3, 3)	ReLu
Convolution2	(32, 3, 3)	ReLu
PrimaryCaps	(128, 3, 3)	ReLu
	Capsule size 8	-
TrafficCaps	Capsule size 16	-

Fig. 3: Architecture of CNN for traffic speed prediction.

region. The output of the last pooling layer is transformed to a vector by the flattening operation and this contains the final and the highest-level features of the input traffic history. Lastly, the flattened output goes through a fully-connected layer to provide the prediction. The output of the fully-connected layer now has the same dimension as the label vector in (2). The parameters of the CNN is presented in Table I.

C. Proposed CapsNet Architecture

CNNs have worked surprisingly well in various applications. Nonetheless, max pooling in CNNs is losing valuable information by just picking the neuron with the highest activation. CapsNet has been proposed in [8], [9] to address the drawback of CNNs.

A capsule is a group of neurons that encodes the probability of detection of a feature as the length of their output vector. Each layer in a CapsNet contains many capsules that represent different properties of the same object. One of the main characteristics of capsules is that capsules have vector forms and their activations provide vector outputs whereas artificial

neurons go through scalar operations. More importantly, the CapsNet is trained by an algorithm called dynamic routing proposed in [9]. The dynamic routing is executed between two successive capsule layers to update weights that determine how the low-level capsules send their input to the high-level capsules that agree with the input. In other words, the weights are determined based on the dot product of the low-level capsule and the high-level capsule where the dot product captures the similarity of two vectors. Each weighted sum of the low-level capsules is then passed through the squash function that forces the length to be no more than 1 while preserving the direction of the vector. Unlike CNNs, the CapsNet does not throw away information that is most likely relevant to the task at hand, like relative relationships between spatio-temporal traffic features.

In the proposed architecture, as depicted in Fig. 4, the first two convolutional layers convert the spatio-temporal traffic image to the activities of local feature detectors used as inputs to the third layer. The third layer, called PrimaryCaps, is another convolutional layer that has 128 channels with a
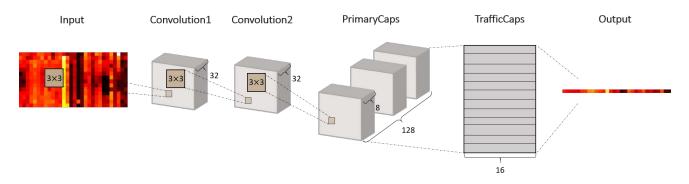

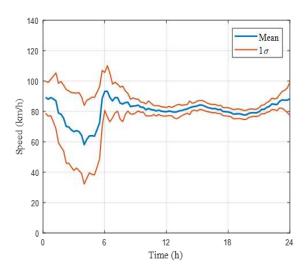Fig. 4: Architecture of CapsNet for traffic speed prediction.

Fig. 5: Mean and 1-sigma variation of all 1-year data on a road segment.

3×3 kernel. All the convolution operations are performed with a stride of 1 with zero padding, involving a ReLu nonlinearity. Each capsule in the PrimaryCaps layer is an 8-dimensional vector and capsules in a cuboid are sharing their weights with each other. The final layer, called TrafficCaps, has a 16-dimensional capsule per road segment. The dynamic routing is performed between PrimaryCaps and TrafficCaps with 3 iterations. The dynamic routing algorithm captures the relationship between all the capsules in the PrimaryCaps layer and each capsule representing each road segment. In this way, any distant local feature can contribute to characterizing the capsules in the TrafficCaps layer. Here we consider the length of each 16-dimensional capsule vector in the TrafficCaps layer as the traffic speed on the corresponding road segment. The parameters of the proposed CapsNet are given in Table II.

III. Performance Validation with Real Data

We use traffic speed data measured every 15 minutes on road segments in the central Santander city for a year of 2016. The dataset is from the case studies of the SETA EU project [10]. Excluding days when the sensors did not work, the spatio-temporal traffic dataset is a matrix with a size of $33054 \times N$ where N denotes the number of road segments. Each sparsely missing measurement is masked with an average of measurements taken at the same time in the other days. We use traffic data from January to September as a training set and the remaining data from October to December as an evaluation set. As an example, the average speed and a 1-year variation on a road segment are presented in Fig. 5. Note that each road segment would have different statistics and no topological information of the road network is given. Understanding and predicting the spatio-temporal relationship of traffic between different road segments in different time slots are the duty of the neural networks. The CNN and CapsNet described in
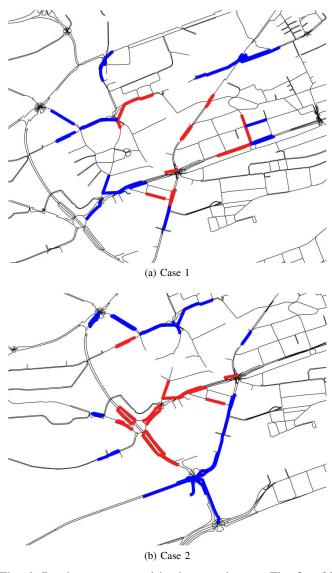

(a) Case 1

(b) Case 2

Fig. 6: Road segments used in the experiments. The first 20 segments are marked in red and the adjacent 30 segments are marked in blue.

Section II. B) and II. C), respectively, performed the following four prediction tasks:

- Task 1: 15-min prediction with 150-min traffic history on 20 road segments ($L = 1, M = 10, N = 20$)
- Task 2: 30-min prediction with 150-min traffic history on 20 road segments ($L = 2, M = 10, N = 20$)
- Task 3: 15-min prediction with 210-min traffic history on 50 road segments ($L = 1, M = 14, N = 50$)
- Task 4: 30-min prediction with 210-min traffic history on 50 road segments ($L = 2, M = 14, N = 50$)

The traffic prediction tasks are performed in two sets of road segments as depicted in Fig. 6. 20 road segments used in Task 1 and Task 2 are marked in red and the other 30 road segments, used in Task 3 and Task 4 together with the red segments, are marked in blue. Note that traffic data in adjacent
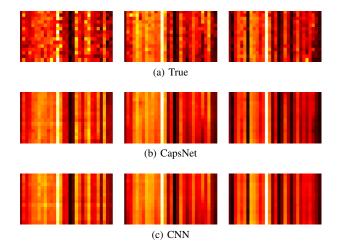
(a) True

(b) CapsNet

(c) CNN

Fig. 7: Prediction result for traffic speed compared in the form of images.

TABLE III: Prediction performance
(unit: % for MRE, km/h for MAE and RMSE).

(a) Case 1

	CNN			CapsNet		
	MRE	*MAE*	*RMSE*	*MRE*	*MAE*	*RMSE*
Task 1	5.668	6.102	10.30	0.444	5.675	8.853
Task 2	0.649	6.204	10.47	0.289	5.791	9.179
Task 3	18.14	6.323	10.68	5.146	5.790	9.257
Task 4	4.661	6.583	10.85	0.876	5.898	9.472

(b) Case 2

	CNN			CapsNet		
	MRE	*MAE*	*RMSE*	*MRE*	*MAE*	*RMSE*
Task 1	37.35	6.519	10.98	1.555	6.109	9.362
Task 2	21.41	6.667	11.19	10.97	6.240	9.718
Task 3	19.76	6.915	11.29	9.746	6.113	9.674
Task 4	2.333	6.957	11.38	4.146	6.243	9.913

road segments are not always located close to each other in the spatio-temporal image. We attempt to verify the methods on larger spatio-temporal images in Task 3 and Task 4 where the neural networks are required to capture the spatio-temporal features scattered in a larger region.

In our Tensorflow implementation, each network employs mean squared error (MSE) as a loss function and we use the Adam optimizer [11] with the exponentially decaying learning rate to minimize the sum of the MSE. We scale the traffic speed data into the range [0,1] before feeding in the neural networks.

The prediction result can be compared with the true values in the form of images. Fig. 7 depicts the image representation of the true traffic speed and predictions by the CapsNet and the CNN. Traffic speed data at 3 different time periods are drawn where the images in the same column represent the traffic data at the same time period. It is observed that the deep learning methods provide similar results as if a smoothing filter is applied to the true traffic images.

The images shown in Fig. 7 are just snapshots of the result. Since we have a lot of data in the evaluation set, statistical performance metrics are required to assess the overall performance of the networks. Mean relative error (MRE) is one of the most common metric to quantify accuracy of different prediction models in general. However, the error of a smaller value of speed might result in larger MRE and vice versa. Thus, we further employ mean absolute error (MAE) and root mean squared error (RMSE) as more intuitive metrics for assessing the speed prediction performance. The three performance metrics are defined as:

$$MRE = \frac{\sum_{i=1}^{I} |y_i - \hat{y}_i|/y_i}{I} \qquad (3)$$

$$MAE = \frac{\sum_{i=1}^{I} |y_i - \hat{y}_i|}{I} \qquad (4)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{I} (y_i - \hat{y}_i)^2}{I}} \qquad (5)$$

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.

The performance of the CNN and CapsNet has been assessed with their best settings. 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 resultant performance of the neural networks on the four tasks with two datasets is presented in Table III. The MRE does not seem to provide consistent results. On the other hand, the MAE and RMSE increase as the input and output sizes increase from Task 1 to Task 4. The CapsNet shows better (smaller) MAE and RMSE than the CNN in all the tasks in both cases. The performance difference is larger in Task 3 and Task 4 where the size of the input image is larger. The CapsNet provided 6.58% smaller MAE in Task 1 and Task 2 and 10.2% smaller MAE in Task 3 and Task 4. We conclude the CapsNet is better at capturing the relationship between distant spatio-temporal features as expected. In average, the CapsNet provides 8.24% and 13.1% improvement in MAE and RMSE, respectively, compared with the CNN.

A drawback of the CapsNet is that it takes a longer time to train the network. In our experiment of Task 1, the CapsNet is about 30 times slower than the CNN. The computation time difference becomes severe for tasks with larger output sizes. The number of trainable parameters in the CapsNet varies from 8.24×10^6 (Task 1) to 143×10^6 (Task 4) whereas that in the CNN varies from 0.374×10^6 (Task 1) to 0.410×10^6 (Task 4). Given increased input and output sizes, the routing algorithm requires a significant increase in the number of trainable parameters because it deals with a full-scale image features by testing all the combinations between multidimensional vectors, called capsules. On the other hand, the number of trainable parameters shows a mere increase in

the CNN, which is contributed by the pooling operation.

IV. Conclusion

This paper presents a capsule net framework that captures the spatio-temporal features of traffic speed and provides short-term traffic speed predictions. The vehicular traffic speed measured by magnetic loop detectors is represented as images that are fed into the developed capsule network. Traffic speed predictions by the proposed CapsNet architecture are compared with those by a CNN-based method. Experiments performed on 1-year data measured on road segments in Santander city demonstrate the proposed CapsNet provides more accurate speed predictions than the CNN. The performance difference is larger in experiments with a larger dataset. This result implies the CapsNet is better at learning spatio-temporal features in the test data, with 13.1% improvement in RMSE with respect to the CNN.

Acknowledgment

References

[1] C.-H. Wu, C.-C. Wei, D.-C. Su, M.-H. Chang, and J.-M. Ho, "Travel time prediction with support vector regression," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, pp. 276–281, 2004.

[2] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[3] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[4] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," *AAAI Conference on Aritificial Intelligence*, pp. 1655–1661, 2017.

[5] 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, 2015.

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

[7] 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.

[8] G. E. Hinton, S. Sabour, and N. Frosst, "Matrix capsules with em routing," *International Conference on Learing Representations*, 2018.

[9] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *Conference on Neural Information Processing Systems*, pp. 3856–3866, 2017.

[10] SETA EU Project, A ubiquitous data and service ecosystem for better metropolitan mobility, Horizon 2020 Programme, 2016. [Online]. Available: http://setamobility.weebly.com/

[11] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.