Traffic flow speed forecast on road graph with convolutional neural networks

Nikolay Prokoptsev[1], Andrey Alekseenko[2], Yaroslav Kholodov[3]

***Abstract*— The short-term prediction of road traffic condition is an essential task of transportation modeling. A number of solutions exist – both model-driven and data-driven – that are successful in capturing the dynamics of traffic flow. Neural networks, one of the prominent data-driven approaches, show promising performance in modeling the complexity of traffic flow. We present a neural network architecture for traffic flow prediction on a real-world road network graph. The model is based on the combination of a recurrent neural network and a graph convolutional neural network. The resulting neural network was trained on raw data from traffic flow detectors from the US highway system and achieved lower prediction error than previously published models, such as Vector Auto Regression, LSTM and Graph Convolution GRU.**

***Index Terms*— Traffic flow prediction, Recurrent Neural Networks, Graph Convolutional Networks**

I. Introduction

Traffic flow and speed prediction are among the main tasks in the field of Intelligent Transportation Systems (ITS). A reliable and accurate system for traffic forecasting is essential for the efficient functioning of intelligent transport infrastructure — and any modern city.

This problem can be formulated as a task of predicting some observable value, such as mean speed or intensity of the traffic flow, at a given node of the transport network based on current and historical data from nodes of the graph. In short-term forecasting problems, the prediction horizon typically does not exceed 1 hour.

Historically, two different approaches were used for this task. In the 1950s the first mathematical models of traffic flow emerged [1], [2]. These models try to describe traffic flow using the method of kinematic waves. As an alternative to model-based approach, the exponential increase in processing power of modern computer systems made data-driven approach possible. Some examples of data-driven approach are ARIMA, KNN, Support Vector Regression, and Artificial Neural Networks.

The ARMA (Autoregressive moving average) model family played a key role in the development of this field of research. It includes such one-dimensional models as ARIMA [3] and Seasonal ARIMA [4], which is a modification of ARIMA that makes use of seasonality in data. The main drawback of these models is they treat each sensor independently, ignoring the dependencies between neighboring sensors.

[1]Innopolis University, 1 Universitetskaya st., Innopolis, Russia n.prokoptsev@innopolis.ru

[2]Institute of Computer Aided Design of the Russian Academy of Sciences, 19/18 2-ya Brestskaya st., Moscow, Russia

[3]Innopolis University, 1 Universitetskaya st., Innopolis, Russia

In the multivariate case, the most used model due to its simplicity is vector autoregression (VAR). It can take into the account spatial dependencies of the data. However, with the increase in the number of analyzed time series, the complexity of VAR-models grows quadratically, as it models dependencies of all sensors with all other sensors, which makes it difficult to apply them to large transport networks. The common limitation of this autoregressive models is that they model only linear dependencies between sensors which is not always the case.

Due to the stochastic and non-linear nature of the traffic data, researchers have paid much attention to nonparametric methods. The k-nearest neighbors (k-NN) method, a classic algorithm for regression, was proposed [5] for short-term freeway traffic forecasting. However, its performance was on par with linear models and not better.

The Online Support Vector Regression (Online SVR) model [6] has been successfully applied for predicting travel time in a transport network. This model allows updating the weights of the prediction function in real time using incremental learning. The method performed a bit better than baselines but significantly outperforms them in atypical non-recurring conditions, such as car accidents.

Researchers have long sought to use the potential of artificial neural networks (ANN) for traffic forecasting [7], but ANN with one hidden layer yielded worse results than SVR with the RBF kernel [8].

Deep learning methods are widely and successfully used for a variety of tasks, such as machine translation, classification, and recognition of objects. Deep Belief Networks (DBN) have shown the potential of understanding the stochastic properties of traffic in unsupervised fashion [9]. Stacked Autoencoders [10] and Stacked Denoising Autoencoders [11] were used to learn generic traffic flow features and find its latent properties. Such neural networks consider spatial and temporal nonlinear correlations of traffic data, but they are difficult to train because of the fully connected layers in their architecture.

Recurrent Neural Networks (RNN) have become increasingly popular for time series and language modeling and achieved state-of-the-art results on popular datasets [12]. Whereas feedforward networks only consider a fixed number of timesteps to predict the next one, recurrent neural networks can take into the account all preceding timesteps. Vanilla RNNs failed to scale to longtime dependencies due to vanishing gradient problem. This problem is addressed by Long Short-Term Memory (LSTM) [13] and Gate Recurrent Unit (GRU) [14] networks. They catch not only short-term

dependencies but also long-term ones. LSTM and GRU cell consists of several fully-connected components that capture spatiotemporal dependencies in data. The required complexity per timestep is $O(n^2)$, where n is the number of detectors.

In GRU and LSTM the use of hyperbolic tangent and the sigmoid activation functions results in gradient decay over layers, as these activation functions quickly saturate. Independent RNN [15] is a modification of RNN where neurons in each layer are independent of each other. Layers can be stacked to allow connection between neurons. Compared to LSTM this architecture allows to process much longer time series and construct deeper networks.

Convolutional Neural Networks (CNN) are the primary tool for modeling data with a spatial structure. They are particularly successful in extracting spatially local input patterns. These networks dont suffer from the curse of dimensionality as they use space-invariant local filters. Moreover, stacking such filters allows capturing global features [16]. Vanilla CNNs work well with grid topology data, such as images, video, speech, but they are not designed for complex topologies such as graph.

Generalization of CNNs to graphs using tools from graph signal processing have been proposed [17]. They approximate the graph Laplacian using Chebyshev expansion with free parameters that are learned during the training. These networks employ fast local spectral filters to extract spatial properties in a neighborhood of a vertex of order K with $O(K|\mathcal{E}|)$ complexity, where $|\mathcal{E}|$ is a number of edges.

Graph Convolutional Networks [18] simplify the previous method using an approximation of first-order Chebyshev expansion in Defferrard model. Each filter is applied only to the neighborhood of each node. In many experiments, this architecture surpasses other methods in terms of predicting accuracy.

The combination of recurrent neural networks and convolutional neural networks makes it possible to model spatially-temporal dependencies in data. The combination LSTM cell with convolution on a two-dimensional grid was proposed, where input-to-state and state-to-state transitions use convolution instead of matrix multiplication [19]. Later it was extended [20] to the case of the graph. Li et al. [21] proposed encoder-decoder architecture with Graph Convolutional GRU (GCGRU) which solves the noise accumulation problem.

II. Methodology

A. Problem statement

The value of flow, occupancy, and speed across all lanes at each detector station are aggregated and averaged for each 5-min interval for each detector station [22]. This historical data of traffic flow from N sensors at M previous timesteps can be represented as the matrix $V = [v_1, v_2, ..., v_M]^T$ of size $M \times N$, where v_i is the vector of sensor readings at timestep i.

The connectivity between the sensors is defined by the undirected weighted graph of the transport network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V}$ is the set of vertices (sensors), $\mathcal{E}$ is the set of edges (roads), $\mathcal{A}$ is the weighted adjacency matrix of the graph that represents the connectivity of the sensors in the transport network.

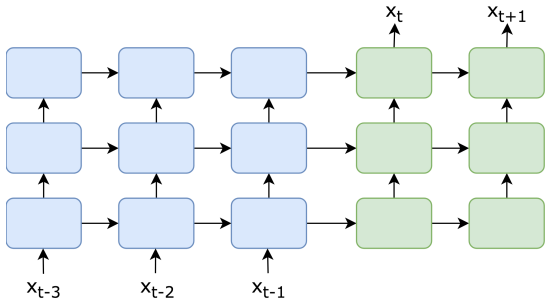

Fig. 1. Encoder-decoder architecture for traffic flow forecasting. Blue cells represent the encoder, green cells represent the decoder.

Consequently, the problem of predicting the traffic flow in the road network can be formulated as follows:

$$v_{M+T} = \arg\max p(v_{M+T}|v_1, v_2, ..., v_M, \mathcal{G}) \qquad (1)$$

B. Proposed method

The traffic flow detectors form a weighted graph, where we can define edge weights as a function of the distance between detectors. We assume that the influence between different parts of the road tends to decrease as the distance between them increases.

In this paper, we propose a new neural network architecture based on Independent RNN (IndRNN) [15] and Graph Convolutional Network (GCN) [18]. We let the Graph Convolutional Network model spatial dependencies between sensors, while IndRNN is responsible for temporal dependencies. Each layer of the proposed model consists of an IndRNN, in which matrix multiplication is replaced with graph convolution followed by ReLU activation. This way, the convolution in each layer models spatial dependencies between adjacent nodes, while IndRNN models temporal dependencies of neurons independently. Stacking k such layers will model spatiotemporal dependencies in k-th order neighborhood of each node.

Graph convolution for signal $X \in \mathbb{R}^{N \times C}$ with C input channels is defined [18] as follows:

$$Z = \widetilde{D}^{-\frac{1}{2}} \widetilde{A} \widetilde{D}^{-\frac{1}{2}} X \Theta, \qquad (2)$$

where $Z \in \mathbb{R}^{N \times F}$ is the convolved signal matrix, $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$ is the diagonal node degree matrix, $\widetilde{A} = A + I_N$, A is an adjacency matrix of graph, I_N is an identity matrix, $\Theta \in \mathbb{R}^{C \times F}$ is the matrix of filter parameters.

Instead of processing inputs at each timestep with learned fully connected weight we use graph convolution. We employ skip connections in our model to speed up training and make each layer model the residuals [23]. To use skip connection in the first layer we have to upscale data, that has one feature map, to the same number of feature maps as are used for layer output. This transformation is performed with same

transformation matrix for each detector. After the last layer, we have to downscale the number of feature maps to one. We tie the weights of matrix used for downscaling and upscaling to reduce the number of parameters.

Traditional models of traffic flow prediction, such as ARIMA and LSTM, suffer from noise propagation. This happens when we use predicted values from the previous step to predict the next ones. To deal with this problem we employ sequence to sequence model with encoder-decoder architecture [14], depicted on Figure 1.

To train the sequence to sequence model we use teacher forcing. The training consists of maximizing the next value in the time series given the recurrent state and the ground truth value of the previous timestep. For inference, we replace the ground truth value with the predicted value. This behavior prevents neural network from adjusting to the accumulating noise during the training phase. To make a smooth transition from feeding true value to feeding generated value, we use scheduled sampling [24].

III. Experiments

We used traffic data obtained from the Caltrans Performance Measurement System (PeMS) [25] District 4. The traffic speed averaged over 5-minute intervals was measured by 256 detectors on the Los Angeles highways for 4.5 months in 2016. We divide data equally into train, validation and test set, 1.5 months each. To construct the detector graph, we define the adjacency matrix as follows:

$$a_{ij} = \begin{cases} \exp\left(-\frac{d_{ij}^2}{\sigma^2}\right) & \text{if } \exp\left(-\frac{d_{ij}^2}{\sigma^2}\right) > \varepsilon, i \neq j, \\ 0 & \text{otherwise,} \end{cases} \quad (3)$$

where d_{ij} is the distance between detectors, and σ and ε are parameters responsible for how sparse this matrix would be.

We compare our model with the following time series prediction baselines: 1) VAR, 2) Multilayered LSTM with fully connected input-to-state and state-to-state transitions (FC-LSTM) [26], 3) Graph Convolutional Gated Recurrent Unit (GCGRU). All models are implemented in Tensorflow [27] and are trained minimizing the mean square error with Adam optimizer [28]. We use batch size 32 and learning rate 0.001 and decrease it by the factor of 10 when performance on validation stops improving. The gradients are clipped by global norm 0.25. In addition, we perform standardization of inputs and fill missing values with zeros.

The VAR model is set to consider 12 lags. FC-LSTM consists of three layered LSTM network with tanh activation and hidden state of size 512. GCGRU is set to stack 2 recurrent graph convolution layers in decoder and encoder with each layer containing 64 GCGRU units. The maximum number of hops in graph convolution is set to 2.

Proposed model contains 7 residual layers of convolutional IndRNN with 64 channels in both encoder and decoder. It was set to consider 40 lags and predict 12 future timesteps. For scheduled sampling we employ sigmoid schedule, where we sample the true previous token at step i with probability ϵ_i:

$$\epsilon_i = k_0/(k_0 + \exp(\max(0, i - k_1)/k_0)), \quad (4)$$

where constants $k_0 \geq 1$ and $k_1 \geq 0$ depend on the expected speed of convergence. We we empirically found that $k_0 = 30$ and $k_1 = 5000$ leads to fastest convergence.

We evaluate each approach for 3, 6 and 12 steps (15, 30, 60 minutes) ahead forecasting of a 5-minute average speed of each detector on the graph. For evaluation we use three metrics: mean absolute error (MAE), mean relative error (MAPE), and standard deviation (RMSE):

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t| \quad (5)$$

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| * 100\% \quad (6)$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2} \quad (7)$$

where n is a number of predictions, y is the ground truth value of dependent variable (speed), $\hat{y}$ is the predicted value.

TABLE I
Performance comparison of different approaches of traffic flow forecasting

	MAE (mph)	MAPE (%)	RMSE (mph)
15 minutes ahead			
VAR	4.29	8.29	6.30
FC-LSTM	2.92	6.22	5.40
GCGRU	2.98	6.15	4.89
Proposed Method	2.54	5.45	4.56
30 minutes ahead			
VAR	5.72	11.15	8.51
FC-LSTM	3.20	7.00	6.10
GCGRU	3.67	7.57	5.97
Proposed Method	2.78	6.14	5.10
60 minutes ahead			
VAR	7.26	14.15	10.83
FC-LSTM	3.52	7.86	6.87
GCGRU	4.92	10.01	7.58
Proposed Method	3.15	7.07	5.81

Table I shows the comparison of different approaches for traffic flow prediction for 15, 30 and 60 minutes ahead forecast.

The proposed method based on recurrent networks is more accurate than other methods. It shows noticeable gains (7% to 15%) in prediction performance and provides the best results considering all the metrics for all the horizons of the prediction. In terms of model complexity, the proposed network and GCGRU has $\sim 10^5$ parameters each, while FC-LSTM has $\sim 10^7$.

IV. Conclusion

In this article, we propose a data-driven approach for short-term traffic flow prediction. We present the neural network architecture capable of capturing the dynamic and complexity of traffic flow. The model is based on the combination of a recurrent neural network and a convolutional neural network on a graph. Encoder-decoder architecture paired with scheduled sampling successfully alleviates accumulating noise problem and allows predictions for multiple steps ahead.

The resulting neural network is trained on raw data from traffic flow sensors from the highway system around Los Angeles, USA, and the proposed model outperforms previous state-of-the-art methods, such as GCGRU, on a real traffic flow dataset by 7-15%.

Acknowledgment

This research has been supported by Russian Science Foundation (grant ID 14-11-00877).
The authors would also like to thank Google Cloud for providing access to GPUs that were used for training some of the models.

References

[1] P. I. Richards, "Shock Waves on the Highway," *Operations Research*, vol. 4, no. 1, pp. 42–51, 1956.

[2] M. J. Lighthill and G. B. Whitham, "On Kinematic Waves. II. A Theory of Traffic Flow on Long Crowded Roads," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 229, no. 1178, pp. 317–345, 1955.

[3] M. S. Ahmed and A. R. Cook, "Analysis of freeway traffic time-series data by using Box-Jenkins techniques.," *Transportation Research Record*, no. 722, p. 116, 1979.

[4] 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.

[5] B. G. A. Davis, A. Member, and N. L. Nihan, "Nonparametric regression and short - term freeway traffic forecasting," vol. 117, no. 2, pp. 178–188, 1991.

[6] M. Castro-Neto, Y. S. Jeong, M. K. Jeong, and L. D. Han, "Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions," *Expert Systems with Applications*, vol. 36, no. 3 PART 2, pp. 6164–6173, 2009.

[7] M. J. M. Jun and M. Y. M. Ying, "Research of Traffic Flow Forecasting Based on Neural Network," *2008 Second International Symposium on Intelligent Information Technology Application*, vol. 2, no. 973, pp. 451–456, 2008.

[8] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 871–882, 2013.

[9] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, 2014.

[10] 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.

[11] Q. Chen, X. Song, H. Yamada, and R. Shibasaki, "Learning Deep Representation from Big and Heterogeneous Data for Traffic Accident Inference," *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI-16)*, pp. 338–344, 2016.

[12] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[13] S. Hochreiter and J. Urgen Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," 2014.

[15] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently Recurrent Neural Network (IndRNN): Building A Longer and Deeper RNN," no. 1, 2018.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[17] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering," no. Nips, 2016.

[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[19] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting," pp. 1–12, 2015.

[20] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured Sequence Modeling with Graph Convolutional Recurrent Networks," no. 2013, pp. 1–10, 2016.

[21] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting," pp. 1–12, 2017.

[22] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Mining Loop Detector Data," no. 01, pp. 96–102.

[23] J. Kim, M. El-Khamy, and J. Lee, "Residual LSTM: Design of a deep recurrent architecture for distant speech recognition," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2017-Augus, pp. 1591–1595, 2017.

[24] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks," pp. 1–9, jun 2015.

[25] "Caltrans Performance Measurement – State of California." http://pems.dot.ca.gov/.

[26] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," pp. 1–9, 2014.

[27] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[28] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," pp. 1–15, 2015.