# Demand Prediction Using NYC Yellow Taxi Data
# A study about CNN-LSTM and Transformer

**Team 1**
Minh Duc Nguyen, Seongouk Kim,
Thu Phuong Nguyen, Sumin An, Sangjune Park

## Abstract

Traffic forecasting problem is an active research field and one of the challenging problems is taxi demand prediction. To predict taxi demand, a lot of approaches suggested. In this project, we trained CNN-LSTM and transformer-based architecture with NYC yellow taxi data set. Results showed that the transformer-based model outperformed CNN-LSTM but CNN-LSTM had advantages in training time.

## 1 Introduction

The problem of traffic forecasting is an active research field with numerous new ideas tackling different aspects of the problem, and it has a lot of great applications such as in transportation, logistics, and city planning (Zheng et al., 2014).

Taxi is one of the essential means of transportation in the current urban era. With the rising of companies such as Uber and Grab, people now start to take taxis to travel more often than in the past. Efficient and accurate taxi demand prediction contributes to understanding the city's traffic flow and also allows taxi companies to optimize the cost since they can strategically place drivers according to the demand heat map. Ultimately, a successful model can provide a better user experience (Xu and Li, 2019).

Taxi demand prediction is a complex problem due to its stochastic behavior, and its dependence on spatial information of the network since nowhere has the same desire to ride a taxi (Liu et al., 2020). Moreover, it is evident that this problem also relies on historical information about people's demands for taxis, hence, in order to suggest an approach, one must solve its dependence on temporal information.

There have been a lot of approaches suggested by various researchers using different methods. Nowadays, with the discoveries of Deep Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), and Transformer (Vaswani et al., 2017), deep learning has now come to play in virtually almost every field, and taxi demand prediction is not an exception. Most of the State-Of-The-Art models jointly tackle the spatial-temporal issue by using the combination of CNN and LSTM (Shu et al., 2020). There are also impressive works that incorporate the attention mechanism to enhance the accuracy of the architecture (Yao et al., 2018). In more recent research, Transformer-based models start to achieve state-of-the-art performance thanks to Transformer's incredible ability to process sequence data (Lin et al., 2020).

Although there are a lot of deep learning approaches to tackle this demand prediction problem, there is a lack of models to produce an accurate heat map of the taxi demand of a city. In this work, we try to address this problem by proposing two different models and comparing their performance against each other: a CNN-LSTM-based model and a Transformer-based model. The data set used in this work is the New York City taxi data collected from the first two months of 2019.

The contributions of our work are as follows

- An exploratory data analysis of NYC Taxi data set.
- A simple CNN-LSTM-based model that tackles the temporal information of each region over an interval of time.
- A simple Transformer-based architecture that emerges both spatial and temporal information of the overall network using information fusion.
- A comprehensive comparison between our built CNN-LSTM and Transformer models.

## 2 Related works

In this section, we will discuss various related works that also provide great approaches to this

problem.

Early traffic forecasting papers suggest the use of regression models such as ARIMA, Historical Average, and Ridge. Due to its temporal nature, later state-of-the-art models start to feature RNN and its variants such as LSTM, and Gated Recurrent Unit (GRU). To capture the spatial information of the network, state-of-the-art architectures rely on the use of CNN, and Graph Neural Networks (GNN). Some examples are ConvLSTM (Shi et al., 2015) which combines CNN and LSTM, and ST-ResNet (Zhang et al., 2017) which harnesses the power of Residual Network and RNN to capture spatial-temporal characteristics of the data. In recent years, with the release of the paper "Attention Is All You Need" (Vaswani et.al., 2017), a new way of constructing solutions is open to traffic forecasting. Notable Transformer-based architectures are ST-TIS (Li et al., 2022) which suggests an optimized attention mechanism to boost the training time, and BDSTN (Cao et al., 2022) which applies a BERT-based model to predict the taxi demand.

## 3 Problem formulation

**Definition 1.** *The city is divided into $n$ non-overlapping regions, called taxi zones. Let's denote $Z = \{z_1, z_2, \ldots, z_n\}$ be the set of all taxi zones in the city, where $z_i$ is the $i^{th}$ zone.* (Li et al., 2022)

Note that this partition is arbitrary, i.e., not necessarily rectangular grids. In this work, we use the partition of the NYC data set that has already been divided into $n = 263$ zones according to the real locations of the city.

**Definition 2.** Let $\mathcal{X}^t = \left(x_1^t, x_2^t, \ldots, x_n^t\right)$ be the vector of taxi demand of NYC at time $t$, where $x_i^t$ is the demand of the $r_i^{\text{th}}$ zone at time $t$. Also, let's denote $x_i^{t':t} = \left(x_i^{t'}, x_i^{t'+1}, \ldots, x_i^t\right)$ indicating the demand of the $r_i^{\text{th}}$ from time $t'$ to $t$. Similarly, $\mathcal{X}^{t':t} = \left(x_1^{t':t}, x_2^{t':t}, \ldots, x_n^{t':t}\right)$ is the demand of the city from time $t'$ to time $t$.

This definition will help formulate the problem formally. The following is the formal definition of the taxi demand prediction

**Definition 3.** *Given the demand data from a fixed length, called $l$, and an interval $[t-l, t]$ for $t \in \mathbb{R}$, i.e., $\mathcal{X}^{t-l:t}$. Define a function $\mathcal{F} : \mathbb{R}^{n \times l} \to \mathbb{R}^{1 \times n}$ that predicts the taxi demand of every zone, i.e., $\mathcal{F}\left(\mathcal{X}^{t-l:t}\right)$ should be the taxi demand at time $t+1$.*

Our objective is to learn the function $\mathcal{F}$.

## 4 Dataset

We conduct the experiment on NYC Yellow Taxi Trip Records dataset in the first two months of 2019, which contains around 14 million data points. For this real-world Yellow dataset, all the pickups were not scheduled in advance but were requested by waving at a passing taxi, which can cut down on the frequency of virtual taxi calls through taxi technology applications. Each taxi record contains the pick-up and drop-off date, time, location ID, trip distances, and passenger counts. We split the dataset into three parts: a training set, a validation set, and a testing set. The first 64% of the dataset are chosen for training our model, the next 16% of the sample are validation set for turning weight parameters and the final 20% is for testing the performance of our model. The temporal feature and spatial features employed in our experiment are comparable to those found in (Shu et al., 2020). We captured all the necessary characteristics of the data such as the temporal features (e.g., the commuters' demand on workdays and weekends) and the spatial features (e.g., the average demands related to geographical features).

### 4.1 Temporal features

The demand for taxis among New York citizens is related to the time frame and the day of week. For example, more people commute throughout the day (from 6 a.m to 6 p.m) and the evening (from 6 p.m to 12 p.m) in comparison to at dawn (between 0 a.m and 6 a.m). The majority of midnight commutes take place on weekends.

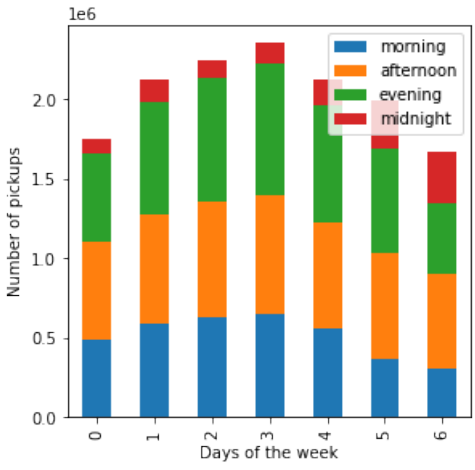


Figure 1: Taxi demand per day of week

In addition to the time frame, the current short

time intervals also have an effect on commuter demand. For instance, the peak hours for commuting to and from work on weekdays are 8 a.m and 6 p.m, respectively. The number of taxi pickups pre and post-peak hours are affected by this traffic congestion time by around an hour.

As can be seen from Figure 2, the taxi demand on weekdays is higher than on weekends. Therefore, the differences between weekends and weekdays and the periodicity of the number of taxi pickups are used as features of the LSTM model to predict taxi demands.

### 4.2 Spatial features

Geographical factors have a great impact on people's desire for taxi services, which is a vital aspect in anticipating demand and scheduling taxis to meet the demand for different regions. As the problem formulation mentioned above, NYC's actual locations were partitioned into 263 zones. The demands in some regions are much higher in both neighboring areas and far distant areas. For instance, there are high demands in the peak hours among residential neighborhoods and areas with a high concentration of office structures. Due to the busy schedule of flights at dawn and at night, there are also frequent demands for taxi services between residential areas and airports, especially at night.

Therefore, the attractiveness of locations determines the taxi demand. Although the direction of demand changes on a regular basis, it shows a clear trend that the taxi demands occurred much more frequently in the Manhattan area and in 2 airports named John F. Kennedy International Airport (JFK) which is in the lower right part of the map and LaGuardia Airport (LGA) in the North of Queens area. Refer to Figure 3 for a visualization.

## 5 CNN-LSTM-based model details

In this section, we will present a simple but effective combination of CNN and LSTM to solve the problem. To begin with, we elaborate on the use of the 1-D convolutional layer, followed by the LSTM layer. Finally, a fully connected layer is deployed to obtain the heat map of demand. Refer to Figure 4 for a visualization.

### 5.1 1-D Convolutional layer

To extract temporal patterns from individual taxi zone, we use a 1-D convolutional operation on each of the input components $x_i^{t-l:t}$, the input then becomes

$$\hat{x}_i^{t-l:t} = relu\left(Conv1D\left(x_i^{t-l:t}\right)\right),$$

where $\hat{x}_i^{t-l:t} \in \mathbb{R}^{1\times l}$ is the embedding of the temporal pattern of the zone $r_i$ from time $t-l$ to $t$, and $Conv1D\left(\cdot\right)$ is the 1D convolutional operation.

### 5.2 LSTM layer

To capture the temporal information of the entire city, one layer of LSTM is used. The cells used in this layer is a traditional ones, namely, (Sak et al., 2014)

$$\begin{aligned}
i_t &= \sigma\left(W_{ii}\mathcal{X}^t + b_{ii} + W_{hi}h_{t-1} + b_{h_i}\right)\\
f_t &= \sigma\left(W_{if}\mathcal{X}^t + b_{if} + W_{fi}h_{t-1} + b_{h_f}\right)\\
g_t &= \tanh\left(W_{ig}\mathcal{X}^t + b_{ig} + W_{gi}h_{t-1} + b_{h_g}\right)\\
o_t &= \sigma\left(W_{io}\mathcal{X}^t + b_{io} + W_{oi}h_{t-1} + b_{h_o}\right)\\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t\\
h_t &= o_t \odot \tanh\left(c_t\right),
\end{aligned}$$

The output of the last layer $h_t \in \mathbb{R}^{n\times 1}$ is a vector of features that is to approximate $\mathcal{X}^{t+1}$.

### 5.3 Fully connected layer

Lastly, to obtain the demand heat map from the extracted feature, one linear layer is appended to the last layer of the model. Finally, the $relu$ activation is applied to the output of the model,

$$\hat{\mathcal{X}}^{t+1} = relu\left(h_t W^{\mathcal{X}} + b^{\mathcal{X}}\right),$$

where $\hat{\mathcal{X}}^{t+1} \in \mathbb{R}^{n\times 1}$ is the output of the model, $W^{\mathcal{X}}$ and $b^{\mathcal{X}}$ are learnable parameters.

## 6 Transformer-based model details

In this section, we will present a simple Transformer-based architecture that is capable of tackling both spatial and temporal dependency of the taxi demand prediction problem. To begin with, we suggest the use of Infomation Fusion Module (Li et al., 2022), which is a linear combination of spatial embedding, temporal embedding, and demand embedding. After that, we pass the embedded feature into a Transformer encoder to extract the dependencies between two different regions, followed by two linear layers appended to the end in order to extract the prediction from the suggested information. Refer to Figure 5 for a visualization.

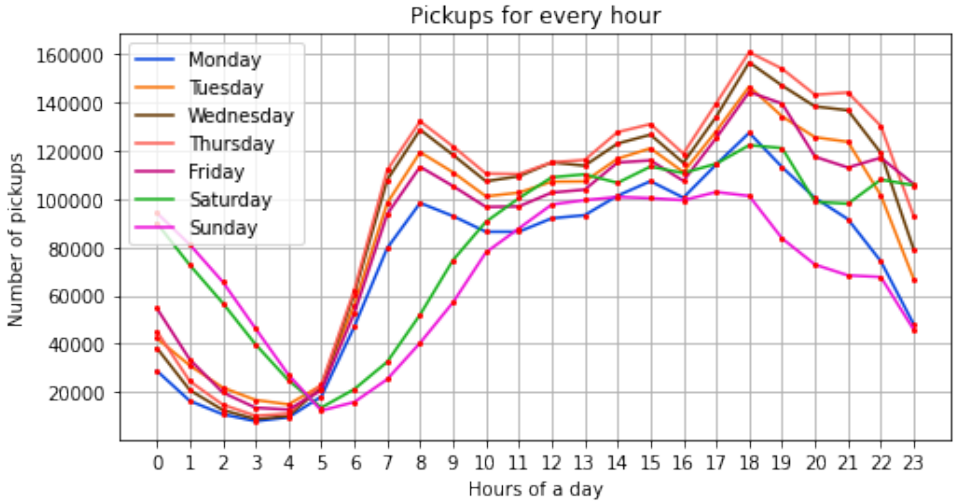


Figure 2: Number of pickups for every hour

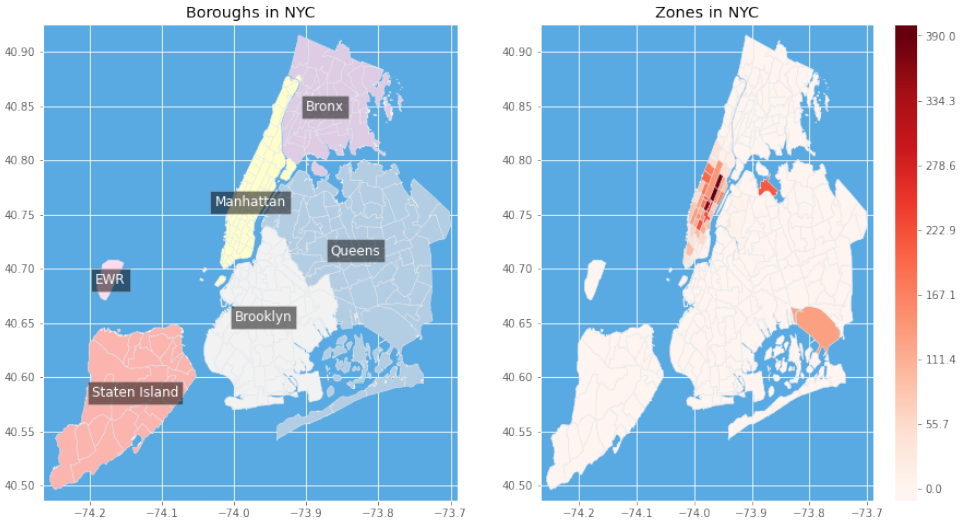


Figure 3: Taxi demand occurred in each zone of NYC

### 6.1 Infomation Fusion

To handle both spatial and temporal information jointly, it is necessary to concatenate both features for each zone, along with demand patterns.

Firstly, we conduct a one-hot-encoding of the $n$ taxi zones, namely $S_i \in \mathbb{R}^{1\times n}$, representing the $z_i^{\text{th}}$ taxi zone. Then, a fully connected layer is applied to extract the spatial embedding

$$\hat{S}_i = S_i W^S + b^S,$$

where $\hat{S}_i \in \mathbb{R}^{1\times k}$ is the spatial embedding of the $z_i^{\text{th}}$ taxi zone, $k$ is a hyperparameter, $W^S$ and $b^S$ are learnable parameters.

Secondly, we conduct a one-hot-encoding of the timestamps in one day. We first split one day into $m$ intervals, and represent each interval as a one-hot vector $T_i \in \mathbb{R}^{1\times m}$. Then, a fully connected layer is applied to extract the temporal embedding

$$\hat{T}_i = T_i W^T + b^T,$$

where $\hat{T}_i \in \mathbb{R}^{1\times k}$ is the temporal embedding of the $i^{\text{th}}$ timestamp, $W^T$ and $b^T$ are learnable parameters.

Thirdly, similar to the CNN-LSTM approach, to retrieve historical patterns from each taxi zone, we use a convolutional operation on each of the input components $x^{t-l:t}$, in addition to that, one more linear layer is added,

$$D_i = relu\left(Conv1D\left(x_i^{t-l:t}\right)W^x + b^x\right),$$

where $D_i \in \mathbb{R}^{1\times k}$ is the embedding of the temporal pattern of the zone $r_i$ from time $t-l$ to $t$, and $Conv1D(\cdot)$ is the 1D convolutional operation.

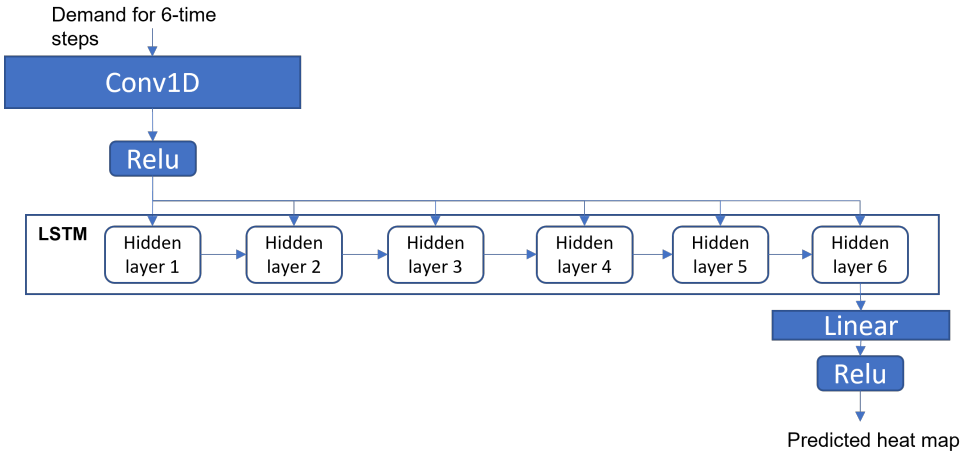


Figure 4: CNN-LSTM overview

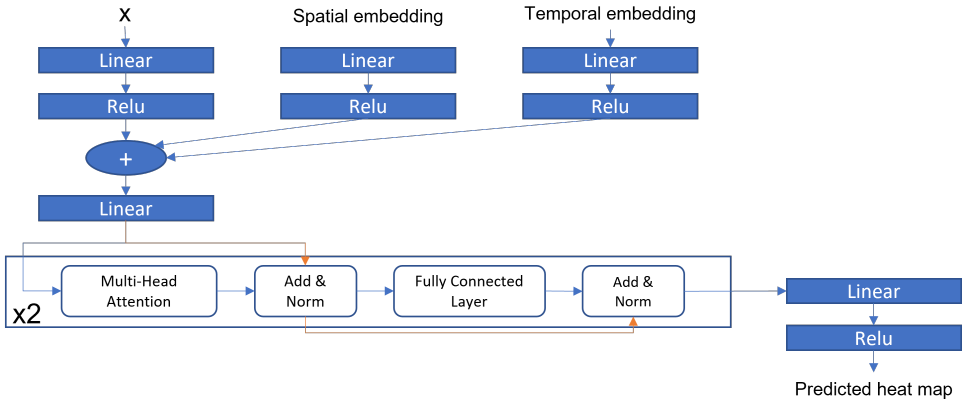


Figure 5: Transformer-based model overview

Lastly, all of the above features will now be fused into a single vector by using a simple linear combination on them

$$P_i = \left(\hat{S}_i + \hat{T}_{f(i)} + D_i\right) W^P + b^P,$$

where $P_i^t$ is the Spatial-Temporal-Demand (STD) embedding of the $z_i$ taxi zone at the time $t$, and $f(i)$ is the correct matching timestamp function.

### 6.2 Transformer encoder

To further capture the dependencies between different taxi zones, we apply a Transformer encoder, which calculates the attention score between any pairs of zones. The formula for computing the attention score is the same as the original Transformer encoder (Vaswani et al., 2017).

$$Attn(r_u, r_v) = \text{softmax}\left(\frac{Q\left(P_u\right) \times K\left(P_v\right)^T}{\sqrt{d_k}}\right)$$
$$Q(P_i) = P_i W^Q$$
$$K(P_i) = P_i W^K,$$

Where $W^Q$ and $W^K$ are learnable parameters. We pass all $n$ regions obtained from the previous layer, namely, $(P_1, P_2, \ldots, P_n)$ to the Transformer encoder resulting in $\hat{P}_1, \hat{P}_2, \ldots, \hat{P}_n$, where $\hat{P}_i$ is the embedding of the taxi zone $z_i$.

### 6.3 Fully connected layers

Last of, we employ two linear layers at the end of the architecture to convert the embedding information to the desired output, i.e.,

$$\hat{\mathcal{X}}_i^{t+1} = relu\left(relu\left(P_i W_1^{\mathcal{X}} + b_1^{\mathcal{X}}\right) W_2^{\mathcal{X}} + b_2^{\mathcal{X}}\right),$$

where $W_1^{\mathcal{X}}$, $b_1^{\mathcal{X}}$, $W_2^{\mathcal{X}}$, and $b_2^{\mathcal{X}}$ are learnable weights.

## 7 Experiment

In this section, we discuss in detail the experiment process including the hyperparameter choices, the result, and visualizations. We train both of our models on google colab using the same data set of NYC Taxi data from 2019/01/01 to 2019/02/28.

### 7.1 Training

As discussed above, we split the data into two parts: $80\%$ for training data and $20\%$ for testing data. In the training data, we select $80\%$ to train our model, and the remaining $20\%$ for validation. Before feeding the data into the model, we apply Min-Max normalization, which scales the data into the range $[0, 1]$, hence, smoothening the landscape of the objective function. Moreover, in this work, we let $l = 6$, i.e., our model takes 6 past data points as input to predict the next one. Also, we decide to divide the day into 30 minutes intervals, which sums up to 48 data points per day. Thus, our model takes the past 3 hours as input. For both of our models, we use ADAM optimizer (Kingma and Ba, 2017). The loss function that we use for both of them is Root Mean Square Error (RMSE), whose formula is

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} \left(\hat{\mathcal{X}}_i^{t+1} - \mathcal{X}_i^{t+1}\right)^2}{n}}$$

We implement our models using PyTorch, data and code can be found in https://github.com/kurone02/Taxi-Demand-Prediction

As for the CNN-LSTM model, we train the architecture for 150 epochs with a batch size of 10. The learning rate is initially set to $10^{-3}$, and is scheduled to decrease using the following formula

$$\theta_{\text{new}} = 0.98^{\theta_{\text{old}}},$$

where $\theta \in \mathbb{R}$ is the learning rate.

As for the Transformer-based architecture, we train it for 500 epochs with a batch size of 16. Early stopping is implemented to save the best model and to stop the training process after 50 consecutive epochs without improvement of the validation loss. Also, a custom learning rate scheduler, which decreases the learning rate by a factor of $0.9$ every 10 epochs, is also added.

### 7.2 Result

The followings are the best results obtained from training the aforementioned models.

As for the CNN-LSTM-based model, it archives a loss value of $RMSE = 8.650454$ on the test set.

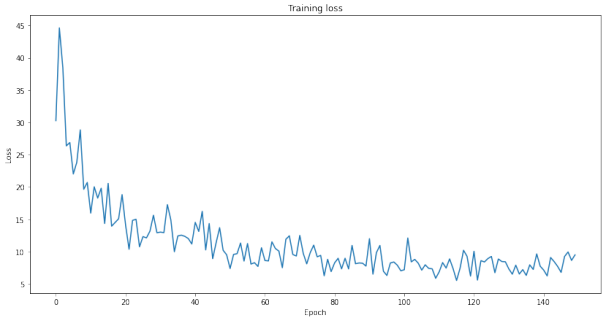


Figure 6: Training loss of CNN-LSTM

As for the Transformer-based model, it archives a loss value of $RMSE = 7.092931$ on the test set. The early stopping mechanism halts the training process at around 200 epochs when the function starts to converge. One could check Figure 10 and Figure 11 to verify the accuracy of the model.

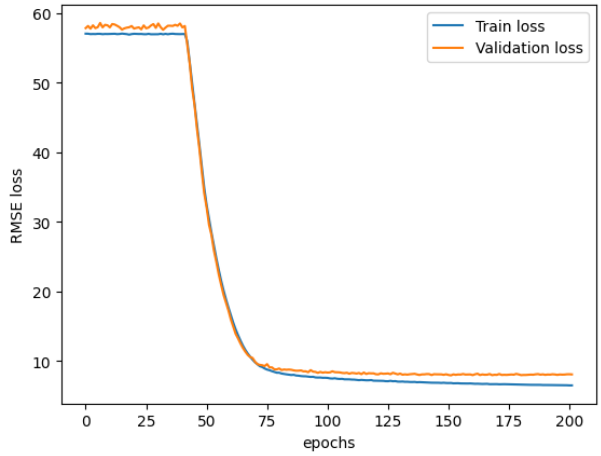


Figure 7: Training loss of Transformer

## 8 Discussion

In this section, we discuss the difference between the two models and compare their performance on the same data set.

One could notice that the loss value of CNN-LSTM converges much more quickly than the other model. On the other hand, the behavior of the loss function in the Transformer seems to suffer from a high learning rate in the first 40 epochs but quickly reaches the best value in just 25 epochs later after

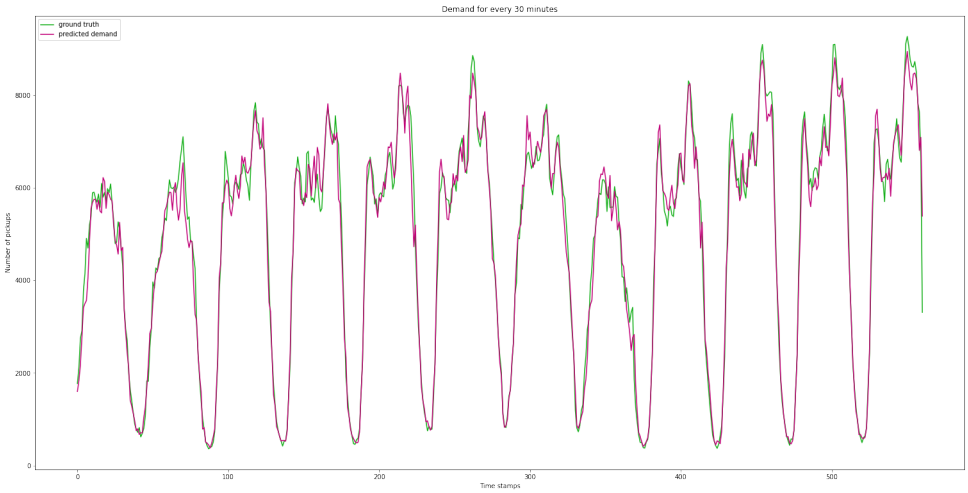


Figure 8: CNN-LSTM demand prediction for every 30 minutes

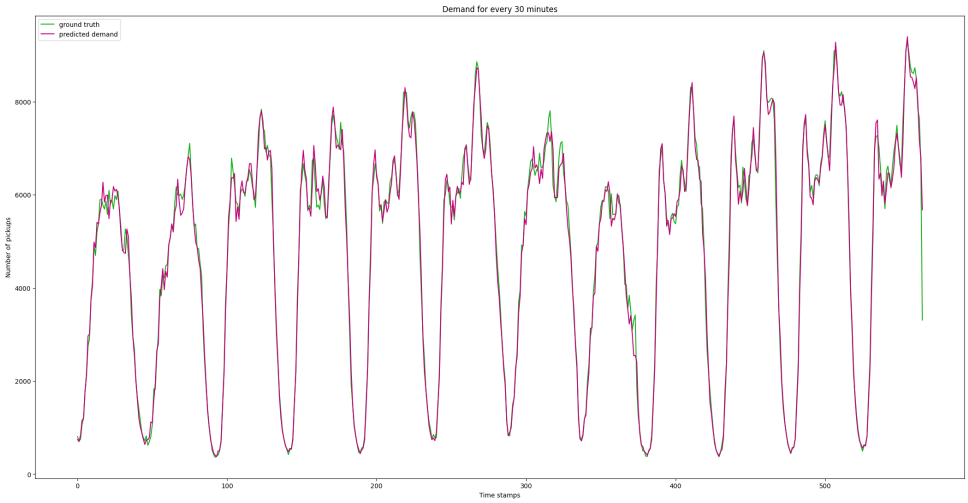


Figure 9: Transformer demand prediction for every 30 minutes

the learning rate has been reduced to a reasonable amount.

The test results in Section 7.2 suggests that the Transformer-based model outperforms the CNN-LSTM-based model. One could also refer the Figure 8. and the Figure 9 to see the difference. With just a glance, normal people barely see the difference between the two. However, with further investigation, one could note that the prediction from Transformer is much closer to the real-world patterns than the one from CNN-LSTM.

However, the training time from CNN-LSTM is faster than the Transformer due to the $O(n^2)$ complexity of attention computation in the Transformer, the shallowness of CNN-LSTM, and the data-hungry nature of the Transformer (Wang et al., 2022).

## 9 Conclusion

In this work, we proposed two approaches for tackling the taxi demand prediction problem. We built a CNN-LSTM-based model which focused on the temporal information, and a transformer-based model which captured both temporal and spatial information. Comparing them showed that each model had its own pros and cons. With these models, we expect to solve the drawing demand heat map problem.

## References

Dun Cao, Kai Zeng, Jin Wang, Pradip Kumar Sharma, Xiaomin Ma, Yonghe Liu, and Siyuan Zhou. 2022. Bert-based deep spatial-temporal network for taxi demand prediction. *IEEE Transactions on Intelligent Transportation Systems*.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Guanyao Li, Shuhan Zhong, S.-H. Gary Chan, Ruiyuan Li, Chih-Chieh Hung, and Wen-Chih Peng. 2022. A lightweight and accurate spatial-temporal transformer for traffic forecasting.

Haoxing Lin, Rufan Bai, Weijia Jia, Xinyu Yang, and Yongjian You. 2020. Preserving dynamic attention for long-term spatial-temporal prediction. *KDD '20: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Zhizhen Liu, Hong Chen, Yan Li, and Qi Zhang. 2020. Taxi demand prediction based on a combination forecasting model in hotspots. *Journal of Advanced Transportation*.

Haşim Sak, Andrew Senior, and Françoise Beaufays. 2014. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition.

Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai kin Wong, and Wang chun Woo. 2015. Convolutional lstm network: A machine learning approach for precipitation nowcasting.

Pengfeng Shu, Ying Sun, Yifan Zhao, and Gangyan Xu. 2020. Spatial-temporal taxi demand prediction using lstm-cnn. *2020 IEEE 16th International Conference on Automation Science and Engineering*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Wen Wang, Jing Zhang, Yang Cao, Yongliang Shen, and Dacheng Tao. 2022. Towards data-efficient detection transformers.

Ying Xu and Dongsheng Li. 2019. Incorporating graph attention and recurrent architectures for city-wide taxi demand prediction. *International Journal of Geo-Information*.

Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2018. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction.

Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction.

Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: Concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology*.

## 10 Appendix

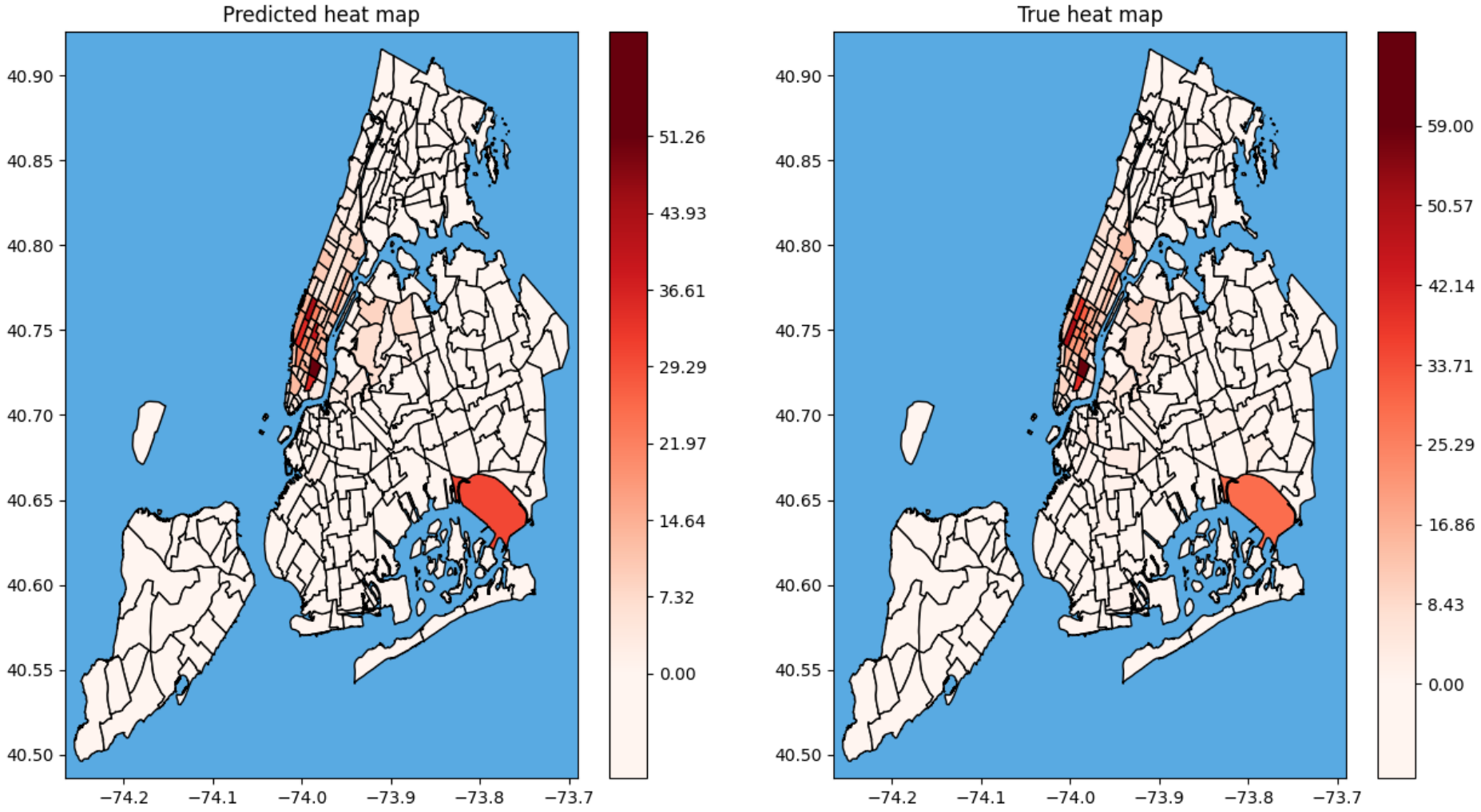


Figure 10: Transformer heat map prediction and ground truth at 3 AM, 2019/02/17

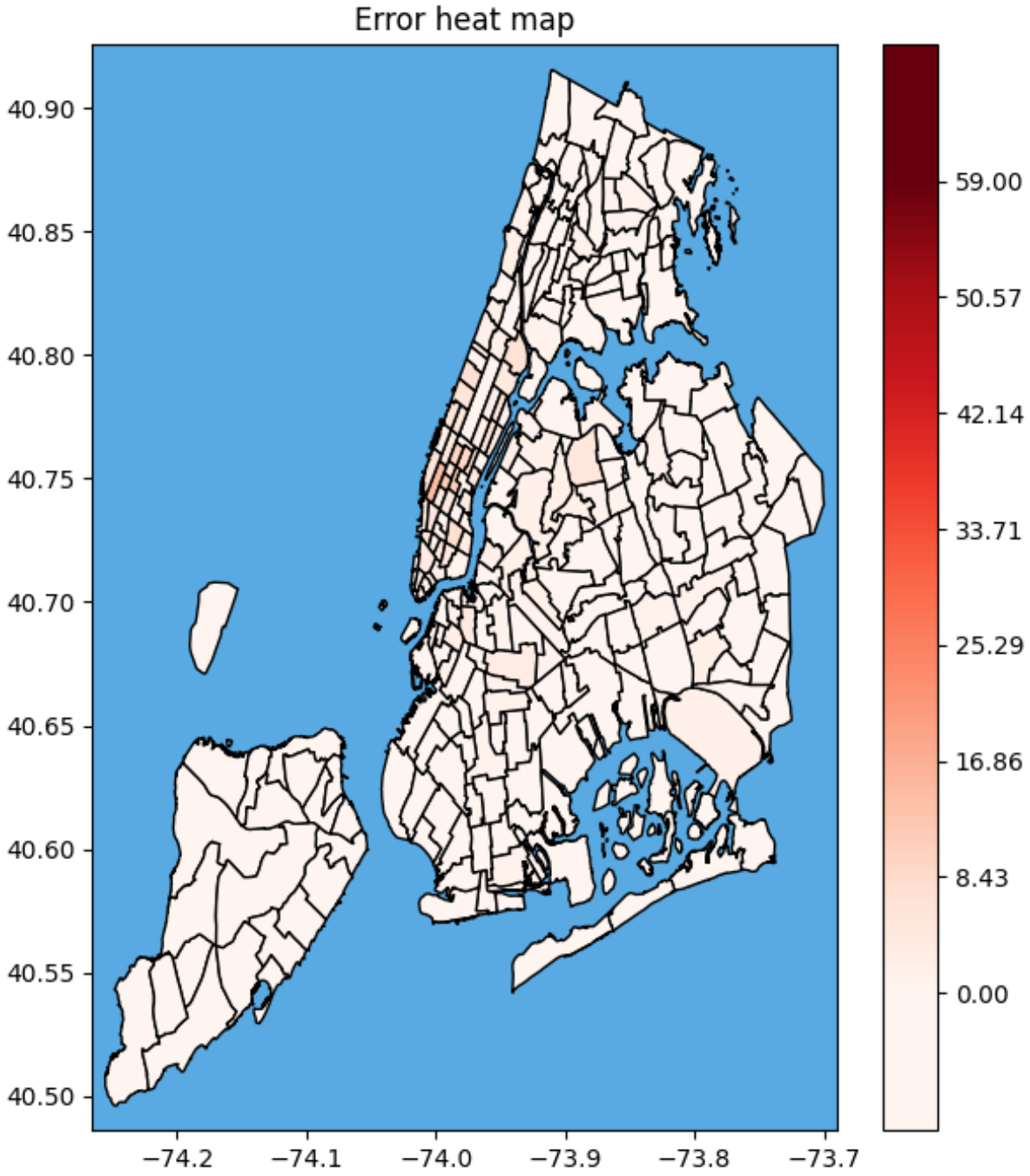


Figure 11: Transformer heat map prediction error at 3 AM, 2019/02/17