Transportation Research Part C 110 (2020) 269–274

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Keep it simple stupid! A non-parametric kernel regression approach to forecast travel speeds☆

Rahul Nair*, Anton Dekusar

IBM Research, Ireland

ARTICLE INFO

Keywords:
Traffic forecasting
Machine learning
Big data

ABSTRACT

The approach taken by the second place winner of the TRANSFOR prediction challenge is presented. The challenge involves forecasting travel speeds on two arterial links in Xi'an City in China for two five hour periods on a single day. Travel speeds are measured from trajectory information on probe vehicles from a fleet of vehicles for a large sub-area of the city. After experimenting with several deep learning methods, we settle on a simple non-parametric kernel regression approach. The method, borrowed from previous work in fixed route transit predictions, formalizes the intuition that in urban systems most failure patterns are recurrent. Our choice is supported by test results where the method outperformed all evaluated neural architectures. The results suggest simple methods are very competitive, particularly considering the high lifecycle cost of deep learning models.

1. Setting

Accurate estimates of near term traffic conditions can aid travelers in planning journeys and help operators in better managing available supply to reduce impacts of congestion. Travel forecasting has been of considerable interest and attracted a slew of methods to generate accurate forecasts of traffic. The recent trend of purely data-driven methods - which use past observations to infer the future have supplanted model-driven approaches - which aim to model the underlying physics of the system. This shift has largely been made possible by a broader availability of data and computational tools.

In this paper, we present our approach to tackle the TRANSFOR prediction challenge. The approach was awarded the second place finish. The challenge involves forecasting travel speeds on two arterial links in Xi'an City in China for two five hour periods on a single day. Travel speeds were inferred from trajectory information on probe vehicles from a fleet of vehicles for a large sub-area of the city. The probe data is from the fleet of DiDi vehicles[1] for a sub area of the city, shown in Fig. 1. Participants were provided probe data for a two month period and given 40 days to generate forecasts for the test data. The relative short challenge period was by design to limit model fine tuning and place emphasis on model approaches.

While we focus on traffic speeds, the broader problem traffic forecasting aims to predict any of state variables - flows, densities or speed. We refer readers to several excellent survey papers on traffic forecasting (Seo et al., 2017; Vlahogianni et al., 2014) and more recent surveys by Lana et al. (2018) and Ermagun and Levinson (2018), the later focusing on spatio-temporal aspects. We limit our review to methods we experimented with. Among data-driven methods, deep learning methods have been gaining traction in the

☆ This article belongs to the Virtual Special Issue on "TRANSFOR 19".

* Corresponding author.
E-mail address: rahul.nair@ie.ibm.com (R. Nair).

[1] https://outreach.didichuxing.com/research/opendata/en/.

https://doi.org/10.1016/j.trc.2019.11.018
Received 30 June 2019; Received in revised form 20 November 2019; Accepted 20 November 2019
Available online 05 December 2019

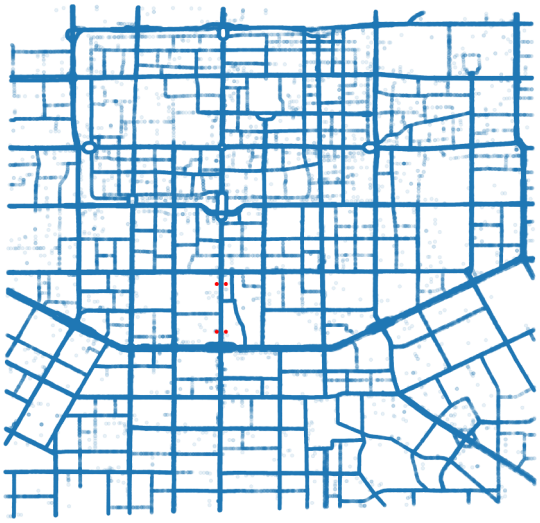


Fig. 1. Probe data for sub area of Xi'an City with study area (red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

community and several papers present results showing better forecast accuracy for short term prediction. (Polson and Sokolov, 2017) present a deep learning model to predict flows and empirically determine a suitable deep learning architecture for forecasting. (Lv et al., 2014) use stacked autoencoders followed by a logistic regression to predict flows and demonstrate better performance for medium and heavy flows. The encoding layers aim to learn traffic features that impact flows directly from the data. Each layer is then trained layer wise. (Horvitz et al., 2012) present a Bayes network based forecasting system which has lead to commercial traffic state estimation services.

Several works have presented Long Short Term Memory (LSTM) architectures for forecasting (Ma et al., 2015). LSTM architectures allow models to have longer term dependence between data points. They have been used to good effect in natural language tasks such dependence can be critical. This dependence structure lends itself to time series forecasts and has been demonstrated in several application domains. Zhao et al. (2017) include origin–destination correlation matrices within the LSTM architecture to model spatial correlation between traffic state variables. In practice, several works have demonstrated that model ensembles perform well for prediction tasks (Fusco et al., 2016; Tan et al., 2009).

Non-parametric approaches, such as the one presented here, have been shown to work well for several mobility processes. Smith et al. (2002) showcase the use of a various similarity measures for a nearest-neighbor model for forecasting traffic flows. Based on two benchmark problems, an ARIMA model performs better than their *k*-nearest neighbor approach. An opposite conclusion was reached in Sinn et al. (2012) albeit in a different setting using a different model specification. For predicting bus arrival times, they show that a kernel regression worked better than other model including nearest neighbor approaches. The method was also demonstrated for use in predictive control for bus bunching (Andres and Nair, 2017). While kernel regression methods work well in capturing recurrent behaviours where delay propogation has been previously observed, it can miss forecasts when there are exceptional congestion conditions. Some works have proposed ensemble approaches, where predictions for different models are combined, to address these limitations (Nair et al., 2019).

After experimenting with several deep learning approaches, primarily LSTMs, SAE, and multi-layered perceptrons, we settle on a simple kernel regression. The method is non-parametric and involves no training. The method formalizes the intuition that in urban systems most failure patterns are recurrent. Our choice is supported by validation results where the method outperformed all evaluated neural architectures. This result (and the second place finish of this approach), suggest that simple methods are very competitive, particularly considering the high lifecycle cost of model specification, training, hyper-parameter tuning, deployment for deep learning approaches. Code used to implement this model is open sourced[2].

2. Data

From the sub region in Fig. 1 we extract data for all probes that crossed the study area. The study area contains two arterial segments, referred to here as *north* and *south*, representing their direction. In addition to these two links, we considered data from several adjacent links that were in/out-bound to the study links. The remainder of the data were not employed by the models. The target variable was computed from the probe data. We computed space mean speeds, which is the harmonic mean of probe vehicle speeds for each 5 min interval. The data had several artefacts, particularly in the early morning hours, with stationary probes and erroneous position data. Simple denoising heuristics were employed to ignore these cases. Specifically, we only consider probes that

[2] https://github.com/rahulnair23/transfor-2019.

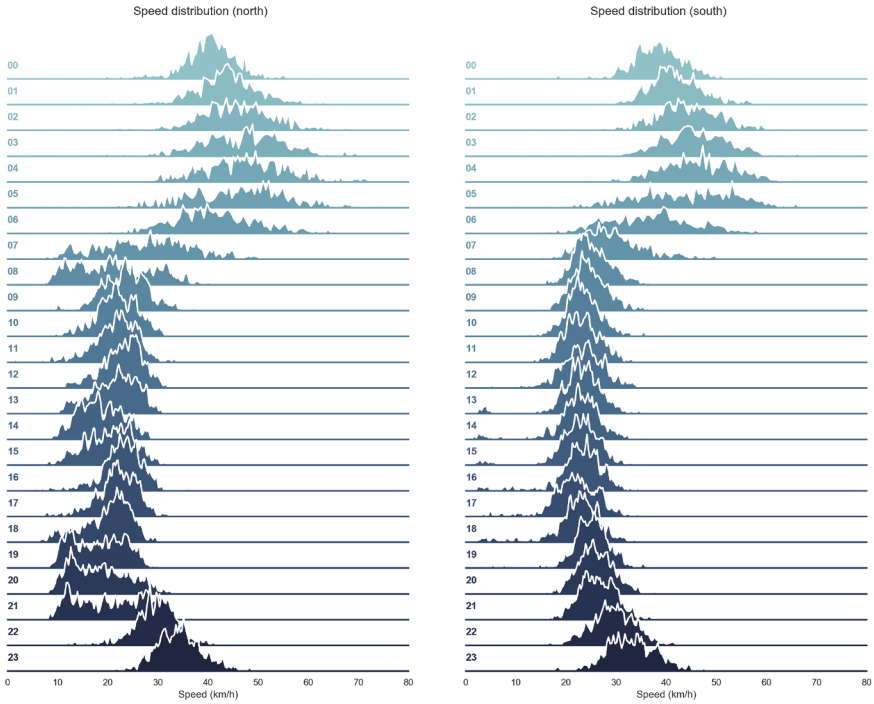
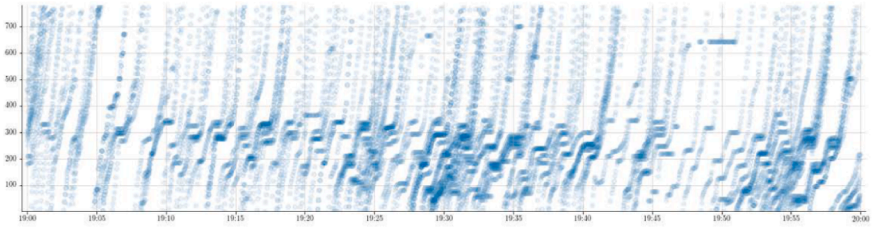


Fig. 2. Speed distributions by hour (kernel density plot).

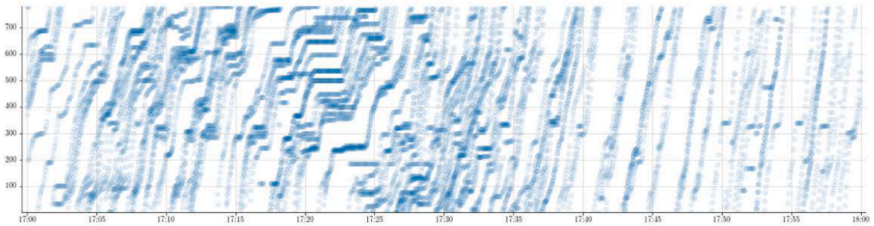
travelled the entire length of the study segment. This excludes vehicles that parked or made turning manoeuvres.

Fig. 2 shows the distributions by hour of day. The north bound shows higher variance than the south where the speed distributions are relatively more compact.

We considered two variants: a univariate model for speed and a multivariate model based on speeds on adjacent links and additional derived features. The traffic dynamics on the study links are governed by spill backs and queuing. Fig. 3 shows space time plots for two different time periods, both in the north bound direction. The link has a mid-block traffic control device of undetermined phase length.

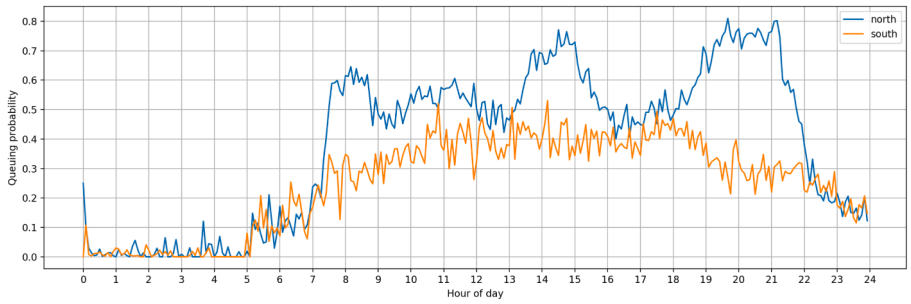


(a) Example of queue due to signal control

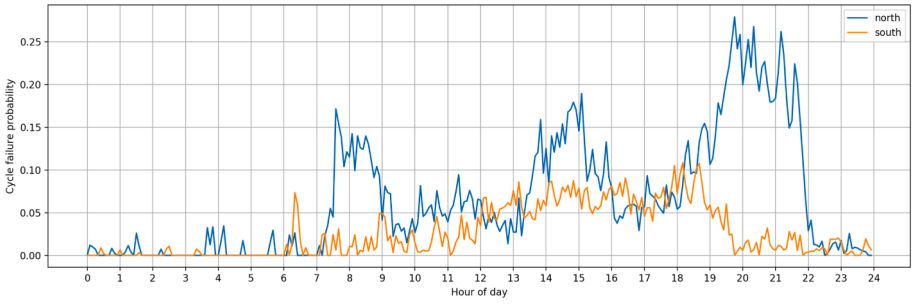


(b) Example of spillbacks

Fig. 3. Space–time plots of probe trajectories.



(a) Probability of vehicle queuing (per 5 minute period)



(b) Probability of vehicle experiencing cycle failure (per 5 minute period)

Fig. 4. Extracted features based queueing/cycle failure probabilities.

We experimented with several features based on queuing of vehicles. The first was the probability of a probe queuing within a five minute interval. The second was the probability of a probe experiencing cycle failures. A cycle failure occurs when a queuing vehicle is not served by the first green phase it sees. We additionally consider expected queuing time. While these are highly correlated with the target variable, they are unavailable at test time and so one must use some expected measure based on time. The expected value of these features, shown in Fig. 4, turned out not to be discriminative and did not improve forecast quality.

3. Models

Given a partial set of observations for travel speeds and a historical set of observations, we seek to predict travel speeds on a link. Formally, we cast this as a sequence forecasting problem. Denote $\boldsymbol{s_T} = \{s_t | t \in T\}$ as the set of known speeds on a link and $\hat{\boldsymbol{s}} = \{\hat{s}_t | t \in T'\}$ the set of speeds that need to be predicted. We assume a historical set of known time series $\boldsymbol{s^h}$ for $h = 1, ...c, H$. We seek to build a model M: $\mathbb{R}^{|T|} \rightarrow \mathbb{R}^{|T'|}$ such that a forecast $\hat{\boldsymbol{s}} = M(\boldsymbol{s^h}, \boldsymbol{s_T})$ can be generated.

Kernel regression: For the non-parametric kernel regression, we use the kernel function to compute similarity between the current (partial) speed data and the historical reference set. Specifically, as in Sinn et al. (2012), we employ a Gaussian kernel function $kernel(x, y) = \exp(-||x - y||^2/b)$, where b is known as the bandwidth parameter which controls how the weights are spread in the reference set. A large value of b implies that all time series in the historical set has equal importance, while a smaller b places all the weight in fewer time series that are most similar. In this application, the kernel function is computed from the known sequence of data, so the kernel function is written as

$$\text{kernel}(\boldsymbol{s_T}, \boldsymbol{s_T^h}) = \exp\left(-\frac{1}{b}\sum_{t\in T}\frac{(s_t - s_t^h)^2}{\sigma_t^2}\right), \tag{1}$$

where, σ_t^2 is the empirical variance at t. With this kernel similarity, predictions can be made by computing the kernel weighted average as

$$\hat{s}_t = \frac{1}{\sum_{h\in H}\text{kernel}(\boldsymbol{s_T}, \boldsymbol{s_T^h})}\sum_{h\in H}\text{kernel}(\boldsymbol{s_T}, \boldsymbol{s_T^h})s_t^h. \tag{2}$$

LSTM: An LSTM model consists of an LSTM unit that contains a cell, an input, output, and forget gates. The cell contains state information that, between units can pass the input signal. This input signal can be changed by the three gates. A forget gate is typically a sigmoid activation function that can remove information. The input gate regulates what part of the information needs to

be updated within the cell state. The output gate then structures the cell state to output a value to the next unit, typically using a tanh activation function. Formally, given x input, model parameters U, V, b and h activation functions, the model is expressed as

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \tag{3}$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{4}$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{5}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \tag{6}$$

$$h_t = o_t \circ \sigma_h(c_t) \tag{7}$$

We experimented with two variants of the LSTM, one where we get the entire sequence prediction, and another where we only consider one time step ahead and then using the prediction to generate subsequent forecasts.

Autoencoders: Autoencoders are semi-supervised model which aim to generated a lower/higher dimensional representation of the input sequence through a number of layers (the encoder) and the use a set of layers to recover the original input back (the decoder). Autoencoders can be viewed as lossy compression of input sequences. For this challenge, we can aim to recover the unobserved part of the sequence (the desired forecast) based on the partially available forecast.

Lastly, we also tried machine learning models, such as a gradient boosted tree to generate the forecast. All deep learning models (LSTM variants, MLP and the autoencoder) were implemented in Keras and trained using the IBM Deep Learning as a Service (DLaaS) using two Tesla v100 GPUs.

4. Results

Since the challenge is based on a test set not revealed during model development, we defined an internal validation set for model design. All data before a cut off was used for training and we used the remaining four days for testing. RMSEs were evaluated for every 5 min period within the time periods (6 am - 11 am, and 4 pm - 9 pm), leading to 480 forecasts for each model. To generate the final submission, we then retrained the chosen model on all data. A naive baseline based on hourly average speeds was also computed.

We tested several variants of Long Short Term Memory (LSTM) networks. We found models based on lag variables within 30 min to perform best. For LSTM sequence forecasts (where the output is a sequence) the training was rather unstable and failed to learn any parameters on several runs. The number of training samples was likely too small. For the kernel regression, we additionally performed a grid search to determine the optimal bandwidth for each link.

Table 1 shows the validation results for the various models tested.

With these results, where the kernel regression based on weekday data only performed the best, we investigated the test day data (Fig. 5). The partial speed data (before 6am) showed deviation from our space mean speeds by an average of 1.4 km/h. We therefore removed this bias from our submission by simply subtracting this from our forecast to account for this difference. The published RMSE by the organizers for the test day show an RMSE of 4.517.

5. Conclusions

We conclude that simple methods are often the most effective. A naive LSTM model (”out-of-the-box”) does not appear to be well suited to short-term time series forecasting. More complex architectures need to be investigated for the purpose of traffic forecasting.

The proposed kernel method is simpler (involves no training) however has some drawbacks. The method is likely to miss exceptional delays, particularly for longer time horizons. In practice, ensemble approaches may be better suited for field applications.

No method, in our investigation, yielded a truly discriminating factor. We limited our feature engineering to queuing and cycle failure probabilities, both of which turned out to be ineffective. A two-staged method, first estimating physical quantities via

Table 1
Summary of model performance on validation set (n = 480).

Model	RMSE	
	North	South
Naive baseline	10.15	9.82
LSTM multivariate single step	6.61	6.68
LSTM univariate single step	6.55	4.30
LSTM univariate sequence	6.59	–
Autoencoder	6.28	–
Gradient boosting trees	6.52	4.97
Multi Layer Perceptron	5.36	5.07
Non-parametric kernel regression	4.05	3.33
Kernel regression - weekday data only	3.68	3.32

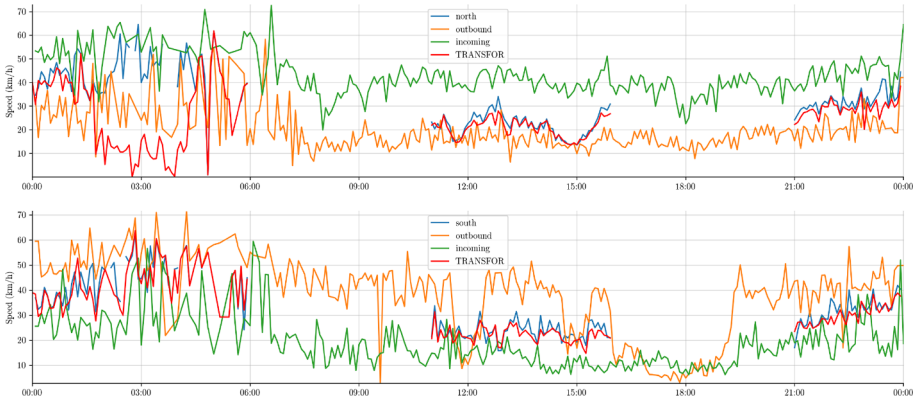


Fig. 5. Test day data.

simulation and the forecasting using these values may be helpful.

Deep learning methods are akin to fishing in the computational sea. Given the limited duration of the challenge, we didn't catch any fish.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.trc.2019.11.018.

References

Andres, M., Nair, R., 2017. A predictive-control framework to address bus bunching. Transport. Res. Part B: Methodol. 104, 123–148.
Ermagun, A., Levinson, D., 2018. Spatiotemporal traffic forecasting: review and proposed directions. Transport Rev. 38, 786–814.
Fusco, G., Colombaroni, C., Isaenko, N., 2016. Short-term speed predictions exploiting big data on large urban road networks. Transport. Res. Part C: Emerg. Technol. 73, 183–201.
Horvitz, E.J., Apacible, J., Sarin, R., Liao, L., 2012. Prediction, expectation, and surprise: Methods, designs, and study of a deployed traffic forecasting service. arXiv preprint arXiv:1207.1352.
Lana, I., Del Ser, J., Velez, M., Vlahogianni, E.I., 2018. Road traffic forecasting: recent advances and new challenges. IEEE Intell. Transp. Syst. Mag. 10, 93–109.
Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.Y., 2014. Traffic flow prediction with big data: a deep learning approach. IEEE Trans. Intell. Transp. Syst. 16, 865–873.
Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y., 2015. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transport. Res. Part C: Emerg. Technol. 54, 187–197.
Nair, R., Hoang, T.L., Laumanns, M., Chen, B., Cogill, R., Szabó, J., Walter, T., 2019. An ensemble prediction model for train delays. Transport. Res. Part C: Emerg. Technol. 104, 196–209.
Polson, N.G., Sokolov, V.O., 2017. Deep learning for short-term traffic flow prediction. Transport. Res. Part C: Emerg. Technol. 79, 1–17.
Seo, T., Bayen, A.M., Kusakabe, T., Asakura, Y., 2017. Traffic state estimation on highway: A comprehensive survey. Annual Rev. Control 43, 128–151.
Sinn, M., Yoon, J.W., Calabrese, F., Bouillet, E., 2012. Predicting arrival times of buses using real-time gps measurements. In: Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on, IEEE. pp. 1227–1232.
Smith, B.L., Williams, B.M., Oswald, R.K., 2002. Comparison of parametric and nonparametric models for traffic flow forecasting. Transport. Res. Part C: Emerg. Technol. 10, 303–321.
Tan, M.C., Wong, S.C., Xu, J.M., Guan, Z.R., Zhang, P., 2009. An aggregation approach to short-term traffic flow prediction. IEEE Trans. Intell. Transp. Syst. 10, 60–69.
Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C., 2014. Short-term traffic forecasting: Where we are and where we're going. Transport. Res. Part C: Emerg. Technol. 43, 3–19.
Zhao, Z., Chen, W., Wu, X., Chen, P.C., Liu, J., 2017. Lstm network: a deep learning approach for short-term traffic forecast. IET Intel. Transport Syst. 11, 68–75.