Transport and Telecommunication, 2023, volume 24, no. 1, 22–32
Transport and Telecommunication Institute, Lomonosova 1, Riga, LV-1019, Latvia
DOI 10.2478/ttj-2023-0003

EFFECT OF SPATIO-TEMPORAL GRANULARITY ON DEMAND PREDICTION FOR DEEP LEARNING MODELS

Ken Koshy Varghese[1], Sajjad Mahdaviabbasabad[2], Guido Gentile[1], Mohamed Eldafrawi[1]

[1]Università di Roma La Sapienza,
Via Eudossiana 18 - 00184 Roma, Italy
{kenkoshy.varghese, mohamed.eldafrawi guido.gentile}@uniroma1.it
[2]Università di Roma La Sapienza,
Via Eudossiana 18 - 00184 Roma, Italy
mahdaviabbasabad.1852942@studenti.uniroma1.it

Advances in machine learning technology and the availability of big data from GPS systems have led to the development of effective methods for modelling transportation demand and forecasting the future. Most previous research concentrated on demand prediction using a variety of machine learning and deep learning models that took into account spatial and temporal relationships. This paper investigates the impact of spaces and time granularity for a Spatio-temporal demand modelling framework. Using taxi demand data from New York City, our study compares the prediction performance of deep learning models such as Long Short-Term Memory (LSTM), Convolution Neural Networks (CNN) and Temporal-Guided Networks (TGNet), modelled with a grid-based tessellation strategy. The findings of this study could assist researchers in better understanding how the granularity of space and time helps deep learning models perform better for demand forecasting problems.

Keywords: Taxi Demand, Demand Forecasting, Spatiotemporal Granularity, Deep Learning, Grid Clustering

1. Introduction

Cities are places where economic activities are accumulated and concentrated. The ability of a city's transportation systems to move labour, consumers, and freight between many origins and destinations is the key to its productivity. Many public transport systems have been overused or underused as public transport demand varies with time and space. During high usage hours, congestion can cause user discomfort as the system copes with temporary spikes in demand. Improper planning and design of transport and its services can lead to a massive failure in the city's productivity. Peak capacity design leaves the system underutilised during off-peak hours, whereas average capacity planning causes congestion during peak hours. Therefore, accurately predicting the passenger demand for transport infrastructure is necessary for planning, designing, evaluating, and regulating transport systems. From a business perspective, estimating expected consumer demand is essential in all scheduling and planning activities to improve business profitability. For instance, a ride-sourcing company can use the predicted taxi demand to deploy taxis in advance to high-demand locations, cutting down on the waiting time for passengers and the idle time for the taxis themselves. Increased use of taxis might also enhance driver pay and reduce energy consumption. As a result, public transportation operators and logistics companies are keen to create and interpret the results of precise and reliable demand forecasting models.

With the help of modern technologies, enormous amounts of information, including GPS locations, time, number of passengers, weather, traffic counts, and other information, can be acquired in real-time and stored in different cloud-based databases. These data can be analysed to indicate different traffic patterns. As a result of these trends, it may be able to predict future traffic issues. This information paves the way for developing an intelligent transportation system capable of controlling and coordinating supply on a broad scale while benefiting businesses. In addition to the exponential growth of historical data, forecasting has improved dramatically due to recent advances in Artificial Intelligence (AI). For example, machine learning (ML), a subfield of AI, has algorithms capable of capturing non-linear correlations between input and output data. These algorithms are commonly known as deep learning (LeCun *et al.*, 2015) and include Artificial Neural Networks (ANNs) based partly on how biological neurons in the brain work.

Numerous research on the prediction of traffic data, such as traffic volume, taxi pick-ups, and traffic inflow/outflow volume, have been published in the literature. Time series prediction techniques have been used frequently to forecast traffic. In particular, autoregressive integrated moving averages (ARIMA) and their variations have been used extensively for traffic prediction (Williams & Hoel, 2003). Recent research has considered geographical relationships and external information like weather data based on the time series, proving that the prediction can be improved by considering additional features (Pan *et al.*, 2012; Wu *et al.*, 2016). However, a noticeable trend in these studies is that a fixed area, as well as a fixed time interval or considering a fixed period of historical data as features are considered. Most of the research neglect the effect of different space and time granularity on the accuracy of demand prediction models.

In this research, our primary goal is to investigate the effects of different Spatio-temporal granularities on the performance of deep learning models and reach a conclusion on the appropriate spatiotemporal feature to select to get the best prediction results from the model. The objectives of this study are to address the following questions:

- Is there an optimal spatiotemporal granularity that produces more accurate predictions?
- Does the forecast accuracy differ between the model trained with aggregated demands for the entire city for each time interval and the model trained with aggregated demands for each city zone after spatial clustering?
- Which deep learning model works best for which space-time granularity?
- Can we use a whole city model to forecast the demand for local city zones?

The rest of this paper is organised as follows: The earlier research and most recent developments in estimating taxi demand are covered in Section 2. Section 3 explains the approach used. The analysis and experimental results are presented in Section 4. This paper is concluded in the final section.

2. Literature Review

The traditional approach used a statistical methods model for data with temporal correlation for travel demand predictions. The ARIMA model is the most extensively used method, assuming traffic forecasting is a stationary process with constant mean, variance, and auto-correlation (Williams & Hoel, 2003). Kalman filters (Okutani & Stephanedes, 1984) and Markov chains (Qi & Ishak, 2014) are also statistical approaches. These methods usually use a linear mathematical model to determine the inner properties of the traffic flow. However, these traditional time series prediction approaches perform well in stable and linear time series prediction but struggle in non-linear and unstable time series prediction. Furthermore, they ignore the spatial correlation and work well for short-term traffic predictions.

To resolve the limitations of the traditional approach, many researchers used deep neural networks. Felix A Gers *et al.* (Gers *et al.*, 2002) studied the suitability of using LSTM networks for time series forecasting. The researchers concluded that LSTM networks could tackle the vanishing gradient problem of basic neural networks and store essential long-term information. However, regarding short-term time series forecasting, LSTM networks may not consistently outperform more straightforward strategies like ARIMA. They recommend that LSTM networks be used only after more traditional methods have failed or for long-term time series forecasting.

Using big data and machine learning, Florin Schimbinschi *et al.* in (2015) explored traffic predictions in complicated metropolitan networks. They concluded that more data leads to better ML model predictions. They also concluded that spatial relationships between road segments are a better predictor than temporal patterns. Finally, they claim that ARIMA-based models have difficulty forecasting spatiotemporal data and cannot capture complicated dynamics. Laith Alzubaidi *et al.* (Alzubaidi *et al.*, 2021) is a survey paper that explains the most used DL method called Convolutional Neural Network (CNN). The researcher has reviewed 300 papers published during 2010-2021 and concluded that the main benefit of CNN compared to other DL models is that it automatically identifies relevant features without human supervision. In addition, the paper describes the development of CNN architecture along with its main features, current challenges, and solutions. Pedro Lara-Benítez *et al.* in (2021) investigated around seven DL methods for time series forecasting in terms of efficiency and accuracy. They trained 38000 models with a dataset having more than 50000-time series. They concluded that long short-term memory (LSTM) and convolutional neural network (CNN) are the best DL methods. LSTM obtains the most accurate forecast, and CNN performs similarly and more efficiently. It is also stated that without feature engineering requirements, CNNs can extract features from high-dimensional raw data having a grid structure, such as pixels in a picture. However, when using CNN, Huaxiu Yao *et al.* in (2019) believed that the complete city image harmed prediction accuracy. He used local CNN to introduce the semantic view and then integrated it with LSTM to improve prediction accuracy to tackle this problem and capture the spatial correlation. Even though the study captures both spatial and temporal correlation in both cases, they interacted between the two individually.

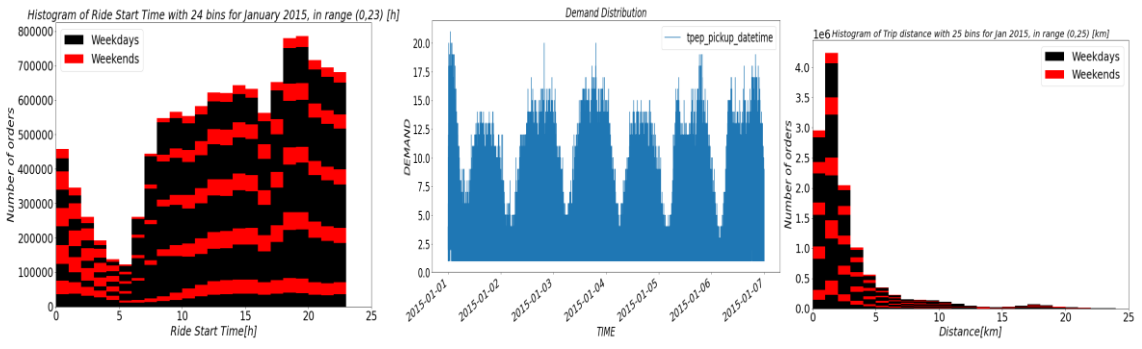
Even though these researches have considered the spatiotemporal correlations, minimal research has studied the effect of different spatial and time partitions. In a recent study, Liu Kai *et al.* (2022) explored the impact of 36 spatiotemporal granularities with departure and arrival demands, revealing that a hexagonal partition with an 800 m side length and a 30 min time interval produces the best overall prediction accuracy. However, in this study, only the previous 8-timestamps were considered for the next prediction. Therefore, our study compares various spatiotemporal granularity, considering a wide range of historical timestamps as features and studying its effect on the prediction performance of the deep learning models.

3. Methodology

3.1. Data Description

The data used in the project were collected by technology providers licensed by the Taxi and Uniform Passenger Improvement Program and provided to the New York City Taxi and Limousine Commission (TLC). This open-source data can be downloaded directly from the TLC website. The taxi trip logs include fields that capture the pick-up date/time, pick-up location, distance travelled, itemised fare, fare type, payment type, and the number of passengers reported by the driver. The data used for model training was the yellow taxis from January 2015 to June 2015; for validation and testing, the data was from January 2016 to June 2016. The six months of data contain almost 70 million records.

Any machine learning model must include data visualisation as a necessary step after data collection to comprehend demand's temporal and spatial distribution, as demonstrated in Figures 1 and 2. The hourly demand distribution shown in Figure 1(a) allows us to pinpoint the peak and low demand periods. The daily seasonality in the distribution is visible when we plot the demand distribution over a week (Figure 1(b)). Taxis are generally employed for short-distance travel, covering an average distance of 7 kilometres, according to the histogram of travel distance in Figure 1(c). Finally, Figure 2 shows the heat map from which we can understand the demand concentration in New York City.



(a) (b) (c)

Figure 1. Data analysis of NYC Yellow Taxi (a) Histogram of hourly demand distribution for January 2015, where black denotes weekdays, and red denotes weekends. (b) Daily demand for a week in January 2015 (c) Histogram of travel distance for January 2015

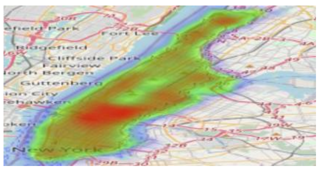


Figure 2. Heat map of demand for January 2015

3.2. Data Cleaning

One of the essential tasks in machine learning is data pre-processing. Because traffic data is acquired automatically using sensors and GPS, databases frequently contain missing numbers, out-of-range values, incorrect temporal information, and other errors. These outliers can mislead machine learning models, resulting in inaccurate forecasts. Because such a database contains millions of records, detecting and deleting outliers is tough. As a result, data pre-processing is the most crucial stage of a machine learning project.

The dataset of NYC yellow taxis underwent two cleaning procedures. Data were initially examined for missing data, inaccurate pick-up or drop-off dates, unwanted columns, outbound coordinates, and sea points. The data was cleaned using the Z-score in the second stage, which involved statistical analysis of explanatory variables such as travel distance, journey length, and speed. In this study, outliers are defined as data points with a Z-score greater than or equal to 3. The Z-score calculation formula is given in Equation 1.

$$Z - score \ = \frac{x - mean}{standard\ deviation}. \tag{1}$$

3.3. Spatio-Temporal 3D-Grid Clustering

After cleaning the data, a Spatio-temporal time series was created by dividing the entire city into uniform grids, each representing a different city zone. Grid side lengths (α) of 700, 1200, and 2000 meters were selected for this study. Using these grid lengths, the city is divided into I x J grids along the latitude and longitude axes, where I and J are calculated using Equations 2 and 3. Similarly, a whole day is divided into several intervals (Δ) like 15min, 30min and 60min, and the number of time steps (t) in a day would be 1440/Δ. Therefore, we have 260640/Δ Timesteps (T) for six months. The demand ($d^t_{(i,j)}$) for each timestamp is then aggregated for each grid (i, j) and represented as a 2D matrix, as shown in Equation 4. Each of these 2D matrices is stacked for each timestamp, therefore forming a 3D matrix of shape (T, I, J). Figure 3 shows the pictorial representation of the Spatio-temporal grid clustering.

$$I \ = \left\lceil \frac{lon_{\max} - lon_{min}}{\alpha} \right\rceil. \tag{2}$$

$$J \ = \left\lceil \frac{lat_{\max} - lat_{min}}{\alpha} \right\rceil. \tag{3}$$

$$D^t_{(I,J)} = \begin{bmatrix} d^t_{(i,j)} & \cdots & d^t_{i,J} \\ \vdots & \ddots & \vdots \\ d^t_{(I,j)} & \cdots & d^t_{(I,J)} \end{bmatrix}, \text{t} = \{1,2, 3,\ldots\text{T}\}, \text{i} \le \text{I}, \text{j} \le \text{J}. \tag{4}$$

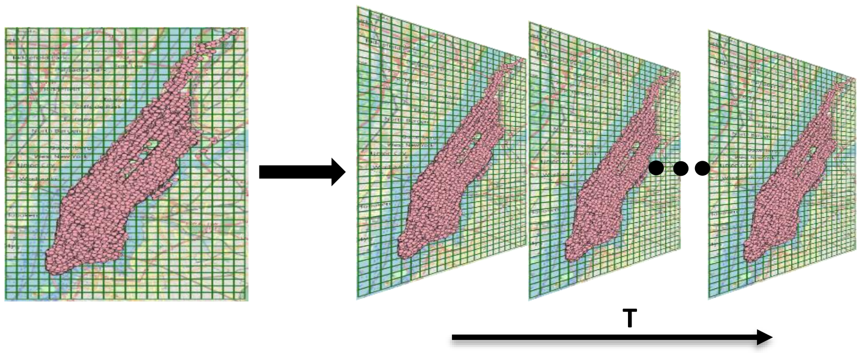


Figure 3. Visualisation of 3D Spatio-Temporal matrix where each cell contains a number representing the aggregated demand of that region at time t

3.4. Deep Learning Models

This study examines the prediction performance of three deep learning models for various spatial and time granularities. The three significant models used are the following:

Long Short-term Memory (LSTM)

Long Short-Term Memory (LSTM) network is a Recurrent neural network (RNN) capable of learning long-term dependencies and handling sequential data. For the model's training, the demand of the city or city zone is converted into a one-dimensional array, and previous t timestamps are given to the model to predict the demand for the next timestamp. The hidden layer used is a Bidirectional LSTM with 128 neurons.

3D-Convolutional Neural Network (3D-CNN)

3D CNN is a convolutional neural network with a temporal component to the convolutional kernel, allowing it to work across a defined timestamp frame. 3D-CNN models employ 3D convolutions, which employ 3-dimensional kernels that slide along the matrix's width, height, and temporal dimensions. In this study, the model's architecture includes four layers of a 3D convolutional layer having 32, 64, 1 and 1 hidden states with the Relu activation function. The 3D convolutional kernel of (3,3,3) for the first three layers and kernel of (window size, 1,1) for the last 3D convolutional layer. Additionally, drop-out layers and batch normalisation are also used.

Temporal-Guided Network (TGNet)

Temporal-Guided Network (TGNet) (Lee *et al.*, 2019) is a baseline for estimating short-term demand. Complex spatiotemporal properties of each region can be extracted through graph networks, and these features are independent of the spatial permutations of neighbouring regions. Instead of relying on long-term demand histories, temporal-guided embedding can learn temporal contexts explicitly and capture temporally recurrent patterns. Six hidden layers are before the fully connected layer, and two layers are used for adding external data sources. A 2D average pooling layer with a 2x2 kernel after the first graph network layer is used for computational efficiency. The number of hidden neurons in the first layer is 32.

3.5. Evaluation Metrics

In this project, we have used four metrics to evaluate the model's performance: Mean Absolute Error (MAE), Symmetric Mean Absolute Percentage Error (SMAPE), Root Mean Square (RMSE) and Normalised RMSE. MAE shows the difference between the predicted and actual values and is mainly used to monitor prediction accuracy. SMAPE gives the percentage error. RMSE, the square root of the Mean Squared Error, gives a broad overview of the generated error. Lastly, NRMSE is calculated by normalising the RMSE with the average of actual values. NRMSE helps us to compare errors among the models since it eliminates the increased error due to different scales of spatiotemporal granularity. The evaluation method of (Lee *et al.*, 2019; Yao *et al.*, 2019) is followed in this paper, where we eliminate those values that are less than the k value. According to the literature, it is a common practice to exclude low-demand values as they are less important to real-world applications. These metrics can be formulated as shown below:

$$MAE = \frac{\sum_{i=1}^{n}|\acute{y}_i - y_i|}{n}. \tag{5}$$

$$SMAPE = \frac{100\%}{n}\sum_{i=1}^{n}\frac{|\acute{y}_i - y_i|}{(\acute{y}_i + y_i)/2}. \tag{6}$$

$$RMSE = \frac{1}{n}\sqrt{\sum_{i=1}^{n}(\acute{y}_i - y_i)^2}\ . \tag{7}$$

$$NRMSE = \frac{RMSE}{\frac{1}{n}\sum_{i=1}^{n} y_i}. \tag{8}$$

4. Experiments

In this section, the experiments are divided into two parts. The aggregated part is where we train the model for the whole city, compare the models' results, and find the best model for the city. The second part is the disaggregated part, where we train the models for each city zone or grid and compare the results to find the best one.

4.1. Experimental setup

Data from the yellow taxi service from January 2015 to June 2015 (a period of six months) was used for training; data from January 2016 to February 2016 (a period of two months) was used for validation, and data from March to June 2016 was used to test the model (4 months). For aggregated and disaggregated parts, models were trained with demand aggregated for time intervals of 15min, 30min, and 60min. For each time interval, models were trained by considering historical records as a prominent feature. Historical records of 4, 6, 12, 24, 72, 96 and 168 hours were selected to understand their effect on the prediction. Although we have data for the entire city of New York, only the Manhattan Borough is considered because 90% of trips were generated there.

LSTM and 3D CNN were trained on Nvidia RTX 3060 6GB, while TGNet was trained in Colab pro plus with Nvidia Tesla P100 16GB memory since the training of six-month data required large computation memory. TGNet and 3D CNN could not train beyond 24 hours' historical demand as features even with this setup. The models for this study were created using the TensorFlow Keras framework in Python 3.9.

4.2. Aggregated

After the Spatio-temporal 3D clustering, the data structure is (T, I, J), where T depends on the time interval (Δ), and I and J depend on the grid lengths (α). When considering the data for the entire city, the demand for each timestep (t) should be the sum of the demands of all the grids (i, j), hence the name Aggregated. CNN and TGNet take 3D data (T, I, J) as input, and the model is trained so that each grid (i, j) will have a prediction based on its temporal characteristics and the spatial neighbours surrounding that grid. In the end, we will have a model capable of predicting the demand of all the grids at a particular timestamp in one go. Therefore, to get the demand of the whole city, we should sum all the demands of each grid (i, j) at timestamp t, and this aggregation might affect the model's performance accuracy. In the case of LSTM, we will have a model which directly gives the city demand at a timestep based on past values.

Performance comparison of models

To compare the chosen models with the selected space and time granularity, a normalised RMSE is used where we divide the average RMSE of the model by the mean demand of the city, which makes the errors comparable. From the plots shown in Figure 4, the effect of historical demand of various lengths is very evident. We can see that for Bidirectional LSTM, the lowest error is obtained when prior one-week values (168 hours) are used for the training. However, we can also observe that there is no significant reduction in the error after the past 24 hours. Among the time intervals, 15 mins are the one which gives the minimum error, and the second-best time interval is 30 minutes for all the chosen models. For 3D CNN and TGNet, the minimum error varies between 6 and 12 hours, depending on the grid length and time interval. When we compare the errors along the grid lengths, we see no relevant increase or decrease in errors. The maximum difference between the minimal errors we can notice is 1%. From all these observations, we can conclude that the city model Bidirectional LSTM with 15 minutes time interval and past one week value is the best model with a percentage error of 3.72%.

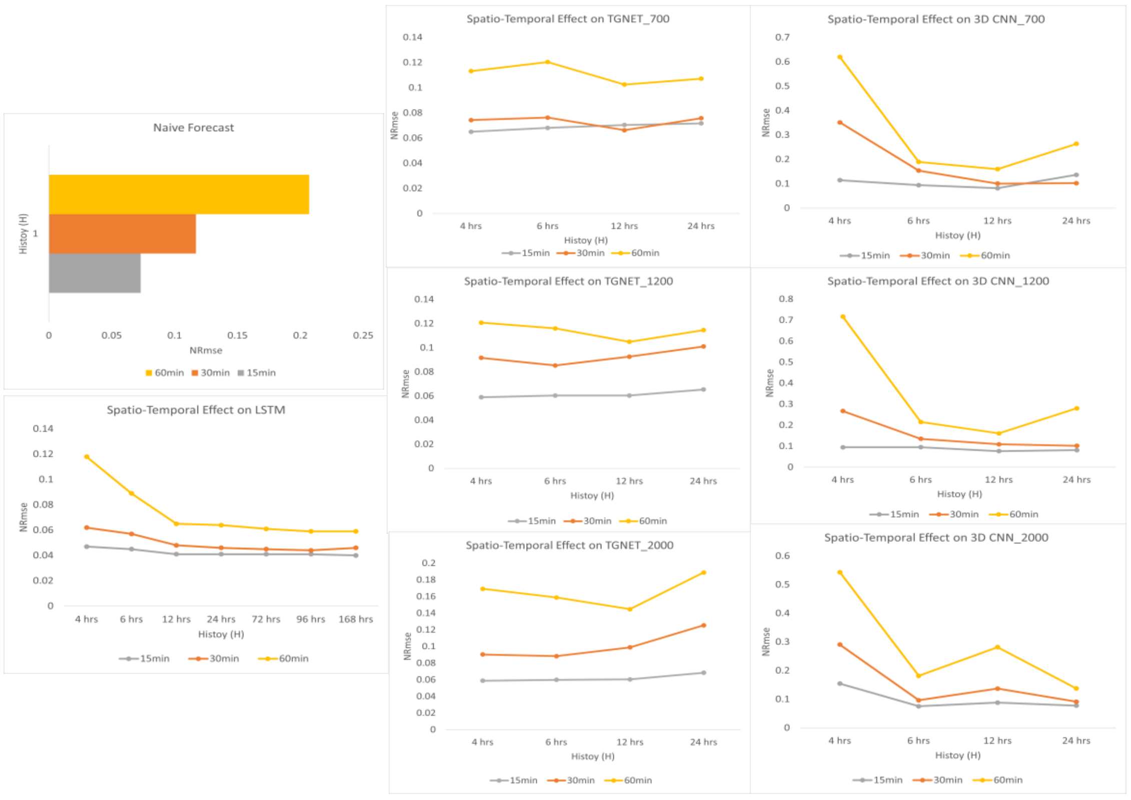


Figure 4. Comparison of normalised RMSE for different models

Table 1 compares these deep learning models with baseline models such as a Na-ïve forecast and a simple linear model, which are trained with demands of 15 minutes time intervals and past 12 hours demand as input features. As observed earlier, Bidirectional LSTM gives the lowest RMSE with a percentage error of 3.883%. The TGNeT is the second-best model for city demand prediction. Bidirectional LSTM outperforms the 3D CNN and TGNet may be due to its strong ability to capture long-term and short-term temporal patterns. Figure 5 shows the forecast of city demand by taking the past 12 hours' demand as input and for a forecast horizon of 15 minutes.

Table 1. Comparison of deep learning models for 15min and past 12 hours demand as features

Model	**RMSE**	**SMAPE (%)**	**Training Time(min:sec)**
Naïve	254.161	7.427	00:00
ML-Linear	207.656	6.289	01:12
Bidirectional-LSTM	**142.130**	**3.883**	**07:04**
TGNET_700	243.253	7.091	27:34
TGNET_1200	209.902	5.425	13:10
TGNET_2000	210.919	4.965	9:09
3D CNN_700	281.662	7.849	180:30
3D CNN_1200	265.727	6.771	30:55
3D CNN_2000	307.944	7.260	14:29

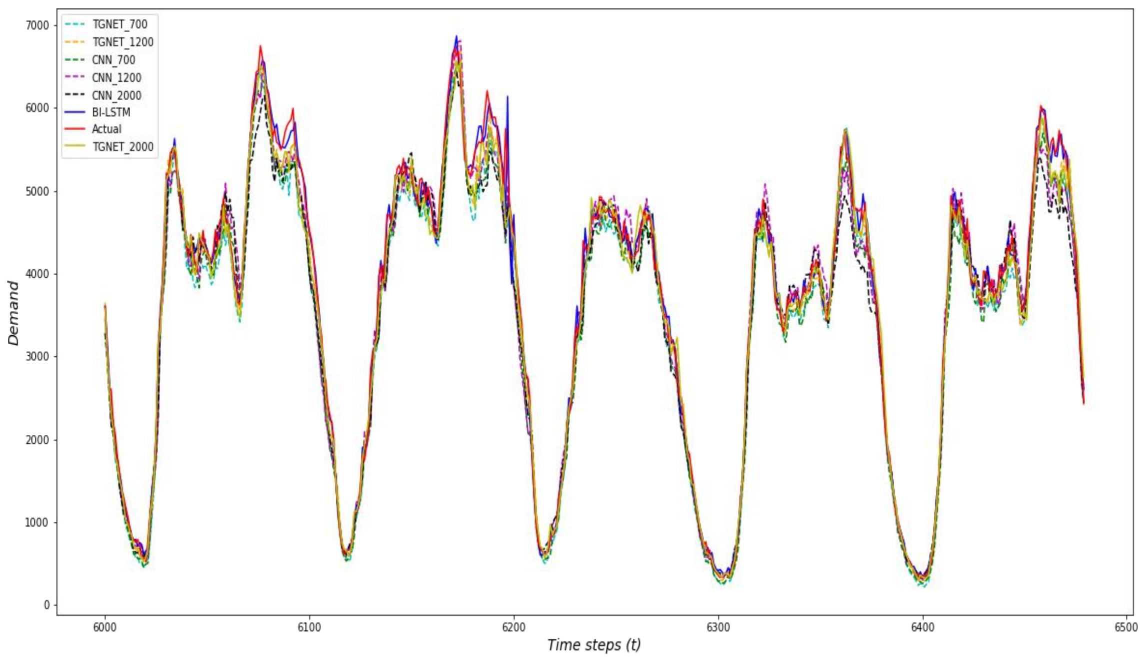


Figure 5. Forecast of deep learning models for a five-day test period for the entire city

4.3. Disaggregated

This section shows the model's performance for city zones (grid (i, j)). The demand for each grid (i, j) is taken from the clustered data and used to build a model for the LSTM. For instance, when we divide the map of Manhattan with a grid length of 2000m, we will have a grid shape of 14x7. Therefore, we will have 98 individual LSTM models representing each grid. In the case of 3D-CNN and TGNet, each chosen grid length has a model and for the comparison, predicted data at gird (i, j) is extracted and compared with the corresponding prediction of LSTM.

Overall performance comparison

In this section, we compare the average error of 3D CNN and TGNet to find the best model. From Figure 6, we can observe that 3D CNN has the same pattern as in section 4.2. However, the range of error has changed due to the change in the average demand of the area. In the aggregated part, the average demand of the city was 8745 trips, and in the disaggregated part, the average demand of each grid was 80 trips. In the case of TGNet, for the area with a smaller grid length of 700m, the 30 minutes time interval achieves the minimum error. As the space-time granularity decreases, the demand distribution becomes more irregular, affecting the model's learning. Hence, the demand distribution in the 30 minutes time interval will be more regular than in 15 minutes, making the model predict with lesser error. The best model among the two is TGNet, with the minimum error for the grid length of 2000m, 15-minute time interval and the past 6 hours as input feature.

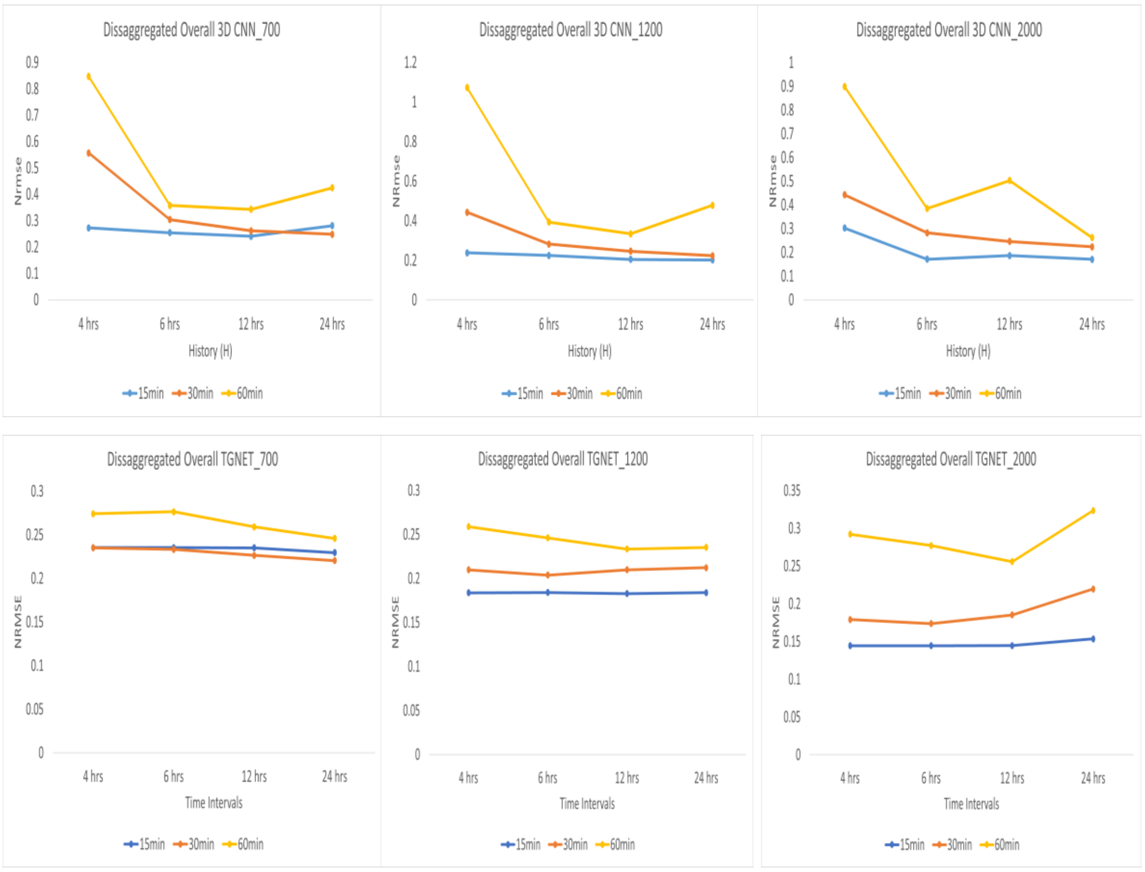


Figure 6. Comparison of normalised RMSE of 3D-CNN and TGNET

Comparison with Bidirectional LSTM

In this study, we selected three high-demand and three medium-demand grids and compared the average NRMSE with the overall NRMSE of TGNet and 3D CNN. For a fair comparison, 3D CNN and TGNet used the same grids used for the LSTM to find the overall NRMSE, as shown in Figure 7. The average NRMSE for the chosen grids matches the abovementioned profile. The plots show that the optimum time interval for Bidirectional LSTM changed to 30 minutes compared to the aggregated model. However, the optimum historical demand remains unchanged, which is 24 hours. Furthermore, the best model in the disaggregated part is the Bidirectional LSTM, with the minimum error for the grid length of 2000m, 30-minute time interval and the past 168 hours as input feature. Table 2 compares the prediction errors of different models for Grid 114 of the map.

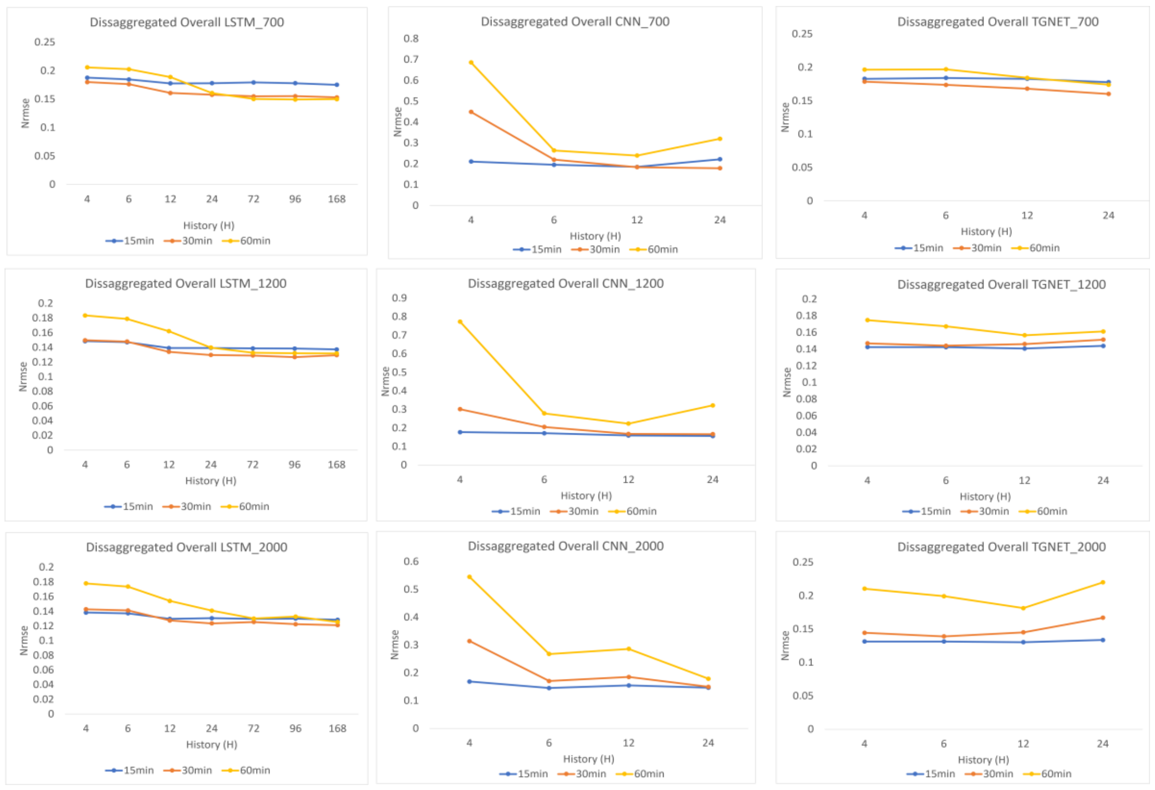


Figure 7. Performance comparison of Bidirectional LSTM with 3D CNN and TGNeT

Table 2. Comparison of deep learning models for 30min and past 12 hours demand as features for Grid 114 of the Map divided with a grid length 1200m

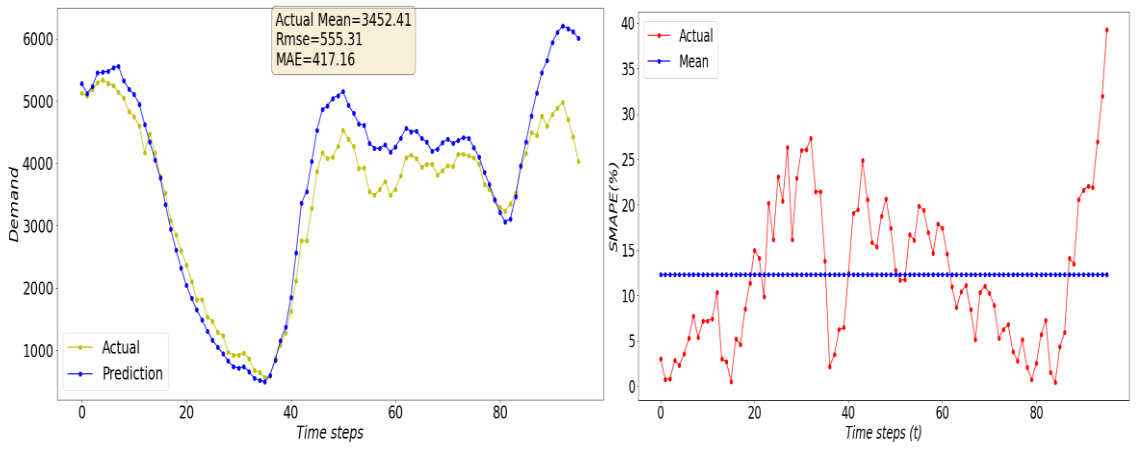
Model	**RMSE**	**SMAPE (%)**	**Training Time(min:sec)**
Naïve	58.64	16.69	00:00
ML-Linear	51.41	13.4	00:53
Bidirectional-LSTM	**43.94**	**11.90**	**1:19**
TGNET_1200	50.89	13.20	06:10
3D CNN_1200	57.74	13.9	09:01

4.4. Error propagation for prediction on prediction

In this section, we selected Bidirectional LSTM for this experiment since it was the model with minimum prediction error.

Bidirectional LSTM city model

In the above sections, we predicted based on the actual values. However, testing the model's performance based on predicted values for real-world applications is essential when we lack actual values for the predictions. For this experiment, we used Bidirectional LSTM trained with 15 minutes time intervals and past 24 hours values as features. Therefore, we have 96 timestamps as input features, and from the error propagation plot, we can observe that the model can forecast with 16 timestamps (4 hours) of predicted values with an average of 4.6% error.



(a) (b)

Figure 8. Error propagation for prediction on predicted values for Bidirectional LSTM for city model for 15 minutes time interval and past 24 hours demand as features. (a) Difference between actual and prediction for one day. (b) Percentage error propagation for one-day prediction

4.5. Predicting city zones with a whole city model

Another interesting experiment was to apply the city model of Bidirectional LSTM for predicting the city zone demand. We created a city model with the optimum configuration for the Bidirectional LSTM obtained from section 4.3 for city zones (with 30 min intervals and past 24 hours demand values as features), and Table 3 compares this model with the individual LSTM model of this zone, TGNeT and 3D CNN. Surprisingly, the results were promising because even though the model did not use the demand of that particular zone for training, it achieved a forecast with an average increasing error of 2.3%. Moreover, as the grid length increases, it obtains a fore-cast error close to the percentage error of TGNet and 3D CNN.

Table 3. Comparison of LSTM city model with individual zone models

Model	**Space (m)**	**Zone**	**Rmse**	**Smape (%)**	**% Increase/Decrease in error**
BI-LSTM	700	217	14.207	16.364	-
BI-LSTM	700	217 (City model)	16.614	19.971	3.606
3D CNN	700	217	14.317	15.267	4.703
TGNet	700	217	13.527	14.989	4.981
BI-LSTM	1200	88	37.264	12.004	-
BI-LSTM	1200	88 (City model)	40.081	12.551	0.547
3D CNN	1200	88	47.370	13.730	-1.178
TGNet	1200	88	42.673	12.609	-0.058
BI-LSTM	2000	52	34.781	12.800	-
BI-LSTM	2000	52 (City model)	43.000	15.712	2.911
3D CNN	2000	52	44.102	13.686	2.026
TGNet	2000	52	47.833	16.128	-0.415

5. Conclusion

In this paper, we explored a combination of space and time granularity, hoping to find an optimum choice for the models. It was interesting to discover different behaviour of deep learning models for space-time granularity. From the error plots, we can observe that the effect of space and time varies with the type and architecture of deep learning models we adopt. We started this paper with four main questions that any researcher would have when dealing with space and time granularity. We can conclude this paper by answering them. From the analysis, we can observe that there is an effect of past demand values affecting the performance of the models. For Bidirectional LSTM, although the optimum value was 168 hours, proving that LSTM can learn long-term temporal patterns, we observed no significant decrease in error after 24 hours of past value. For 3D CNN, the minimum value varies between 6 and 12 hours, depending on the time interval and chosen space. TGNeT was built for short-term demand estimation, which is why the errors increase after 6 hours of past values. The time interval of 15 minutes was the optimum for all the models in the aggregated part, representing whole city demand, and for the disaggregated part, 30 minutes was the optimum, mainly for Bidirectional-LSTM. Finally, we can discover from Table 3 that it is possible to create a whole city model and use it to estimate the demand of city zones with the Bidirectional LSTM model. From this research, we can conclude that even though 3D CNN and TGNet had the advantage of both temporal and spatial correlation, Bidirectional LSTM, with its strong capability to remember long and short-term temporal patterns, outperforms the other models.

Reference

1. Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021) Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, 8(1), 1–74.
2. Gers, F. A., Eck, D., & Schmidhuber, J. (2002) Applying LSTM to time series predictable through time-window approaches. In: *Neural Nets WIRN Vietri-01*, 193–200. Springer.
3. Lara-Benítez, P., Carranza-García, M., & Riquelme, J. C. (2021) An experimental review on deep learning architectures for time series forecasting. *International Journal of Neural Systems*, 31(03), 2130001.
4. LeCun, Y., Bengio, Y., & Hinton, G. (2015) Deep learning. nature, 521(7553), 436-444. *Google Scholar Google Scholar Cross Ref Cross Ref*.
5. Lee, D., Jung, S., Cheon, Y., Kim, D., & You, S. (2019) Demand Forecasting from Spatiotemporal Data with Graph Networks and Temporal-Guided Embedding. *ArXiv Preprint ArXiv:1905.10709*.
6. Liu, K., Chen, Z., Yamamoto, T., & Tuo, L. (2022) Exploring the impact of spatiotemporal granularity on the demand prediction of dynamic ride-hailing. *ArXiv Preprint ArXiv:2203.10301*.
7. Okutani, I., & Stephanedes, Y. J. (1984) Dynamic prediction of traffic volume through Kalman filtering theory. *Transportation Research Part B: Methodological*, 18(1), 1–11.
8. Pan, B., Demiryurek, U., & Shahabi, C. (2012) Utilizing real-world transportation data for accurate traffic prediction. In: *IEEE 12th International Conference on Data Mining*, 595–604.
9. Qi, Y., & Ishak, S. (2014) A Hidden Markov Model for short term prediction of traffic conditions on freeways. *Transportation Research Part C: Emerging Technologies*, 43, 95–111.
10. Schimbinschi, F., Nguyen, X. V., Bailey, J., Leckie, C., Vu, H., & Kotagiri, R. (2015) Traffic forecasting in complex urban networks: Leveraging big data and machine learning. In: *2015 IEEE International Conference on Big Data (Big Data)*, 1019–1024.
11. Williams, B. M., & Hoel, L. A. (2003) Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of Transportation Engineering*, 129(6), 664–672.
12. Wu, F., Wang, H., & Li, Z. (2016) Interpreting traffic dynamics using ubiquitous urban data. In: *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 1–4.
13. Yao, H., Tang, X., Wei, H., Zheng, G., & Li, Z. (2019) Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In: *Proceedings of the AAAI Conference on Artificial Intelligence,* 33(01), 5668–5675.