Machine Learning for Traffic Flow Prediction in Urban Areas

Yulin Huang[1] and Genggeng Zhou[2]

[1]yuh245@stanford.edu
[2]g9zhou@stanford.edu

June 2, 2021

1 Introduction

As the urbanization and industrialization improve the life quality of humans, more and more people tend to purchase their own cars instead of using public transportation. The increasing number of vehicles intensifies the traffic congestion in the world, especially in the urban areas. However, it seems that there are trends when traffic increases tremendously during a week and even in a specific period in a day.
The goal of this project is to simulate authentic traffic situation in urban areas. With proper prediction on traffic flows, people can adjust and rearrange their schedules in advance to avoid traffic congestion that leads to delays for important events.
To obtain a better model and prediction on the traffic, we will remain heavily on neural networks to train with a large number of examples. The input to our algorithm is a label indicating the traffic in a specific time in a day in the range between 0 and 4 with 0 meaning low traffic and 4 meaning significant traffic congestion.

2 Related Work

'A Neural Network Based Traffic-Flow Prediction Model'[1]is highly related to what we are interested in. They applied artificial neural networks and backward propagation to predict traffic flow at junctions in Istanbul based on historical data of traffic flow volume. However, their model is trained specifically to predict traffic flows in the next 5-minute or next hour.
'An Improved Fuzzy Neural Network for Traffic Speed Prediction Considering Periodic Characteristic'[3] used a different approach called Evolving Fuzzy Neural Network(EFNN) to train their model. It is different from the traditional neural network method. They first use a K-means method to partition inputs into different clusters and then use Gaussian fuzzy membership function to measure degree of membership of each input. The clusters and Gaussian membership function are iteratively updated as input size increases, after which a weighted recursive least square estimator is used to optimize parameters of the function. In addition, a trigonometric regression function is also used to capture periodic trend in traffic data. Based on their results, EFNN performs better than artificial neural networks and other traditional models.
The paper 'DeepTrend: A Deep Hierarchical Neural Network for Traffic Flow Prediction'[5] is an

interesting paper proposing an algorithm to extract the time-variant trend and residues in traffic flow and make better predictions using a combination of extraction layer and prediction layer, which is an Long Short-Term Memory layer. Their approach requires pre-training each layer separately, and then fine tuning each layer in the entire neural network. Such method was proven to greatly increase the accuracy of traffic flow predictions over traditional models. However, their method only applies to temporal data, and does not include spatial dependency between different locations.

3 Dataset and Features

The model is trained using data from Outscraper[4]. The raw data consists of the following information: name of the road, coordinates of the start and end point of a road, the total length of the road in meters, the total time used to travel through the road in minute, and the date and time the data was recorded. Data are processed through the following procedure:

1. Assign a unique integer to each road.
2. Assign integer 1-7 to the corresponding days in a week, and 0-23 to corresponding times in a day.
3. Assign integer to each road indicating the level of traffic congestion with 0 indicating perfectly clear road and 4 indicating extremely bad traffic. The level of traffic congestion is labelled by first calculating the average speed of travel, then normalize the speeds with average speed and standard deviation, and lastly within 2 standard deviation of the maximum speed are assigned level 0, data within increasing standard deviations are assigned at increasing levels. Those levels are then converted to one-hot label vectors for classification.

Therefore, the input data for this project has three features, road label, days in a week, and hours in a day. For this project, 35 roads around Central Park, New York City are included. Data are collected in a four-week span, from which one week is used as the validation set, and other three weeks are used as training sets. There are around 5,880 examples per week for 35 streets in total.

4 Method

The method used in this project is a neural network with classification to categorize data points into corresponding level of congestion. As its name suggests, a neural network consists of multiple layers of neurons, which are highly interconnected nodes that can process information in a highly efficient manner. Each neuron in the neural network is simply a mathematical function, and each neuron computes a weighted sum of its inputs. A neural network is trained by adjusting the neuron input weights to achieve better performance.

A neural network with a single hidden layer is used to train the model. The input layer has 3 neurons, road label, day and time, the output layer has 5 neurons, corresponding to five levels of congestion. For a single input $x^{(i)}$, the forward propagation equations are:

$$a^{(i)} = \sigma(W^{[1]}x^{(i)} + b^{[1]}) \in \mathbb{R}^3$$
$$z^{(i)} = W^{[2]}a^{(i)} + b^{[2]} \in \mathbb{R}^4$$
$$\hat{y}^{(i)} = softmax(z^{(i)}) \in \mathbb{R}^5$$

Where $W^{[1]}$ and $W^{[2]}$ are weight matrices, $b^{[1]}$ and $b^{[2]}$ are biases, $\hat{y}$ is a vector of predicted result , σ and $softmax$ is the Sigmoid and Softmax function respectively,

$$\sigma(z^{(i)}) = \frac{1}{1+e^{z^{(i)}}}$$
$$softmax(z^{(i)}) = \frac{e^{z^{(i)}}}{\sum_{j=1}^{N} e^{z^{(j)}}}$$

The predicted congestion level can be retrieved by finding the corresponding index of maximum of $\hat{y}$ The backward propagation equations are:

$$\frac{\partial J}{\partial W^{[2]}} = \frac{1}{B} a^T (\hat{y} - y)$$
$$\frac{\partial J}{\partial b^{[2]}} = \frac{1}{B} \sum_{i=1}^{B} \left(\hat{y}^{(i)} - y^{(i)}\right)$$
$$\frac{\partial J}{\partial W^{[1]}} = \frac{1}{B} x^T \left(a(1-a)(\hat{y} - y) W^{[2]^T}\right)$$
$$\frac{\partial J}{\partial b^{[1]}} = \frac{1}{B} \sum_{i=1}^{B} \left(a(1-a)(\hat{y}^{(i)} - y^{(i)}) W^{[2]^T}\right)$$

Where B is the mini-batch size, y is the true label, and λ is the regularization coefficient. The cross entropy loss can be calculated through:

$$J = -\frac{1}{B} \sum_{i=1}^{B} y_k^{(i)} \log \hat{y}_k^{(i)}$$

5 Experiments, Results, and Discussion

5.1 Hyperparameters

The size of the hidden layer is chosen to be 4, which is the mean of number of neurons in input and output layers. The learning rate is chosen to be 3, which yields the highest accuracy. The total amount of data point is 17640, the minibatch size is chosen to be 1000, therefore it takes 18 iterations to finish the entire dataset.

5.2 Accuracy and Loss

To evaluate the performance of the model, accuracy and loss during the training are plotted against episodes in Figure 1. As shown in Figure 1, it is clear that training accuracy and loss curves began to flatten around 20,000 episodes. The final training accuracy reports to be 74.6%, and the accuracy on validation set is 76.8%.

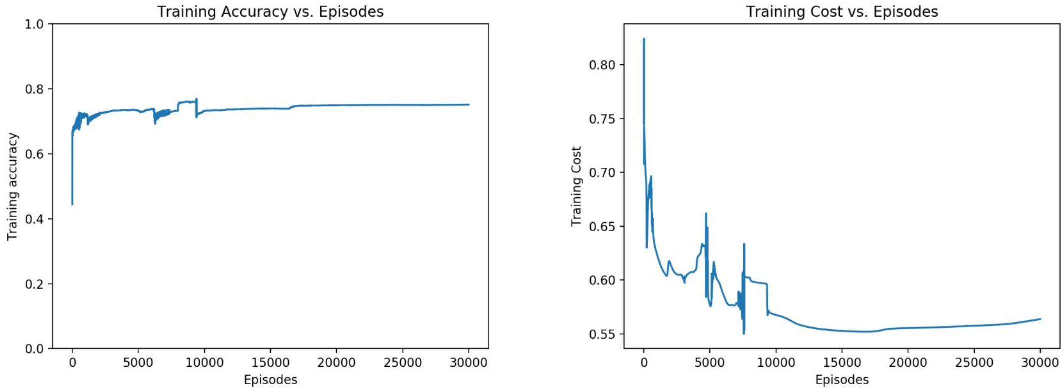

Figure 1: Training Accuracy (left) and Loss (right)

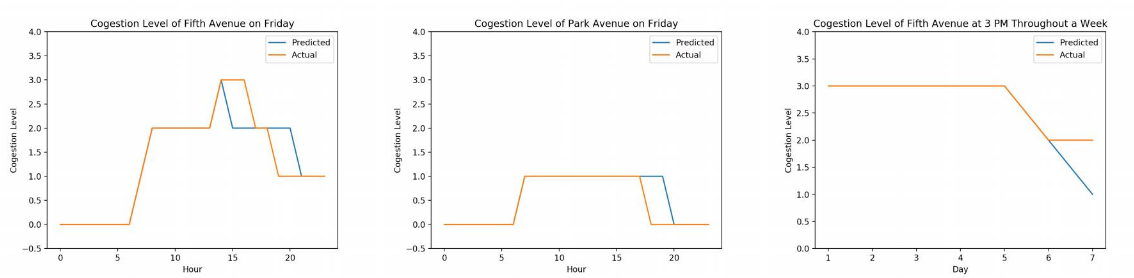

Figure 2: Fifth Avenue (left) and Park Avenue (right) Traffic Congestion level

5.3 Graphical Comparison

To better visualize the results, the predicted congestion level of Fifth Avenue and Park Avenue on Friday, which is the most crowded day in a week, is plotted in Figure 2. It is very clear that the predicted model captures the trend of traffic generally. For Fifth Avenue, rush hours starts around 6 AM, worsens around noon and comes back to normal around 8 PM. On the contrary, the traffic on Park Avenue is pretty consistent throughout the day. To view the traffic change at specific time throughout the day, the congestion level of Fifth Avenue at 5 PM everyday throughout a week is plotted.

To have a more generalized view of all street, a color-coded map is generated using coordinates obtained from the original dataset, where color green, orange, red, and dark red corresponds to level zero, one, two, and three of traffic congestion. Screenshots from google map of corresponding area is also included for direct comparison.

The predicted traffic condition accurately captured the road condition of most streets in Manhattan. Even though there are still some discrepancies, which are likely caused by the difference of resolution between the predicted map and screenshot from Google map. Traffic condition in Google Map is evaluated at each block, whereas the prediction evaluate congestion level by each street due to the limitation of raw data.

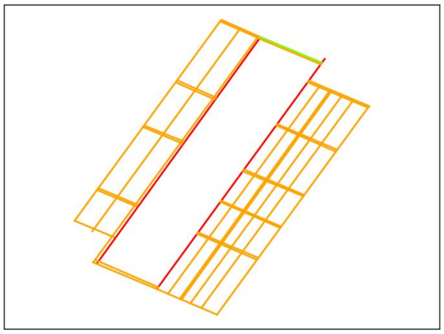 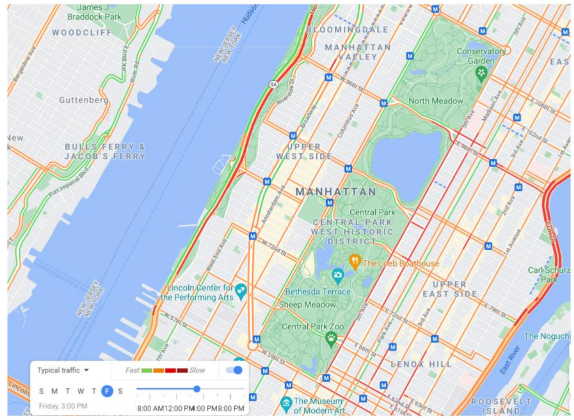

Figure 3: Friday 3 PM Traffic in Manhattan Prediction (left) and Screenshot from Google Map[2]

6 Conclusion and Future Work

In this project, we feed the traffic flows in a range between 0 and 4 to our neural network. We then trained the model with historical traffic data to predict traffic in an specific area in New York. The algorithm captures the trends of traffic well. However, we are not quite satisfied with the accuracy we obtained. What we consider would be helpful to improve the accuracy is that we can obtain a larger number of more accurate historical data. We might look for some GPU centers for training with that large number of data if we have the chance. Besides that, we would like to explore other algorithms we discovered in the papers we cited above. There are obvious trends in traffic situations and some algorithms, for instance LSTM, capture the trends in traffic very well. Those algorithms can help us obtain a model and hence improve the performance of our model to simulate more authentic situations in real-life.

7 Contributions

Yulin Huang: Searched for potential databases online and contacted service providers. Labelled data, adjusted the map, and helped with debugging code for the neural network.
Genggeng Zhou: Labelled data, coded up the neural network, tuning the parameters to train a better model.

References

[1] Murat Sari B. Gültekin Çetiner and Oğuz Borat. A neural network based traffic-flow prediction model. Mathematical and Computational Applications, 15(15):269–278, 2010.

[2] Google. Google maps], 2021.

[3] Yajie Zou Weibin Zhang Jinjun Tang, Fang Liu and Yinhai Wang. An improved fuzzy neural network for traffic speed prediction considering periodic characteristic. IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, 18(19):2340–2350, 2017.

[4] Outscraper. Google maps traffic extractor, 2018-2021.

[5] Yilun Lin Li Li and Fei-Yue Wang Xingyuan Dai, Rui Fu. Deeptrend: A deep hierarchical neural network for traffic flow prediction. arXiv:1707.03213v1, 2017.