# MIT gated parking prediction using parking reader records

Tianyu Su, Zhuangyuan Fan, MCP 20', MIT DUSP

*Full report for DL4T*

## 1. Introduction

Boston region has been challenged by long commute time and increased parking demands, and in recent years, MIT campus saw a rising issue of providing enough subsidized parking for faculty and employee. In order to maintain a sustainable campus, the school office of sustainability initiated various travel demand management (TDM) programs such as Access MIT[1] and carpooling[2] to encourage travel mode shiftings among MIT faculties, employee, and students. While most of these programs and initiatives are adopted across the whole institute, the 2019 MIT Commuting Survey shows that individual travel mode choices and their responses towards the same program vary significantly. This phenomenon provides us with a chance to consider mirco-targeting strategies designed for different subgroups.

In order to design an actionable "tailored" commute-related recommendation/program, it is crucial to understand the current spatial temporal pattern of campus parking and its relationship to the regional context. In this project, we focus on predicting the number of daily gated parking coming from each census tract in Parking Year 2017-2018 (Sep. 16, 2016 - Sep. 15, 2018), as well as the overall number of daily parking of all parking permit holders. We trained a dual-stage attention-based recurrent neural network (DA-RNN) model with the campus parking history from each census tract, and exogenous variables such as weather, holidays, transit health, and use the model to predict the daily number of parking coming from each census tract. We also conduct a comparison of the baseline ARIMA model and our DA-RNN model using the same variables.

## 2. Research questions

1.On a particular day, how many MIT parking permit holders will drive from a selected census tract and park at MIT gated parking? As well the overall number of MIT gated parking.

2. How could predictions of the number of parking at MIT campus help better manage the campus parking resources and potential carpooling/vanpooling programs?

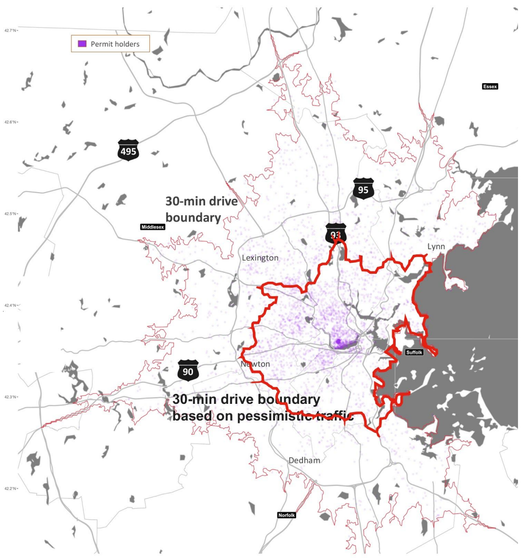
---

[1] https://sustainability.mit.edu/access-mit
[2] https://web.mit.edu/facilities/transportation/ride_sharing.html

## 3. Data

1) parking history:

   The parking history dataset includes the number of daily MIT gated parking coming from each census tract, extracted from a sample of individual commute data. We sample the individual commute data based on the overlapping between the commute dataset and the HR dataset, which include the address information on census tract level. Then, we aggregate the individual data into census tracts based on the GEOID. Figure 2 suggests an average daily gated parking from each census tract based on our sample data.



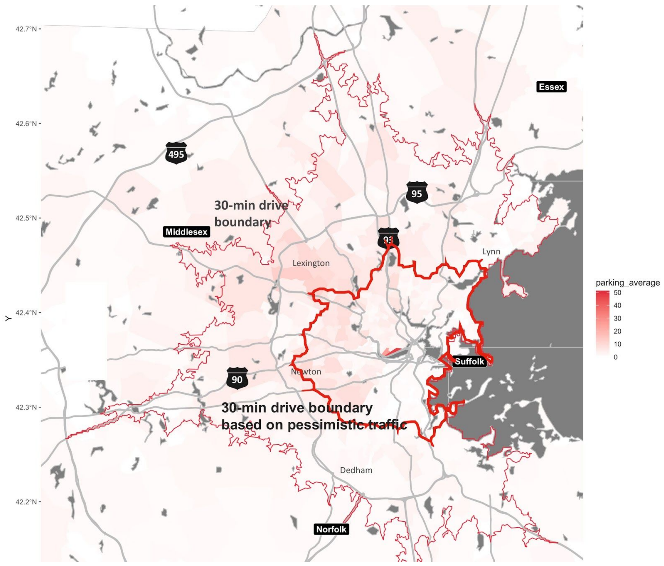




*Figure 1. Spatial distribution of MIT parking permit holders in PY 2018*

*Figure 2. Average number of MIT gated parking from each census tract PY17 - 18*

2) day of the week:

   Since we notice a significant difference in the gated parking patterns across the days of the week, we encode the day of the week as a dummy variable.

3) weather data:

   The daily temperature, precipitation, snow depth, and weather type information is extracted from the climate dataset from the National Centers for Environmental Information and is shifted by one day.

4) transit health data:

   We get daily disruption data of MBTA Red Line, other subway lines, and Commuter Rail from VHB. Red Line is likely to have the most direct impact to MIT employee decisions, but we included all to give some more nuance to the nature and severity of disruptions. The metrics were validated against known one-day and extended incidents.

5) national holiday data:

   The national holiday data is downloaded from a GitHub repo and is shifted by one day.

6) class days or vacation:
   We collect the start and end dates of semesters, summer vacations, spring breaks, and winter breaks, then include the information as a dummy variable.
7) MIT event data:
   We collect the date of MIT events and turn it into a dummy variable.

## 4. Methodology

4.1 Deep Learning Model

This project mainly used a dual-stage attention-based recurrent neural network (DA-RNN) for time series prediction(Qin et al., 2017) and a simulated Bayesian Neural Network(Zhu and Laptev 2017)
for prediction.

We chose these two models due to two reasons: 1) Encoder-Decoder lstm models have shown success in machine translation, however, when the series gets longer, the performance of this system deteriorate rapidly. Thus, an attention mechanism could adaptively extract relevant driving features. 2) Number of parking could subject to many uncertain circumstances, the traditional RNN that only produces single prediction could not capture the uncertainties. Thus a Bayesian Neural Network could estimate the uncertainties and produce a distribution of the prediction.

4.2 BNN Architecture

The BNN architecture we used in this project is simulated by adding two dropout layers in between two LSTM layers. Since randomly turning on and off neurons is similar to performing a Bernoulli sampling, if the model was trained multiple times, it will produce a distribution of predictions (Figure 3a).

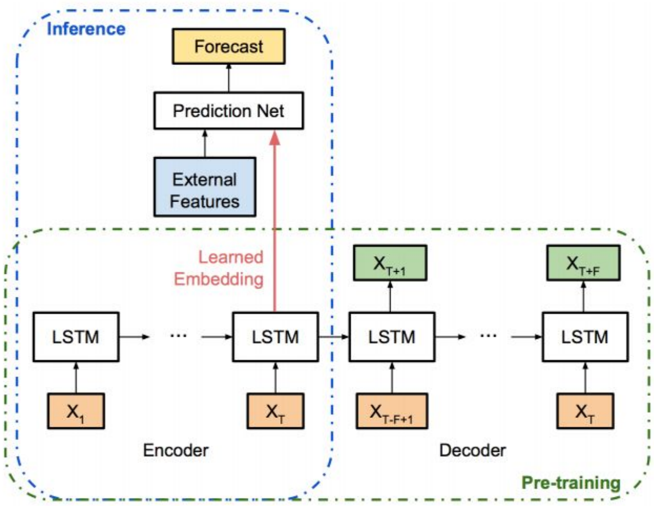


*Figure 3a. The Uber LSTM forecasting architecture (Zhu & Laptev, 2017)*

## 4.3 DA-RNN Architecture

In this project, we will mainly break down the steps of DA-RNN architecture. The DA-RNN model includes two Long Short-Term Memory (LSTM) networks in the encoder and decoder.

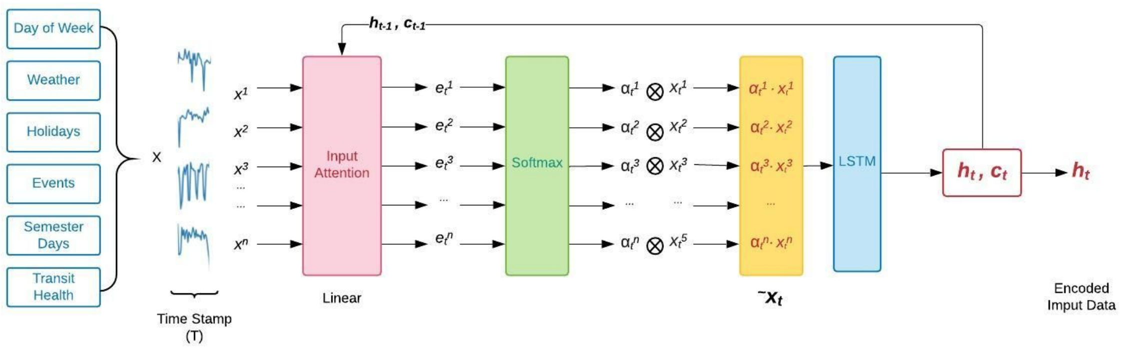



*Figure 3b. Graphical Illustration of Input Attention Encoder*
*The input attention mechanism computes the attention weights $\alpha^n_t$ for multiple exogenous series $\{x^1, x^2, \ldots, x^T\}$ into the encoder LSTM series, and output the hidden states as encoded input data*

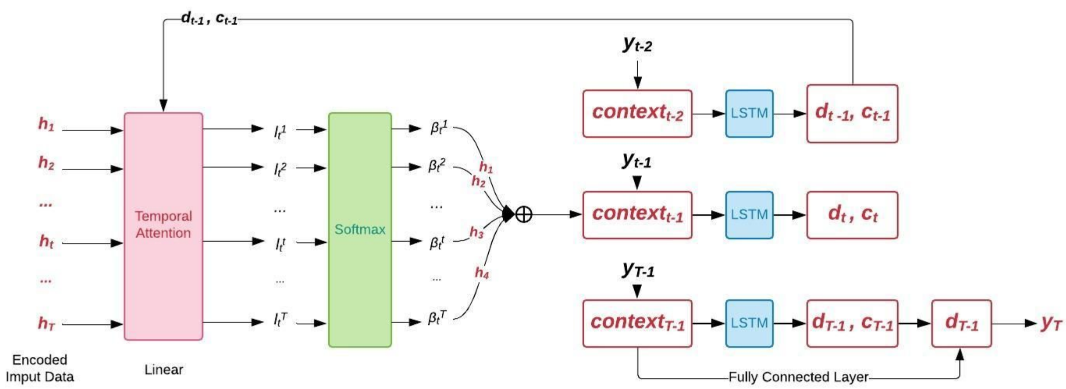



*Figure 3c. Graphical Illustration of Temporal Attention Decoder*
*The temporal attention system computes the attention weights based on the previous decoder hidden state dt-1 and represents the input information as a weighted sum of the encoded input data across all the time steps. The context vector is used as an input to the decoder LSTM unit. The output $d_{T-1}$ of the last LSTM unit is connected with the context vector by a fully connected layer. The $y_T$ of the fully connected layer is the predicted result.*

**Encoder:**

The encoder here is an RNN that encodes the input (here are exogenous variables such as weather, transit) into a feature representation in machine translation. For this project, we have extract 700 days of data, thus our input sequence $\mathbf{X} = (x_1, x_2, \ldots, x_T)$ with $\mathbf{T} = 700$, and $\mathbf{x}_t \in \mathbf{R}^n$ , where n is the number of exogenous driving series (n = 16). The encoder was applied to learn a mapping from **x**t to **h**t:

$$\mathbf{h}_t = f_1(\mathbf{h}_{t-1}, \mathbf{x}_t)$$

Here $f_1$ is an LSTM unit to capture long-term dependencies.

We used an input attention based encoder that selects relevant exogenous series with attention weights(Qin et al. 2017) rather than treating all the input driving series equally. The exogenous series are vectors whose values change with time (*Figure 3a*).

$$\tilde{\mathbf{x}}_t = (\alpha^1 t \mathbf{x} t^1, \alpha^2 t \mathbf{x}^2 t, \cdots, \alpha^n t\ \mathbf{x}^n t)$$

(α is the attention weight measuring the importance of the k-th input feature at time t. )
Then the hidden state at time *t* will be updated as

$$\mathbf{h}_t = f_1(\mathbf{h}_{t-1}, \tilde{\mathbf{x}}_t)$$

**Decoder**:
The model prediction goal is to predict the parking demand at t+1 given the data till t for the entire MIT campus and each census tract. Thus another LSTM-based recurrent neural network was used to decode the encoded input information.

**Training Procedure**
We used the Adam optimizer (Kingma and Ba, 2014) to train the model. The size of the minibatch is 64. The learning rate starts from 0.01 and is reduced by 30% after each 100 iterations. The parameters were learned by standard back propagation with mean squared error as the loss function.

## 5. Experiments

In this section, we will introduce the parameter settings of DA-RNN and the evaluation metrics.

### 5.1 Setup Dataset

For the goal of this research, we test the DA-RNN model on two scopes of data.

| **Target** | **driving series (exogenous variables)** | **training size** | **test size** |
|---|---|---|---|
| Total number of parking on a given day at MIT campus | 17[3] | 489 | 211 |

[3] See Appendix I for dataset

| | | | |
|---|---|---|---|
| Number of parking on MIT campus from each census tract | 17 | 489 | 211 |

*Table 1. Dataset Setup Preview*

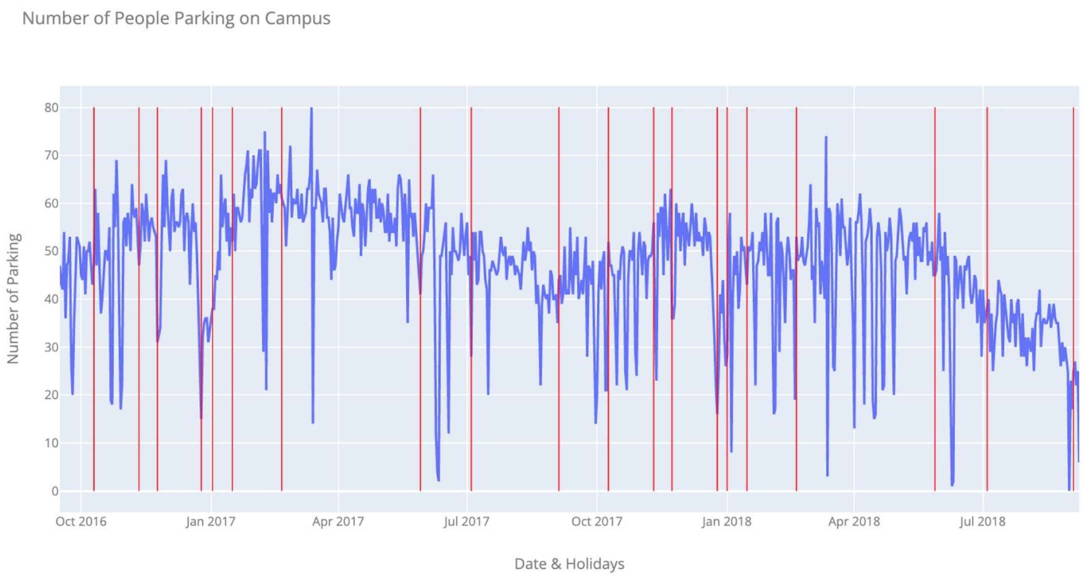



*Figure 4a. Number of people parking at MIT campus from census tract 25017353102 and major holidays*

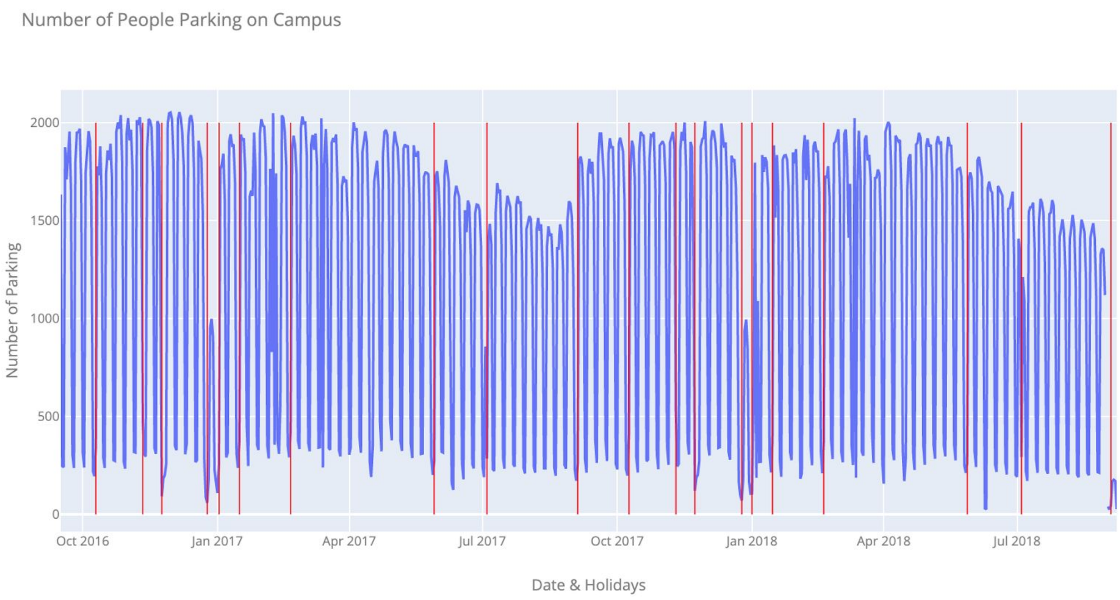



*Figure 4b. Total number of people parking at MIT campus and major holidays*

## 5.2 Parameter Experiment

There are three hyper parameters in the model: the number of time steps in the window $T$, the size of hidden states ($m$) for the encoder, and the size of hidden states ($p$) for the decoder. Since parking sequence data is very sensitive to weekday and weekend, we set conducted a search for the best window over $T \in \{7, 14, 21, 28\}$(*Figure 5*). The one (T = 21) that achieves the best performance for the validation set was used for the test set. And for the size of encoder and decoder, we use $m = p$ for simplicity. In the experiment, we assumed that the size of encoder and decoder are independent of the input parameters and search over $m \in \{16, 32, 64, 128\}$,It turns out that m = 64, 128 outperformed than others, however, the model is not stable enough to tell which one works better. (*Figure 6*). But for the purpose of this research, we will use the conventional hidden states size 64.

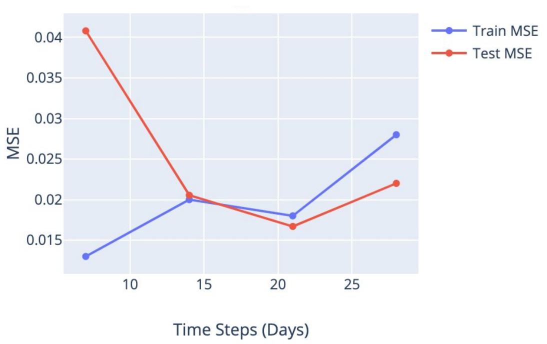



*Figure 5. Loss over different Time Step (T) selection*

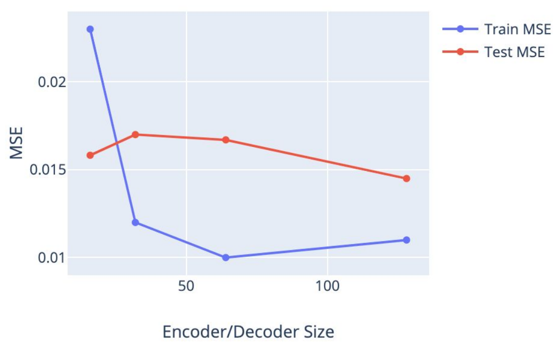



*Figure 6. Loss over hidden states (cell) size (m, p)*

To measure the effectiveness of the DA-RNN model and compare it with ARIMA, we use root mean squared error (RMSE) (Plutowski et al., 1996) and mean absolute error (MAE) .

5.3 Results

5.3.1 Time-series prediction at the whole campus level

After 400 epochs, the model was able to capture most of the character of the time-series data. As we could tell from the fitting graph (Figure 8), the model is very good at capturing shocks such as winter break and trend such as the end of a semester.

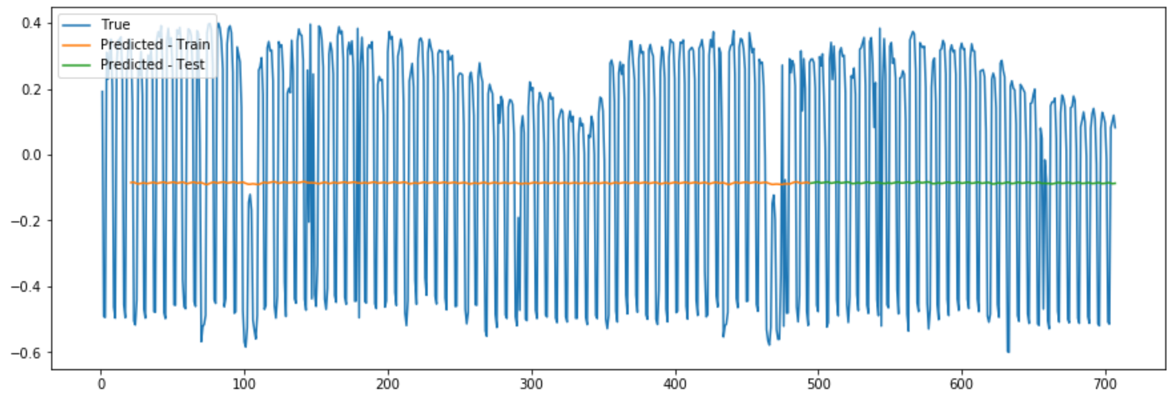



*Figure 7. Model training at 0 epoch*

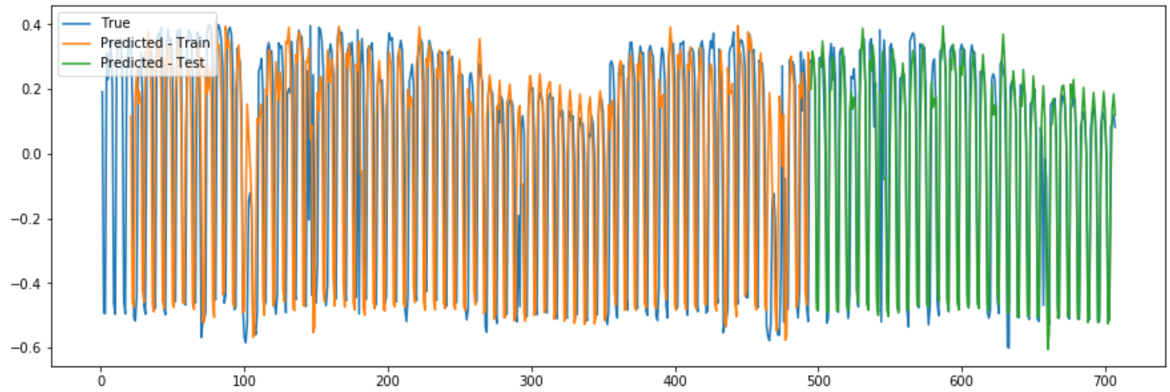



*Figure 8. Model training at 400 epoch*

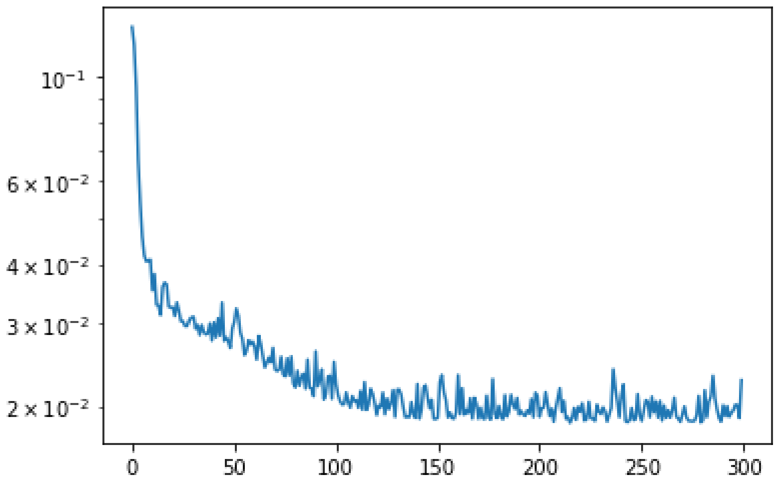



*Figure 9. Model training loss progress as number of epoch goes up*

### 5.3.2 Model Comparison Between ARIMA, DA-RNN and BNN

The DA-RNN model is quite sensitive to the learning rate adjustment, as the model tends to overfit very quickly if the learning rate is too high. While ARIMA is relatively stable.

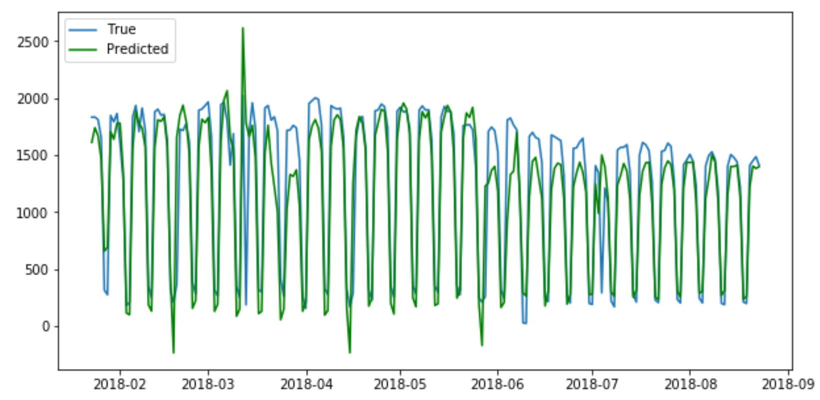
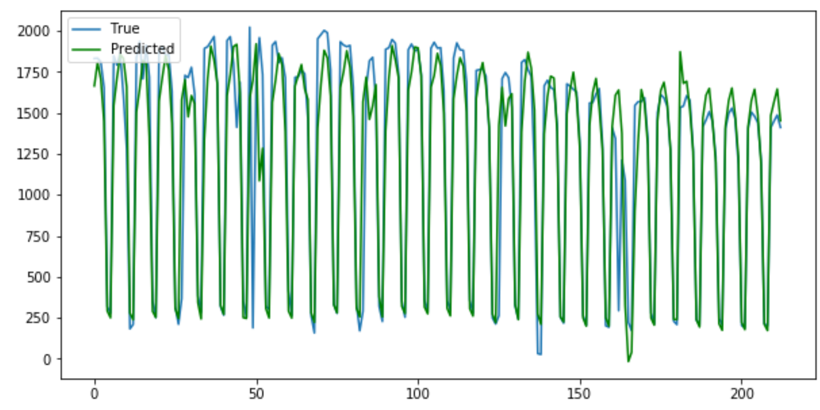
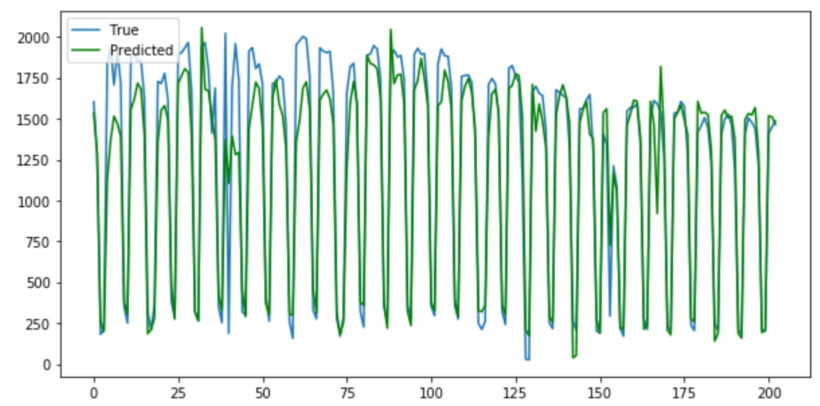
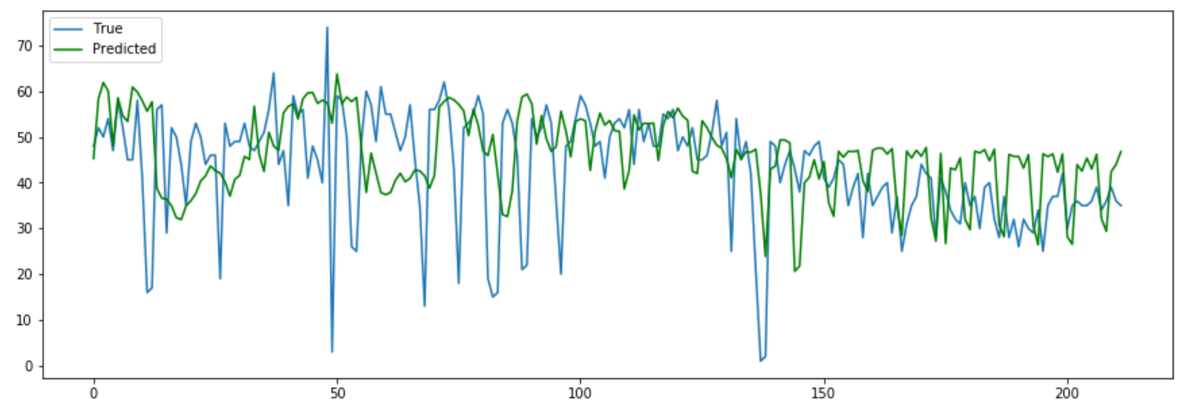
<table>
<tr><td>ARIMA</td><td>
</td><td>MAPE:<br>59.82%<br>MSE:<br>0.1644<br>MAE:<br>0.1108</td></tr>
<tr><td>DA-RNN</td><td>
</td><td>MAPE:<br>41.91%<br>MSE:<br>0.015823<br>MAE:<br>0.0677<br>T=21<br>Encoder Size:<br>64</td></tr>
<tr><td>BNN</td><td>
</td><td>MAPE: 60.39<br>%<br>SMAPE: 16.06<br>%<br>RMSE:<br>204.25371<br>MAE :<br>139.82098<br>(RMSE:<br>0.45162192<br>MAE:<br>0.3454574<br>scaled)</td></tr>
</table>

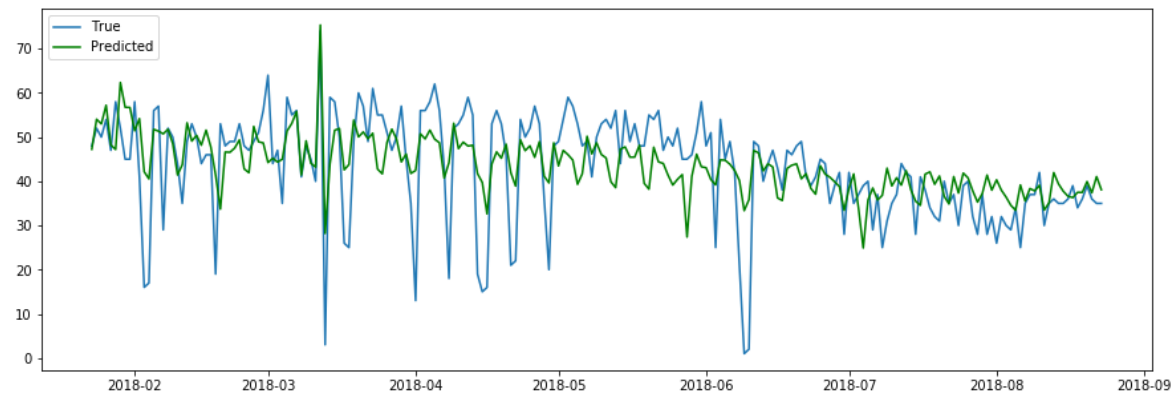
*Table 3. Comparison between the DA-RNN and ARIMA at whole campus level*

### 5.3.3 Time-series prediction at the individual census tract level

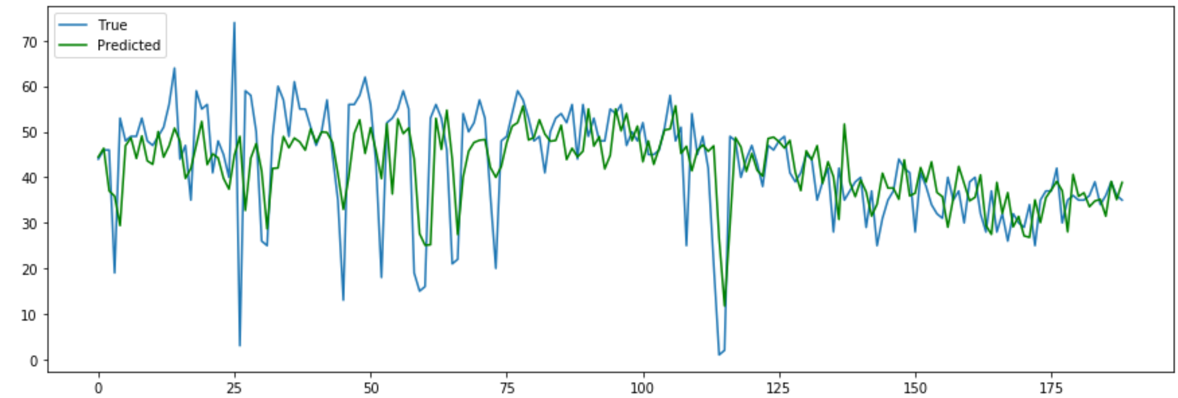
For census tract level prediction, we also conduct a comparison between DA-RNN model, DNN and the ARIMA model. We found all three models performed worse than the MIT campus prediction. We assume the reason could be for the whole campus level, given the cap of total parking spaces, the pattern is relatively stable, while for one independent census tract, without fitting the parking pattern of campus as a variable, the pattern is harder to learn from. However, the BNN model was able to capture the pattern much better than the other two.




DA-RNN
MAPE: 290.61 %
SMAPE: 25.83 %
RMSE: 13.592277108285394
MAE: 10.152618640617018




Seasonal ARIMA model
MAPE: 139.89 %
SMAPE: 19.22 %
RMSE: 9.66945101719838
MAE: 7.061715331593507




BNN (Bayesian Neural Networks)
MAPE: 105.40 %
SMAPE: 18.81 %
RMSE: 9.683583
MAE: 6.831998
(RMSE: 0.1780956
MAE: 0.13968109 scaled results)

Table 3. *Comparison between the DA-RNN and ARIMA at Census Tract Level*

## 6. Reflections, limitations, and next steps

6.1 Reflections

As the vehicles and parking facilities are getting more connected today, street-scale parking management has seen a great progress, for example streetlight sensor systems like CityIQ™ developed by GE Current and implemented by San Diego and other cities. Also, regulations like parking time limits and restrictions based on license plate numbers enable city level parking policies. However, how could we manage parking activities on an urban district level, for example campus-scale, by taking advantage of both fine-grained parking record data and the power of policy-making.

Based on our multi-level prediction, we are proposing some possible scenarios in campus parking management:

- Prediction on overall number: Coordinate parking resources between MIT and local parking facilities.
- Prediction on parking area level: Manage subsidies for parking and build parking guidance and information (PGI) systems.
- Prediction on census tract level: Uncover the possibilities of carpooling or vanpooling programs.

6.2 Limitations and next steps

Limited by the duration of the time-series data and the nature of the models, we encounter the following limitations:

1. The tradeoff between dataset size and data quality, we have cleaned data of PY 16, which we find pretty spotty. Since the LSTM model is sensitive to the temporal pattern, we decide to only use PY 17 and PY 18. However, PY 19 data is being cleaned by MITOS, which may improve the performance of our model.
2. Model performance on census tract level and parking area level need more investigations and optimizations.
3. The nature of the model makes it less interpretable than classical statistical models.

And in the next steps, we hope we could take in knowledge from the behavior science field and zoom in to some specific time periods. Also, we will review more cases for potential applications of campus level parking number prediction, and think about the possibilities from the perspective of parking managers (the under-prediction/over-prediction problem). We will be more focused on the overall number predictions and parking area level predictions in our future work.

References:

Cantillo, V., Ortúzar, J. D. D., & Williams, H. C. (2007). Modeling discrete choices in the presence of inertia and serial correlation. *Transportation Science*, *41*(2), 195-205.

Hübner, R., Steinhauser, M., & Lehle, C. (2010). A dual-stage two-phase model of selective attention. *Psychological review, 117(3)*, 759.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., & Cottrell, G. (2017). A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971.*

Rajabioun, T., & Ioannou, P. A. (2015). On-street and off-street parking availability prediction using multivariate spatiotemporal models. *IEEE Transactions on Intelligent Transportation Systems*, *16*(5), 2913-2924.

Vlahogianni, E. I., Kepaptsoglou, K., Tsetsos, V., & Karlaftis, M. G. (2016). A real-time parking prediction system for smart cities. *Journal of Intelligent Transportation Systems*, *20*(2), 192-204.

Zhao, Z., Koutsopoulos, H. N., & Zhao, J. (2018). Individual mobility prediction using transit smart card data. *Transportation research part C: emerging technologies*, *89*, 19-34.

Zhu, Lingxue, and Nikolay Laptev. 2017. "Deep and Confident Prediction for Time Series at Uber." *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, November, 103–10. https://doi.org/10.1109/ICDMW.2017.19.

## Appendix 1 - Data Structure

| date | 6037138000 | … | 53033004000 | Mon | Tue | Wed | Thu | Fri | Sat | Sun |
|---|---|---|---|---|---|---|---|---|---|---|
| 2016/9/16 | 1 | … | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 2016/9/17 | 0 | … | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 2016/9/18 | 0 | … | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 2016/9/19 | 1 | … | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2016/9/20 | 0 | … | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 2016/9/21 | 1 | … | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| date | ... | class | iapclass | event | RedLine | Other T Lines | Commuter Rail | next_is_holiday * | next_prcp * | next_snow * | next_temp * | parking_all |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2016/9/16 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 65 | 1633 |
| 2016/9/17 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 73 | 247 |
| 2016/9/18 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.33 | 0 | 72 | 241 |
| 2016/9/19 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 70 | 1875 |
| 2016/9/20 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 75 | 1710 |
| 2016/9/21 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 71 | 1848 |

https://towardsdatascience.com/time-series-in-python-part-3-forecasting-taxi-trips-with-lstms-277afd4f811