12-2023

M2-CNN: A macro-micro model for taxi demand prediction

Shih-Fen CHENG
Singapore Management University, sfcheng@smu.edu.sg

Prabod Manuranga RATHNAYAKA MUDIYANSELAGE
Singapore Management University, prabodr@smu.edu.sg

M^2–CNN: A Macro-Micro Model for Taxi Demand Prediction

Shih-Fen Cheng
School of Computing and Information Systems
Singapore Management University
Singapore
sfcheng@smu.edu.sg

Prabod Rathnayaka
Centre for Data Analytics and Cognition
La Trobe University
Melbourne, Australia
P.Rathnayaka@latrobe.edu.au

***Abstract*—In this paper, we introduce a macro-micro model for predicting taxi demands. Our model is a composite deep learning model that integrates multiple views. Our network design specifically incorporates the spatial and temporal dependency of taxi or ride-hailing demand, unlike previous papers that also utilize deep learning models. In addition, we propose a hybrid of Long Short-Term Memory Networks and Temporal Convolutional Networks that incorporates real-world time series with long sequences. Finally, we introduce a microscopic component that attempts to extract insights revealed by roaming vacant taxis. In our study, we demonstrate that our approach is competitive against a large array of approaches from the literature on the basis of detailed moving logs of more than 20,000 taxis and 12 million trips per month over a three-month period. Our analysis of the effectiveness of individual components reveals that microscopic information is essential for generating high-quality predictions.**

1. Introduction

In large cities, taxis (or ride-hailing cars) play a pivotal role in providing point-to-point transportation service, which is complementary to other modes of public transport systems. However, a big problem in operating any type of point-to-point transportation service is the imbalances of taxi supplies and passenger demands. As these imbalances are highly dynamic, and when drivers are left on their own to rebalance themselves based on historical information or their own limited local observations, the resulting system performance would be far away from optimal. A number of researchers have proposed to apply both centralized [1], [2] and decentralized approaches [3] to provide guidance to drivers, and a recent field trial of such system demonstrates great promises in reducing inefficiencies [4].

However, for such systems to be effective, good demand predictions, both immediate and in the near future, need to be provided. This is thus one of the most active areas of research in urban computing. With the advances in deep-learning-based techniques, researchers have begun to model complex demand prediction problems using the deep-learning framework. As illustrated by recent studies such as [5] and [6], we can see that these deep-learning-based approaches greatly outperform traditional approaches, which are mostly based on regression or time series approaches.

In this paper, we continue the investigation of the use of deep learning techniques in advancing the state of the art in predicting taxi demands. There are three major design gaps in the literature that we aim to address. Firstly, most deep-learning-based demand prediction approaches incorporate spatial dependency by using local Convolutional Neural Networks (CNNs), using the fact that demands in a particular region should only depend on "nearby" or "similar" regions. However, no matter how the "closeness" is defined, only demands from the same time period are considered. We argue that this might not be sufficient, as the demands from a neighboring region a couple of time periods earlier could also have an impact on the demand prediction in the current time period. To explicitly consider such dependency, we come up with a three-dimensional variant of the local CNN that explicitly considers the spatial and temporal demand dependency simultaneously.

Secondly, we adopt the insight by [7], who empirically demonstrate that for datasets with long sequence, the widely used Recurrent Neural Networks (RNNs), specifically the Long Short-Term Memory (LSTM) Networks, might not be the best choice; instead, the Temporal Convolutional Networks (TCN) might produce better results. When incorporating long sequence time series from sources other than demand counts, we come up with a hybrid LSTM-TCN design, which combines both the strength of the LSTM network and TCN.

Finally, we extract valuable information from the microscopic movement data of vacant taxis, which is based on the empirical observation that the longer a road link has not seen a visit from a vacant taxi, the more likely a demand would occur. The incorporation of this micro-feature further improves our prediction accuracy.

In summary, we have made the following contributions in our paper:

- We design a three-dimensional spatio-temporal CNN model that explicitly captures both the spatial and temporal correlations among regions for demand predictions.
- We propose a hybrid temporal model that combines both the LSTM networks and the TCN to handle long

979-8-3503-2445-7/23/$31.00 ©2023 IEEE

sequence time series and use it to incorporate a wide variety of real-world time series observations.
- We discover the relationship between vacant taxis' visits to a road link and the likelihood that demands would be generated, and we incorporate this insight via an LSTM network to contribute to the demand prediction.
- With a real-world dataset that contains more than 20,000 taxis and 12 million trips per month in Singapore, we demonstrate that our approach is competitive against a wide variety of baseline approaches from the literature.

2. Related Work

The spatio-temporal prediction has become a trending topic in the urban computing research area. Many researchers have been working in this research area for the past few years yielding interesting research topics such as traffic flow prediction [8], bike flow prediction [9], [10], and taxi demand prediction [5], [6], [11]. The majority of the aforementioned research studies divide an urban area into a grid-like structure and make predictions based on the aggregated grid regions. There have been a few spatial grid types used in the literature to aggregate urban data, such as rectangular shape [5], [6] and hexagon shape [12], [13]. In previous studies, a grid structure is used to represent spatial urban data, and convolution neural networks are used to model local and spatial correlations between regions and extract spatial features of a certain region [5], [14]. Recurrent Neural Networks (RNNs), specifically Long Short-Term Memory (LSTM) Networks [15], [16] are used to model temporal patterns and sequences.

In the domain of taxi demand prediction, [5] have recently proposed the DMVST-Net approach which makes use of multiple views, i.e., spatial, temporal, and semantic views for the taxi demand prediction. DMVST-Net mainly separates each view into a sub-model and extracts features from each model to generate predictions. Similarly, for the traffic speed prediction studies, [14], [17], a CNN is trained for the whole city and predictions are made on regions. These previous studies use residual CNN on the images of traffic flow considering both the spatial and the temporal dimensions, yet none of them consider the spatial-temporal dependencies explicitly.

Another thread of related study is on the use of CNNs in sequence learning. A recent empirical study in sequence learning by [7] reveals that for long sequence learning tasks, the CNNs are better than the RNNs. The CNN framework adopted by [7] is called the Temporal Convolutional Network (TCN) and is shown to perform better than RNNs for problems such as the adding problem, sequential MNIST, P-MNIST problem, and copy memory problem. We adopt their insights in handling exogenous time series datasets.

From the literature review, we can see that the major gap in the spatio-temporal prediction literature is the lack of explicit consideration of spatio-temporal dependencies and the utilization of CNN for real-world time series. Almost all past works on taxi demand prediction also do not consider the information embedded in the movement of vacant taxis. These gaps are the main contributions we intend to make to the literature.

3. The M^2–CNN Framework

In this section, we describe the design of our M^2–CNN framework for the taxi demand prediction. When describing the framework, we try to focus on the general design principle and not the scenario-specific features. The details of all the scenario-specific features will be provided in the experiment section.

3.1. Background

As in most past works on taxi (or ride-hailing) demand prediction, we discretize both the spatial and temporal dimensions using fixed interval sizes. For the spatial dimension, we define the unit grid region to be 1km by 1km; the unit grid regions are mutually exclusive, and collectively they cover all the city areas that we would want to generate demand predictions for. For the temporal dimension, we define the unit time period to be 15 minutes long.

We denote the unit grid n as l_n and let the set $L = \{l_1, ..., l_n, \ldots, l_N\}$ be the collection of all unit grids. Similarly, we denote the time period m as t_m and let the set $T = \{t_1, ..., t_m, \ldots, t_M\}$ be the collection of all time periods. Almost all features included in our model are aggregated into a particular (n, m) tuple (grid l_n, time period t_m).

Our framework contains three major components, as illustrated in Fig. 1. We will describe them in greater detail next.

3.2. 3D-CNN: 3D-Spatio-Temporal Convolutional Neural Network

In the deep-learning-based taxi demand prediction literature, the spatial relationships are usually captured by using a convolutional neural network (CNN) (e.g., see [5]). The basic idea is to treat the demand prediction problem as an image recognition problem, where each pixel stores demand-related information of a grid region in its RGB channels.

The critical design decision of this approach is on what grid regions to include in the image for each n. The most naive design is to include all grid regions; however, this will result in a very large image, and the prediction quality, as a result, will deteriorate (as pointed out by [5]). A better approach is to include only the relevant grid regions. This is where domain knowledge comes into play, and there are several different ways of identifying relevant grid regions. For example, in [5], the authors apply the proximity principle, and for each grid region n, all grid regions that are within 3 unit grids of Chebyshev distance from the grid n are included. Their image size is thus 7 by 7. In [6], the authors have proposed another approach, which uses a

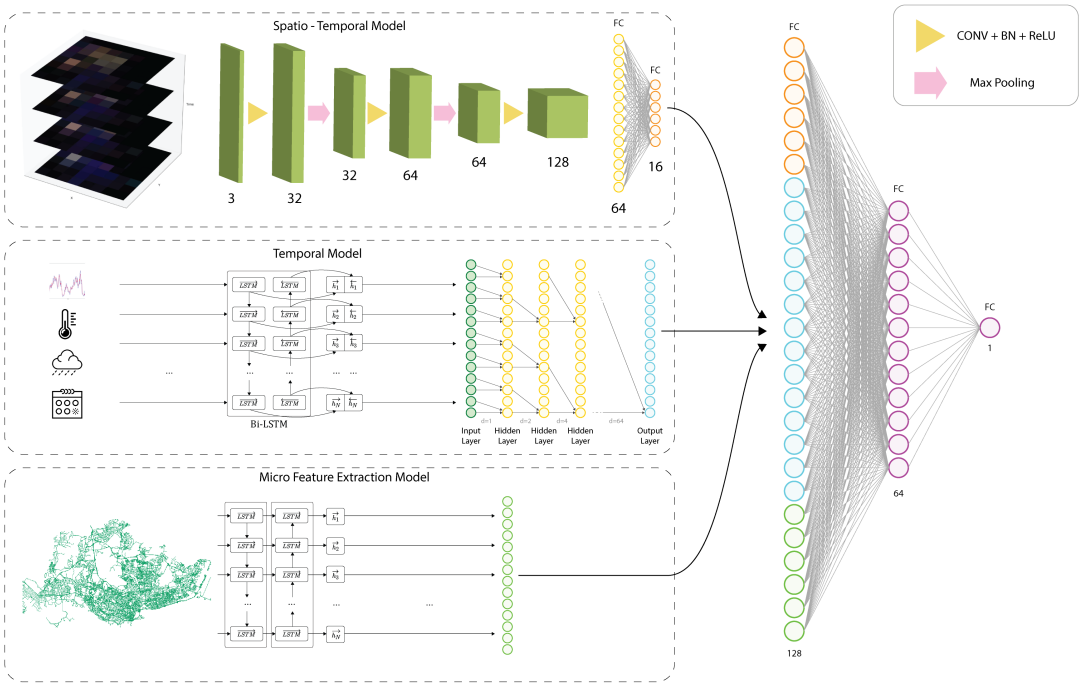

Figure 1. High-level design of our M^2–CNN model.

connectivity graph that reflects physical express road linkage to capture the *closeness* of any two regions. The composed image is thus based on the strength of linkages. In both designs (which are both competitive), the authors decide to encode only demand information that comes from the same time period m.

Our design is based on the principle of proximity, yet we also consider the temporal correlation; i.e., when the demand occurrence in region A from an earlier time period $m - k$ is correlated with the demand occurrence in region B in time m.

To realize this design while still using the CNN framework, we define our image to be of 3 dimensions, (x, y, m): (x, y) refer to the grid region's spatial location, and m refers to the temporal dimension, indicating the number of time periods from the current time.

As shown in Fig. 2, for each grid region n, we construct an $K \times K \times h$ image to encode all information on demand and supply counts. The parameter h represents how far into the past we would want to include. And K specifies the size of the included neighborhood around the region n. In our implementation, we use $h = 16$ and $K = 9$ (i.e., we include all regions that are within 4 unit grids of Chebyshev distance).

For a pixel (x, y, m) in the 3D image, we store the information related to the region (x, y) in time m in channels R, G, and B. For channel R, we encode the number of trips originating from that region; for channel G, we encode the number of trips ending in that region; finally, for channel B, we encode the number of vacant taxis in that region (during the time interval).

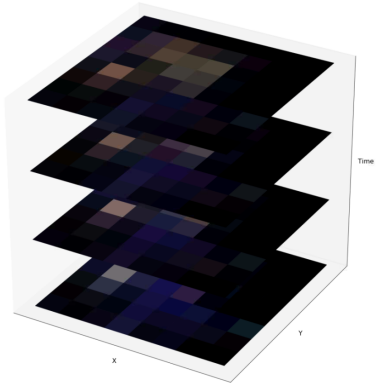

Figure 2. An illustration of 3-dimensional neighborhood structure capturing both spatial and temporal relationships.

We use 0 as the padding value for regions that are at the boundary. As a result, we end up with a tensor $Q_m^n \in \mathbb{R}^{\mathrm{K}\times\mathrm{K}\times\mathrm{h}\times 3}$, for each location l_n and time interval t_m.

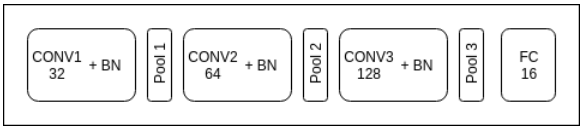

Figure 3. The 3D-CNN has 3 convolutions, 3 max-pooling, and 1 fully connected layer.

Tensor Q_m^n is passed to a deep 3D-CNN described in Fig. 3 having a depth of p. All 3D convolution filters are $3 \times 3 \times 3$ with stride $1 \times 1 \times 1$. All 3D pooling layers are $2 \times 2 \times 2$ with stride $2 \times 2 \times 2$. Rectified Linear Unit (ReLU) activation function is used for each convolution layer.

3.3. LSTM-TCN: A Hybrid Model of Long Short-Term Memory and Temporal Convolutional Networks

The 3D-CNN component handles all information directly related to the demand occurrence. In this section, we propose to incorporate various exogenous time series that could potentially have an impact on the demand occurrence (we list all included data sources in the Experiment section).

To handle sequential data, the neural network model of choice is usually the Recurrent Neural Network (RNN). However, a recent study by [7] demonstrates that the Temporal Convolutional Network (TCN) sometimes outperforms RNN in certain sequence modeling tasks.

In our framework, we propose a hybrid model that combines both the RNN and TCN, to take advantage of the strengths of both methods. As we incorporate a large number of data sources, the features we choose to include tend to be rather noisy. The RNN models work well with these noises and we use a popular RNN architecture, the Long Short-Term Memory (LSTM) network [18], to perform encoding and automatic feature selection. The TCN then makes use of the encoded features to model the taxi demands as a sequence.

All RNNs have a chain of repeating modules of neural networks. The LSTM networks also have this chain-like structure. A unit in a typical LSTM network contains four unique components: the cell, the input gate, the forget gate, and the output gate. While the state is kept in the cell, the three gates control the flow of information.

Conceptually speaking, given a previous cell state c_{m-1}, the previous hidden state h_{m-1}, and the input x_m, the LSTM network has to decide what new information to store in the cell state. To achieve this, the *input gate* (a *sigmoid* layer) decides which values to update, and processes these values using the *tanh* layer. The resulting vector of candidate values, $\tilde{c}_m$, is then added to the cell state. Before updating the cell state with $\tilde{c}_m$, the *forget gate* also needs to decide which part of c_{m-1} to forget. Finally, the *output gate* generates a filtered version of the new cell state c_m. The governing equations of the LSTM are listed below:

$$f_m = \sigma(W_{if} x_m + W_{hf} h_{m-1} + b_f), \quad (1)$$
$$i_m = \sigma(W_{ii} x_m + W_{hi} h_{m-1} + b_i), \quad (2)$$
$$o_m = \sigma(W_{io} x_m + W_{ho} h_{m-1} + b_o), \quad (3)$$
$$\tilde{c}_m = \tanh(W_{i\tilde{c}} x_m + W_{h\tilde{c}} h_{m-1} + b_{\tilde{c}}), \quad (4)$$
$$c_m = f_m \circ c_{m-1} + i_m \circ \tilde{c}_m, \quad (5)$$
$$h_m = o_m \circ \tanh(c_m), \quad (6)$$

where $\circ$ is the operator for element-wise product, σ is the *sigmoid* function and *tanh* is hyperbolic tangent function.

After the input x_m is sent through two LSTM layers, the sequence of the latent representations, $(h_{m-k}, ..., h_m)$, are considered as the inputs to the TCN.

The TCN uses the Dilated Convolutions to accommodate an exponentially large receptive field. For an 1-D sequence input $\mathbf{h} \in \mathbb{R}^n$ and a filter $f : \{0, \ldots, k-1\} \to \mathbb{R}$, the dilated convolution operation F on the element s of the sequence is defined as:

$$F(s) = (\mathbf{h} *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot \mathbf{h}_{s-d \cdot i}, \quad (7)$$

where d is the dilation factor, k is the filter size, and $(s - d \cdot i)$ accounts for the direction of the past. A residual block contains a branch leading out to a series of transformations, $\mathcal{F}$, whose outputs are added to the input h of the block:

$$w = \text{Activation}(h + \mathcal{F}(h)). \quad (8)$$

In our later experiments, we used one LSTM layer of size 128, and a TCN with residual blocks, containing $k = 128$ filters and dilation factors $d = 1, 2, 4, 8, 16, 32, 64$ (as illustrated in Fig. 4).

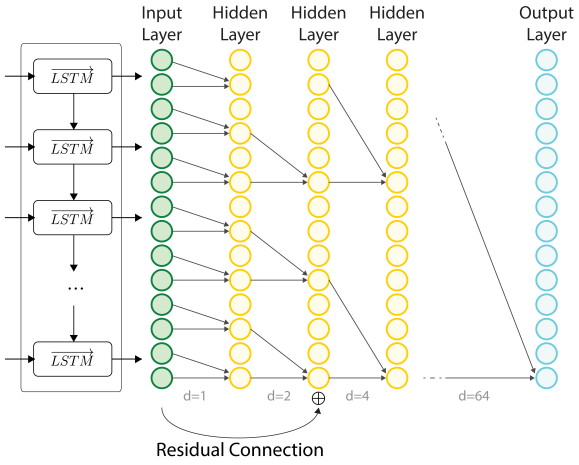

Figure 4. The LSTM-TCN architecture. One LSTM layer followed by a TCN with dilated causal convolution with dilation factors $d = 1, 2, 4, 8, 16, 32, 64$, and filter size $k = 2$, and residual connections.

3.4. Micro-Feature Model

A unique design of our framework is the incorporation of the vacant taxi's microscopic movement. We introduce this module to capture the hidden information in vacant taxi's movement: when a vacant taxi enters and exits a road link without status change, it implies that no street-hail demands are observed along that road link. At the road link level, an extension to this observation is the strong positive correlation between the time elapsed since the last visit by a vacant taxi and the likelihood that the next incoming taxi would discover a demand.

To incorporate this insight in our prediction framework, we first identify road links that are worth monitoring (we

only monitor road links that generate at least 600 demands per month; in aggregate these road links generate around 70% of all street-hail demands). After identifying these road links, we monitor the arrival of vacant taxis to these links and update the elapsed time since the last visit by a vacant taxi (the elapsed time increases as time progresses, but resets when a vacant taxi arrives).

For the region of interest, we collect elapsed times of all monitored links in this region, and together with the summary statistics of recent elapsed time observations, we send them to two stacked LSTM layers. The sequence of the latent representations $(h_{m-k}, ..., h_m)$ were extracted as features for the fully connected layer.

3.5. Model Fusion

Fully connected layers are used to fuse all three models together. We fuse output u_m^l from the 3D-CNN model, v_m^l from the hybrid LSTM-TCN model, and w_m^l from the Micro Feature model to form a tensor:

$$z_m^l = u_m^l \oplus v_m^l \oplus w_m^l. \tag{9}$$

We feed z_m^l to the 2-layer fully connected network, and finally to a sigmoid layer to get the final prediction value $\hat{y}_{m+1}^l$

$$\acute{y}_m^l = ReLU(W_{f_1} z_m^l + b_{f_1}), \tag{10}$$

$$\tilde{y}_m^l = ReLU(W_{f_2} \acute{y}_m^l + b_{f_2}), \tag{11}$$

$$\hat{y}_{m+1}^l = \sigma(W_{f_3} \tilde{y}_m^l + b_{f_3}). \tag{12}$$

where $\{W_{f_1}, W_{f_2}, W_{f_3}\}$ and $\{b_{f_1}, b_{f_2}, b_{f_3}\}$ are learned parameters, and σ denotes the $sigmoid$ activation function. The final prediction value $\hat{y}_{t+1}^i$ is in $[0, 1]$, since we scale our inputs also to the range of $[0, 1]$.

3.6. Loss Function

When designing our prediction model, we want it to work well for both high-demand and low-demand scenarios. However, most available loss functions are scale-dependent, making them sensitive to outliers; yet the taxi demand time series tends to be highly volatile. To handle this concern, we define a hybrid loss function that combines both the Root Mean Squared Error (RMSE) and the Symmetric Mean Percentage Error (SMAPE) since RMSE is more sensitive to larger values and SMAPE is insensitive to outliers. The resulting loss function is thus defined as:

$$J(\Theta) = \sum_{i=1}^{N} \left(\alpha \sqrt{(y_{t+1}^i - \hat{y}_{t+1}^i)^2} + \beta \frac{|y_{t+1}^i - \hat{y}_{t+1}^i|}{(|y_{t+1}^i| + |\hat{y}_{t+1}^i|)/2} \right),$$

where Θ represents all learnable parameters in the model, and α, β are adjustable hyperparameters.

4. Experiments

To demonstrate the effectiveness of our demand prediction framework, we use a real-world dataset from Singapore which contains detailed taxi movement logs of all taxis (the fleet size is around 20,000) from January to March in 2018. We use January and February data for training and March data for testing. On average there are around 12 million taxi trips per month. We divide Singapore into 626 1km-by-1km grid regions, and we set the time interval length to be 15 minutes. For each prediction, the past 16 time intervals (4 hours) are used.

Besides the number of taxi trips and count of taxis in a region (both are extracted directly from the taxi fleet dataset), we also incorporate the following features that are derived from other data sources:

- **Temporal features**: For each data point, we label it with the following temporal features: the day of the week, the day of the month, the hour of the day, the minute of the hour, indicator of whether the day is a weekday, and whether the day is a public holiday. All these features are categorical.
- **Meteorological features**: For each data point (l_n, t_m), we also include meteorological features such as average temperature, UV index, humidity, and rainfall. Each feature is represented as a numerical value. As there are 62 weather stations across Singapore, for each grid region, we assume that its meteorological features are provided by the closest weather station.
- **Microscopic movement features**: This set of features is derived from the microscopic movement of vacant taxis. As described in the previous section, the most important feature we extracted from the microscopic movement data is the "elapsed time since last visit by a vacant taxi".
- **Polynomial features**: Polynomial features are artificially generated. We generate a new feature matrix consisting of all polynomial combinations of the features with degrees less than or equal to the specified degree. For example, if an input sample is two-dimensional and of the form $[a, b]$, the degree-2 polynomial features are $[1, a, b, a^2, ab, b^2]$. In our experiments, we use degree-2 polynomial features.

In Table 1 we list all features included in our prediction framework.

4.1. Preprocessing, Feature Engineering, and Hyperparameters

All the demand values and continuous numerical features are scaled to [0,1] using $min\text{-}max$ normalization for the training process. Binary features like weekend/weekday indications are encoded as 0 or 1. The outputs of the prediction model are in the range of [0,1], which will be rescaled to get the absolute demand values.

Feature engineering techniques are used to enrich the dataset. In particular, the polynomial feature engineering

TABLE 1. LIST OF FEATURES FROM DIFFERENT SOURCES.

Feature Type	Feature Description
Meteorological	• Average temperature in the time interval • Average UV index in a time interval • Average humidity in a time interval • Cumulative rainfall in a time interval
Temporal	• Hour of day • Minute of hour • Day of week • Day of month • Whether the day is a weekday • Whether the day is a holiday
Taxi Related	• # of trips starting in the target region • # of trips starting in the immediate neighbors of the target region • # of trips ending in the target region • # of trips ending in the immediate neighbors of the target region • # of free taxis

approach is used to grow 29 features to 464 features. These new features include interaction features and self-multiplications.

We set $9 \times 9 \times 16$ as the image size for the 3D-CNN, i.e., 16 images covering the same geographical area (9km$\times$9km) over the past 16 time periods are stacked together to form a 3D image. The 3D-CNN consists of 3 convolutions and 3 max pooling operations. Each convolution has 32, 64, and 128 filters respectively. Filter size is set to $3 \times 3 \times 3$. The pooling layer size is set to $2 \times 2 \times 2$ with stride $2 \times 2 \times 2$. Batch normalization is used after each convolution layer. The LSTM Layer of size 128 is used with a residual-TCN with 128 filters and dilation factors of 1, 2, 4, 8, 16, 32, and 64 for the LSTM-TCN model. For the micro-feature model, we use two stacked LTSMs of size 128. For model fusion, we use two fully connected layers of size 128 and 64 respectively. The $sigmoid$ activation function is used to get the final output and the ReLU activation function is used at every other instance. Batch size is set to 16 and early stopping is used to control the over-fitting of the model. The previous 16 demand values (4 hours) are considered as the sequence length.

4.2. Evaluation Metric

We use Rooted Mean Square Error (RMSE), Symmetric Mean Average Percentage Error (SMAPE), and Mean Average Percentage Error (MAPE) to evaluate our prediction algorithm:

$$\text{RMSE} = \sqrt{\frac{1}{\varepsilon}\sum_{l=1}^{\varepsilon}(y_{m+1}^{l} - \hat{y}_{m+1}^{l})^2}, \tag{13}$$

$$\text{SMAPE} = \frac{1}{\varepsilon}\sum_{l=1}^{\varepsilon}\frac{|y_{m+1}^{l} - \hat{y}_{m+1}^{l}|}{(|y_{m+1}^{l}| + |\hat{y}_{m+1}^{l}|)/2}, \tag{14}$$

$$\text{MAPE} = \frac{1}{\varepsilon}\sum_{l=1}^{\varepsilon}\frac{|y_{m+1}^{l} - \hat{y}_{m+1}^{l}|}{y_{m+1}^{l}}, \tag{15}$$

where y_{m+1}^{l} and $\hat{y}_{m+1}^{l}$ are the actual and predicted demand values in region l at time $m+1$, and ε is the total number of samples.

4.3. Competing Approaches

We compared our prediction approach against the following approaches from the literature. All hyperparameters (if any) in these approaches are tuned to maximize the results.

- **Historical Average (HA)**: We employ the historical average of the demand values to predict the demand value in the next time interval.
- **Linear Regression (LR)**: Linear Regression is a classical statistical approach to model the linear relationship between a scalar response (or dependent variable) and one or more explanatory variables.
- **Support Vector Regression (SVR)**: Support Vector Regression (SVR) is a regression version of the Support-Vector Machine.
- **XGBoost**: XGBoost is a widely used gradient boosting framework.
- **Multi-Layer Perceptron (MLP)**: Multi-Layer Perceptron is a class of feedforward artificial neural network (ANN).
- **Auto-Regressive Integrated Moving Average with Weekday/Weekend Indicator (ARIMA)**: In time series analysis, an autoregressive integrated moving average model is a generalization of an autoregressive moving average model. ARIMA is widely used for time series prediction tasks. Here we have used an ARIMA Model with different models for weekdays and weekends. We use the Auto-ARIMA method in R to automatically select the best orders for the ARIMA model.
- **Deep Multi-View Spatial-Temporal Networks (DMVST-Net)**: Deep Multi-View Spatial-Temporal Networks (DMVST-Net) is a deep learning-based approach that uses multiple views of data to model and predict taxi demand [5]. CNN is used for spatial features, LSTM is used for temporal features, and Graph Embedding is used to capture semantic features. These three views are then integrated to produce the final prediction. DMVST-Net is one of the most recent state-of-the-art deep-learning-based methods in ride-hailing demand prediction.

For fair comparison, we used the same set of features that we used in our model to compare these approaches. The same loss function is used where applicable.

We summarize the performance comparison in Table 2. Our approach outperforms all other competing approaches across all metrics. To highlight the advantage of our approach over other baselines, we compute the performance gap of the baselines against our approach in percentage in Table 3. From Table 3 we can see that when compared against the best baseline, DMVST-Net, the advantage of our approach increases with the demand density: the higher the demand, the larger the performance gap.

TABLE 2. PERFORMANCE COMPARISON AGAINST WELL-KNOWN BASELINES FROM THE LITERATURE. WE PERFORM COMPARISONS UNDER DIFFERENT DEMAND PROFILES: LOW, MEDIUM, AND HIGH, REFERRING TO THE PERCENTILE OF DEMANDS AT BELOW 25%, 25% – 75%, AND ABOVE 75% RESPECTIVELY. THE BEST-PERFORMING MODEL IS HIGHLIGHTED IN BOLDFACE FOR EACH COLUMN.

	RMSE			MAPE			SMAPE		
	Low	Medium	High	Low	Medium	High	Low	Medium	High
HA	6.8392	10.6237	15.2724	2.8719	3.1438	3.762	0.9912	1.101	0.9986
LR	2.4078	4.5630	7.1004	1.0307	0.9745	0.4142	0.7210	0.5591	0.3832
SVR	1.2218	2.8600	8.2159	0.7951	0.8463	0.6050	0.5177	0.4539	0.3227
ARIMA	1.0607	2.2094	5.4988	0.5380	0.4555	0.2480	0.5925	0.3848	0.2138
XGBoost	1.0670	2.1769	5.4163	0.5108	0.4371	0.2403	0.5852	0.3662	0.2053
MLP	1.1328	2.3367	6.1637	0.5758	0.4934	0.2772	0.6142	0.4103	0.2389
DMVST-Net	1.0189	2.1676	5.5287	0.4540	0.4254	0.2384	0.5091	0.3511	0.2074
M^2-CNN	**1.0166**	**2.0740**	**5.1228**	**0.4487**	**0.4139**	**0.2273**	**0.5084**	**0.3426**	**0.1957**

TABLE 3. THE ADVANTAGES OF OUR APPROACH AGAINST BASELINES IN PERCENTAGES.

	RMSE			MAPE			SMAPE		
	Low	Medium	High	Low	Medium	High	Low	Medium	High
HA	572.8%	412.2%	198.1%	540.0%	659.6%	1555.1%	95.0%	221.4%	410.3%
LR	136.8%	120.0%	38.6%	129.7%	135.4%	82.2%	41.8%	63.2%	95.8%
SVR	20.2%	37.9%	60.4%	77.2%	104.5%	166.2%	1.8%	32.5%	64.9%
ARIMA	4.3%	6.5%	7.3%	19.9%	10.1%	9.1%	16.5%	12.3%	9.2%
XGBoost	5.0%	5.0%	5.7%	13.8%	5.6%	5.7%	15.1%	6.9%	4.9%
MLP	11.4%	12.7%	20.3%	28.3%	19.2%	22.0%	20.8%	19.8%	22.1%
DMVST-Net	0.2%	4.5%	7.9%	1.2%	2.8%	4.9%	0.1%	2.5%	6.0%

TABLE 4. PERFORMANCE COMPARISON OF DIFFERENT MODEL VARIANTS.

	RMSE			MAPE			SMAPE		
Model Variant	Low	Medium	High	Low	Medium	High	Low	Medium	High
3D-CNN	1.1131	2.1445	4.6725	0.506	0.3501	0.2243	0.4838	0.4299	0.2628
3D-CNN+LSTM-TCN	1.1327	2.1742	4.6967	0.5104	0.3537	0.2284	0.4841	0.4354	0.2727
3D-CNN+Micro	1.111	2.1363	4.5756	0.4951	0.348	0.2233	0.4833	0.4273	0.2647
M^2-CNN	**1.1021**	**2.1319**	**4.5554**	**0.4849**	**0.3477**	**0.2227**	**0.4817**	**0.4252**	**0.2623**

TABLE 5. THE ADVANTAGES OF THE COMPLETE MODEL AGAINST ALL MODEL VARIANTS IN PERCENTAGES.

	RMSE			MAPE			SMAPE		
Model Variant	Low	Medium	High	Low	Medium	High	Low	Medium	High
3D-CNN	1.0%	0.6%	2.6%	4.4%	0.7%	0.7%	0.4%	1.1%	0.2%
3D-CNN+LSTM-TCN	2.8%	2.0%	3.1%	5.3%	1.7%	2.6%	0.5%	2.4%	4.0%
3D-CNN+Micro	0.8%	0.2%	0.4%	2.1%	0.1%	0.3%	0.3%	0.5%	0.9%

4.4. Contributions by Different Components

To understand the contributions of different components in our approach, we also experiment with the following variants of our prediction model:

- **3D-CNN**: In this model variant we only use 3D-CNN to predict the taxi demand.
- **3D-CNN + LSTM-TCN**: In this model variant we used both 3D-CNN and LSTM-TCN by combining model outputs to predict taxi demand. Combining spatio-temporal features with sequence modeling gives better performance than purely 3D-CNN.
- **3D-CNN + Micro-Feature Model**: This variant uses both 3D-CNN and Micro-Feature Model to predict the taxi demand.
- **M^2-CNN**: This is the complete version of our prediction model.

Similar to Table 3, we also compute the performance gaps in percentage and present them in Table 5.

From the summary of the performances in Table 4 and Table 5, we can see that for the medium-demand and high-demand scenarios, the micro-feature component is the most influential and it complements 3D-CNN nicely. This finding validates our conjecture that microscopic observations could potentially bring significant values to the demand prediction.

5. Conclusions and Future Work

In this paper, we propose the M^2–CNN model as the framework for computing taxi demand prediction. Our framework considers both spatial and temporal relationships among regions, and we devise a new hybrid LSTM-TCN model for handling long time series data. Finally, we also incorporate the microscopic observations on the movement of vacant taxis to further improve demand prediction.

The numerical studies are based on a large-scale real-world taxi dataset containing more than 20,000 taxis and 12 million trips per month over a three-month period in Singapore. Through the experiments, we demonstrate that

our approach outperforms a wide array of approaches from the literature. We also discover that the performance gap between baselines and our approach widens as the demand density increases. Coupling with the observation that the micro-feature model seems to be most influential, we demonstrate that the inclusion of microscopic observations could significantly improve the prediction of taxi demands in dense cities. This shows the importance of incorporating information related to demand or supply, whether this information is explicit or implicit.

To further improve the performance of demand prediction, we could look for other not-so-obvious information sources that are consequential. For example, [19] has explored the use of real-time surge pricing of ride-hailing platforms in predicting taxi demands, and demonstrated that such pricing information could significantly improve the accuracy of demand prediction. In the area of urban planning, [20] has established that the changes in retail amenities in a neighborhood can also permanently alter the taxi demand patterns. This suggests that regional urban features (such as building types and facilities) could be a useful sources in predicting demands as well.

Acknowledgment

This research is supported by the Ministry of Education, Singapore, under its Social Science Research Thematic Grant (MOE Reference Number: MOE2020-SSRTG-018).

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

References

[1] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "T-Drive: Enhancing driving directions with taxi drivers' intelligence," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 220–232, 2011.

[2] M. Lowalekar, P. Varakantham, and P. Jaillet, "Online spatio-temporal matching in stochastic and dynamic domains," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 3271–3277.

[3] P. Varakantham, S.-F. Cheng, G. Gordon, and A. Ahmed, "Decision support for agent populations in uncertain and congested environments," in *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[4] S.-F. Cheng, S. S. Jha, and R. Rajendram, "Taxis strike back: A field trial of the driver guidance system," in *Seventeenth International Conference on Autonomous Agents and Multiagent Systems*, 2018, pp. 577–584.

[5] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[6] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

[7] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[8] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2014.

[9] D. Singhvi, S. Singhvi, P. I. Frazier, S. G. Henderson, E. O'Mahony, D. B. Shmoys, and D. B. Woodard, "Predicting bike usage for new york city's bike sharing system," in *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[10] D. Chai, L. Wang, and Q. Yang, "Bike flow prediction with multi-graph convolutional networks," in *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2018, pp. 397–400.

[11] J. Ye, L. Sun, B. Du, Y. Fu, X. Tong, and H. Xiong, "Co-prediction of multiple transportation demands based on deep spatio-temporal neural network," in *Twenty-Fifth ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 305–313.

[12] Y. Yang, A. Heppenstall, A. Turner, and A. Comber, "Examining the impact of new metro services on dockless bike sharing mobility patterns."

[13] J. Ke, H. Yang, H. Zheng, X. Chen, Y. Jia, P. Gong, and J. Ye, "Hexagon-based convolutional neural network for supply-demand forecasting of ride-sourcing services," *IEEE Transactions on Intelligent Transportation Systems*, 2018.

[14] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[15] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, 2013.

[16] J. Xu, R. Rahmatizadeh, L. Bölöni, and D. Turgut, "Real-time prediction of taxi demand using recurrent neural networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2572–2581, 2017.

[17] J. Zheng and L. M. Ni, "Time-dependent trajectory regression on road networks via multi-task learning," in *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] S. Agarwal, B. Charoenwong, S.-F. Cheng, and J. Keppo, "The impact of ride-hail surge factors on taxi bookings," *Transportation Research Part C: Emerging Technologies*, vol. 136, p. 103508, 2022.

[20] K. O. Lee and S.-F. Cheng, "Neighborhood retail amenities and taxi trip behavior: A natural experiment in singapore," *Habitat International*, vol. 131, p. 102714, 2023.