Urban traffic congestion mapping using bus mobility data

Shiva R. Iyer[1] and Kate Boxer[2] Lakshminarayanan Subramanian[3]

[1] Department of Computer Science - New York University
shiva.iyer@cs.nyu.edu
[2] Department of Computer Science - Columbia University
[3] Department of Computer Science - New York University
lakshmi@cs.nyu.edu

Abstract Mobility traces of public transportation vehicles, maintained by local government departments, are a very widely available but hardly used data source for traffic congestion analysis in a city. In this paper, we describe our experiences with the historical bus mobility trace data from the New York City MTA, collected over a period of 3 months in 2014. Predicting the congestion state of a road segment is difficult because of complex spatiotemporal dependencies. Our work focuses on the prediction of congestion state of a road segment, given historical trends and information from neighboring road segments. We leverage deep learning architectures such as LSTMs in order to help capture the longer term dependencies in this modelling task. We demonstrate the feasibility of using such a data source for traffic speed prediction and forecasting by showing that we are able to estimate future speeds in a segment with an error of less than 2 m/s (average RMSE $\sim$ 1 m/s) despite the presence of large amount of gaps and noise in the data. We believe a system such as this for congestion prediction and mapping using limited data would help both citizens as well as urban planners.

Keywords: road traffic congestion, bus mobility traces, deep learning, time series

1 Introduction

Analysis of traffic congestion on the roads in a city is a problem that has been looked at for decades. The ultimate goal is usually the estimation, as accurately as possible, of travel times to help in route planning. To that end, the subproblems are usually one or more of the following – estimating traffic speeds in roads [3,5], predicting occurrences of traffic jams in advance [9], minimizing traffic jams to enable smooth flows [4] and others [12]. Today, widespread services like Google Maps or Waze provide solutions to all of these problems by largely relying on crowdsourced information by mobile phone users on the road [2]. While they provide acceptable performance in many cases, the problem currently is that there are parts of the world where crowdsourced road traffic data are either not available or are not the most recent [18]. Even in a city with good mobile phone penetration among the population, depending on the residents of a city to report their location is usually not easily achieved, especially in places with strict regulations on free data access and privacy concerns among citizens.

We propose an approach to traffic congestion modelling using public bus mobility trace data, which we believe is easier to obtain since it is freely available, as opposed to crowdsourced data from private companies or speed data from loop detectors on selected roads. One of the disadvantages of loop detector data is that the spatial coverage is rather limited. Speed data is usually collected only from highly limited number of locations in a city including selected arterial roads, bridges and freeways [15,14]. New York City Department of Transportation (DOT) provides speed data using loop detectors over most freeways, bridges and tunnels, but the data is generally not available continually from all the locations. In contrast, buses ply on the city's roads 24 hours a day[4], 7 days a week, and so data is available continually across tens of thousand more mobile locations. Another advantage of bus mobility trace data is that buses repeatedly

[4] Even if not 24 hours, traffic congestion prediction is usually of importance only during peak hours or when traffic levels on the roads are higher

ply on the same routes for long periods of time, sometimes decades, thus enabling us to study changes in traffic behavior at specific locations of interest. Buses, being bigger vehicles on the road occupying more space, also give us a lower bound on the how fast traffic can move in the city.

Our contribution in this paper is primarily demonstrating the feasibility of using bus mobility trace data to build traffic speed prediction models. We obtain reasonable performance in prediction of travel speed in segments using Long Short-Term Memory (LSTM) networks [7], a type of recurrent neural network (RNN) that has been used in speech prediction, language modelling and other sequence prediction tasks with great accuracy. We employ historical speed data as well as the graph structure, and show prediction with a mean RMSE $\sim$ 1 m/s. In certain segments, we are able to predict the speeds with an error of less than 1 m/s. Our main takeaways in this work – i) it is feasible to use public bus mobility data to build models for traffic speed prediction, as prediction errors are reasonably low and training times are also low ($\sim$ 10 s for a segment), ii) LSTM networks are fairly robust and well-suited for traffic speed prediction, including capturing of peaks and spikes.

2 Related Work

More traditional approaches to travel time prediction such as ARIMA models and Kalman filters work well to estimate future values in timeseries, and have been used for short-term traffic flow forecasting [11,12]. But ARIMA approaches do not model spatial dependencies effectively, which is important in a traffic congestion prediction system. Also, timeseries models rely on the stationarity assumption, which does not hold in the case of traffic speeds. The traffic state in a segment depends strongly on the state of traffic upstream as well as downstream. With this idea, there has been more recent work on using probabilistic graphical model based approaches [8,1] as well as neural networks for capturing such dependencies and obtaining better results [3,10,17].

Vlahogianni et al. [16] put together an excellent review of a large number of works on traffic state prediction – speed, travel time and volume. Many of them use input data gathered from loop detectors that are placed throughout the road network of a city, both in downtowns as well as on freeways, a fact also pointed out by [6]. There are far fewer works that use data from GPS devices and virtually none using public bus mobility data. We differentiate from other works in this space by using a dataset of completely different type and showing feasibility of prediction.

3 The Problem

The problem of traffic congestion estimation is to estimate a *congestion parameter* for a road segment, given as input features of the road segment and the surrounding network. A road segment could be any well-defined portion of a road, such as between two intersections or traffic lights or bus stops. A set of features might include historical values of the congestion parameter in that segment and neighboring segments, number of outgoing segments, number of incoming segments, number of lanes, time of day and so on. The congestion parameter is typically one of two quantities – the average speed of vehicles traversing the segment at a given time, or the average time taken by vehicles to traverse the segment. We have experimented with using both, and from our experience, believe that speed is a better parameter as it is a better indication of the level of congestion on the road segment rather than the time, since the time would depend on the length of the road segment as well. Hence we use the speed as the congestion parameter in this paper.

We define a *segment graph* as the connected and directed graph of all road segments in the city or region of interest. Every road segment is a node in the graph, and there is a directed edge from node A to node B if traffic from segment A is allowed to enter segment B according to the local traffic rules. The rationale behind defining edges this way will become clear when we define neighborhoods shortly. All edges have unit weight.

Next we define the neighborhood of a segment. Let us represent the set of neighbors of segment i as $\mathcal{N}(k, i)$, where k is a parameter that represents the number of *hops*. Neighbors do not have to be adjacent,

but could be a fixed number of *hops* away. A k-hop neighborhood of segment i is defined as the set of all segments that are at a distance of at most k from i, as represented by equation 1.

$$\mathcal{N}(k,i) = \{j \mid dist(i,j) \leq k\} \tag{1}$$

Let us represent the congestion parameter (speed) for a segment i at time t as $X_{i,t}$. Then, equation 2 illustrates our prediction problem. We design a neural network architecture to estimate this function F that maps some of the input features mentioned above to the output speeds. The δ represents the length of history to be used in the estimation problem.

$$\mathrm{F}(\bigcup_{\delta \in \{0,1,\ldots\}} \{X_{j,t-\delta} \mid j \in i \cup \mathcal{N}(i)\}) = X_{i,t+1} \tag{2}$$

Thus, with the output being the future speed at a time t in segment i, the inputs are one or more of – historical speeds in segment i and each of its neighbors. We experiment with various lengths of history and the neighborhood size and present our results in this paper.

4 Dataset

We use the historical mobility traces of public buses in NYC [13]. The dataset consists of complete GPS traces of all of the MTA buses throughout the city day-wise over a period of three months in the year of 2014. Each bus has a GPS transponder fixed in it that transmits information every 30-40 seconds. Apart from lat-lon coordinates, the information includes – the timestamp of transmission, the bus route, the vehicle ID, distance travelled on the trip, the next bus stop ID, distance to the next bus stop and a few other quantities. We define a *segment* as the road portion of a bus route between two consecutive bus stops.

This dataset was processed through several steps to obtain, for each road segment, a timeseries of average vehicular speed values at every 10 minute block for the entire 90 days, giving timeseries of total length 12960. We used the "next stop" and "distance traveled" information to compute speeds of every vehicle in every segment, which we then averaged for every 10 minute block. There were also a large number of segments with missing values, mostly because of lesser frequency of buses plying on them. However, there were at least 1369 segments with number of valid data points more than 10000 in number out of 12960, amounting to about 8% of the entire number of segments, which were amenable to our analyses.

5 The Model

We use a traditional LSTM as the building block in our architecture, shown in figure 1. There are two layers in the network, with 25 hidden nodes in each LSTM cell. In this model, the input graph is a subgraph of the segment graph.

The input feature vector is simply an augmentation of individual speed values in segment i and its neighbors over the length of history desired (Eq 2). The length of the input feature vector to the network is given by length of history times the size of the neighborhood. An optional graph convolutional operation takes as additional input the adjacency matrix and the weight matrix and computes a graph convolution. Our GC formulation is borrowed from the work of Cui et. al [3], with slight modifications and simplifications. Let us define $A_{i,k}$ as the adjacency matrix of a subgraph containing i and its k-hop neighbors. Let $W_{i,k}$ be the corresponding weight matrix. $W_{i,k}$ has the same dimensions as $A_{i,k}$ and represents distances between segments. Recall that every segment is at unit distance from itself and its immediate neighbors, and that every k-hop neighbor is at distance k. If our subgraph includes only 1-hop neighbors, then every element of $W_{i,k}$ is either 1 or 2. If our subgraph includes 1-hop and 2-hop neighbors, then an element of $W_{i,k}$ belongs to $\{1,2,3,4\}$. Then, the graph convolution is defined as follows.

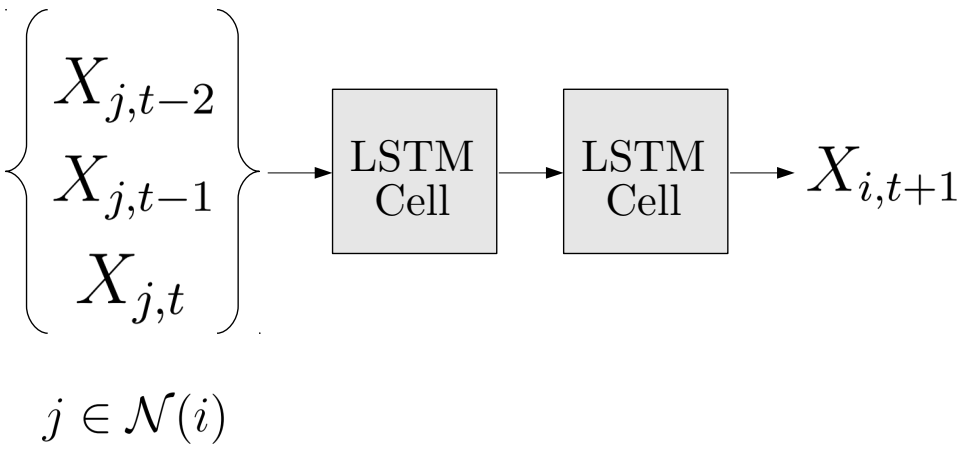


Figure 1: Neural network model

$$GC_k = (W_{i,k} \odot A_{i,k})X_{i,t} \tag{3}$$

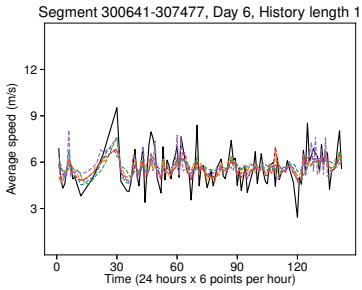
Here, $X_{i,t}$ is the input feature vector at time t and $\odot$ denotes element-wise multiplication. The length of the feature vector at time t depends on the number of neighbors. If m neighbors are used, then the length of feature vector at time t is $m+1$. But this is not the final input to the network if we are using longer history. If our history length is n, then the graph convolution is applied for every feature vector up to n steps back in time. Then the input to the network is the concatenated vector i.e. $[GC_k^1, GC_k^2, \ldots]$, where GC_k^n represents graph convolution of the feature vector n steps back in time.

6 Experiments and Results

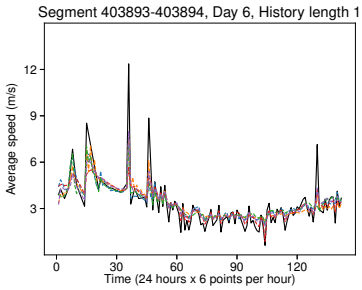
Our implementations were all done using Python, with PyTorch for implementing the neural networks. We experimented with five different configurations – *i)* no hop (using only historical values of the segment in question), *ii)* 1 hop with no GC, *iii)* 2 hops with no GC, *iv)* 1 hop with GC and *v)* 2 hops with GC. The last two are both similar to the second and third respectively, with the difference being that the hardcoded graph convolution (GC) operation (equation 3) is applied to the input feature vector before feeding into the neural network. The network was trained on the first 60 days of the dataset and then tested on the last 30 days. For the sake of brevity, we show our results with only two segments in the city – one in Brooklyn (A) and one in Manhattan (B) – even though we had performed this analysis on several segments spread throughout the city.

Predicting speed at $t+1$

Figures 2 and 3 show predictions for the two segments on a single day in the test period (day 6). Each set of 2 plots show the run sequence plots of actual speeds and predicted speeds for the five different neural network configurations for a given length of history in the input feature vector. The solid lines represent the real data and the dotted lines show the output from the different network configurations.



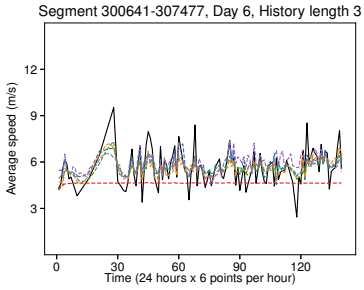
(a) Segment A



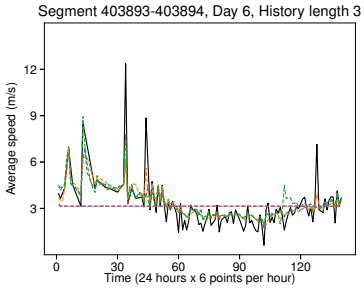
(b) Segment B

Configuration	RMSE (A) (m/s)	RMSE (B) (m/s)
No hop	0.912	0.894
1 hop	0.893	0.855
2 hops	0.927	0.805
1 hop w/ GC	0.847	0.761
2 hops w/ GC	0.950	0.806

Figure 2: Speed prediction at $t+1$ on a sample day in the test period with history length 1 (i.e. using only values at t)



(a) Segment A



(b) Segment B

Configuration	RMSE (A) (m/s)	RMSE (B) (m/s)
No hop	0.815	1.606
1 hop	1.579	0.93
2 hops	0.844	0.902
1 hop with GC	0.867	0.852
2 hops with GC	1.011	1.607

Figure 3: Speed prediction at $t+1$ on a sample day in the test period with history length 3 (i.e. using values at t, $t-1$ and $t-2$)

These results present a glimpse of some key observations that we noted with many other segments and on other days too:

1. A simple neural network using LSTMs is able to achieve impressively good predictive performance with errors averaging close to 1 m/s (MAPE $\sim$ 20-25%)
2. We are able to predict sudden spikes and dips in the speed, which is important for applications such as traffic jam detections and computation of best travel route.
3. We ran each configuration for each segment for 20 iterations, but most of the time the performance capped after 5-6 iterations, only very occasionally going beyond 10. This performance was measured in terms of prediction error on the test data. Thus, peak performance was achieved in less than 10 seconds.

We also highlight other observations, which are possibly attributed to the limitations of a simple LSTM network and/or the noise and non-linearity in the input data.

1. Including more history length in the input feature vector improves prediction performance only up to a certain limit, which we empirically observed to be 2, which translates to 20 minutes. Including more than 2 historical timesteps of speed data does not further reduce prediction and in some cases increases the error.
2. Using information from farther hops either improves or hampers the prediction performance and it seems to be unpredictable. For a simple LSTM network, including information from 1-hop or 2-hop neighbors does not have a consistent effect.
3. The graph convolution operation does not dramatically improve prediction performance and in some cases reduces the performance. We suspect this is due to increase in noise in the feature vector due to convolution operation.

7 Conclusions and Future Work

In this paper, we have presented a new type of dataset for traffic congestion modelling, namely bus mobility data, and have shown how we are able to leverage it fairly easily for traffic state prediction on urban road segments. With an acceptable predictive performance on real segments using an LSTM network, we have shown the feasibility of the use of such a dataset for this task. This is ongoing work and we are currently exploring newer methods or techniques that make better use of traffic states of neighbors and also to be able to predict a few timesteps farther into the future. We believe that a practical system for informing drivers of travel times based on bus mobility is far more feasible proposition in the face of growing privacy concerns surrounding user data and location reporting.

References

1. Avinash Achar, Venkatesh Sarangan, Rohith Regikumar, and Anand Sivasubramaniam. Predicting vehicular travel times by modeling heterogeneous influences between arterial roads. *CoRR*, abs/1711.05767, 2017.
2. Dave Barth. The bright side of sitting in traffic: Crowdsourcing road congestion data, Aug 2009.
3. Zhiyong Cui, Kristian Henrickson, Ruimin Ke, and Yinhai Wang. High-order graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *CoRR*, abs/1802.07007, 2018.
4. Vikramaditya Dangi, Amol Parab, Kshitij Pawar, and SS Rathod. Image processing based intelligent traffic controller. *Undergraduate Academic Research Journal (UARJ)*, 1(1), 2012.
5. Corrado de Fabritiis, Roberto Ragona, and Gaetano Valenti. Traffic estimation and prediction based on real time floating car data. In *2008 11th International IEEE Conference on Intelligent Transportation Systems*. IEEE, oct 2008.
6. Juan C. Herrera, Daniel B. Work, Ryan Herring, Xuegang (Jeff) Ban, Quinn Jacobson, and Alexandre M. Bayen. Evaluation of traffic data obtained via GPS-enabled mobile phones: The mobile century field experiment. *Transportation Research Part C: Emerging Technologies*, 18(4):568–583, aug 2010.

7. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
8. Aude Hofleitner, Ryan Herring, Pieter Abbeel, and Alexandre Bayen. Learning the dynamics of arterial traffic from probe data using a dynamic bayesian network. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1679–1693, dec 2012.
9. Vipin Jain, Ashlesh Sharma, and Lakshminarayanan Subramanian. Road traffic congestion in the developing world. In *Proceedings of the 2nd ACM Symposium on Computing for Development*, page 11. ACM, 2012.
10. Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.
11. 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*, 14(2):871–882, June 2013.
12. Wei Liu, Yu Zheng, Sanjay Chawla, Jing Yuan, and Xing Xie. Discovering spatio-temporal causal interactions in traffic data streams. In *SIGKDD 2011*. KDD 2011, August 2011.
13. NYC MTA. Mta bus time historical data. `http://web.mta.info/developers/MTA-Bus-Time-historical-data.html`.
14. New York City Dept of Transportation (DOT). Nyc dot - data feeds. `http://www.nyc.gov/html/dot/html/about/datafeeds.shtml#realtime`.
15. US Dept of Transportation (DOT). Seattle 20 second freeway data. `https://catalog.data.gov/dataset/seattle-20-second-freeway`.
16. Eleni I. Vlahogianni, Matthew G. Karlaftis, and John C. Golias. Short-term traffic forecasting: Where we are and where we're going. *Transportation Research Part C: Emerging Technologies*, 43:3–19, jun 2014.
17. Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional neural network: A deep learning framework for traffic forecasting. *CoRR*, abs/1709.04875, 2017.
18. Resty Woro Yuniar. Google maps: a lost cause for indonesian drivers, Apr 2018.