Passenger Demand Forecasting with Multi-Task Convolutional Recurrent Neural Networks

Lei Bai[1], Lina Yao[1], Salil S. Kanhere[1], Zheng Yang[2], Jing Chu[2], and Xianzhi Wang[3]

[1] School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
baisanshi@gmail.com, {lina.yao,salil.kanhere}@unsw.edu.au
[2] School of Software, Tsinghua University, Beijing, China
hmilyyz@gmail.com, j-zhu16@mails.tsinghua.edu.au
[3] School of Software, University of Technology Sydney, Sydney, Australia
sandyawang@gmail.com

Abstract. Accurate prediction of passenger demands for taxis is vital for reducing the waiting time of passengers and drivers in large cities as we move towards smart transportation systems. However, existing works are limited in fully utilizing multi-modal features. First, these models either include excessive data from weakly correlated regions or neglect the correlations with similar but spatially distant regions. Second, they incorporate the influence of external factors (e.g., weather, holidays) in a simplistic manner by directly mapping external features to demands through fully-connected layers and thus result in substantial bias as the influence of external factors is not unified. To tackle these problems, we propose an end-to-end multi-task deep learning model for passenger demand prediction. First, we select similar regions for each target region based on their Point-of-Interest (PoI) information or historical demand and utilize Convolutional Neural Networks (CNN) to extract their spatial correlations. Second, we map external factors to future demand levels as part of the multi-task learning framework to further boost prediction accuracy. We conduct experiments on a large-scale real-world dataset collected from a city in China with a population of 1.5 million. The results demonstrate that our model significantly outperforms the state-of-the-art and a set of baseline methods.

Keywords: Demand Prediction · Muti-Task Learning · Spatial-Temporal Correlations · Convolutional Recurrent Neural Networks .

1 Introduction

Taxis are an integral mode of transportation in cities and serve a large number of passengers on a daily basis. However, traditional taxi services are slow in adopting Information and Communication technologies (ICT) to improve their efficiency and provide better services to commuters. In recent years, many online peer-to-peer ridesharing services such as Uber and Didi have successfully filled this void. These services allow customers to book a ride through their mobile apps. Drivers are then matched with customers based on their proximity. These services often employ dynamic pricing models and have significantly impacted the taxi markets in most countries. Despite the sophisticated ICT technologies

adopted, these ride-sharing solutions have still not cracked the code on the relationship between passenger demand and ride supply. On the one hand, drivers often have to drive a long way before they can find passengers due to low demand volumes in their locations [5]; on the other hand, passengers may experience long delays in obtaining a ride due to the high demand in their locations. This imbalance between the demand and supply incurs excessive delays and energy consumption, thus calling for an effective passenger demand prediction method for efficient scheduling of taxis and shared cars.

A variety of techniques have been proposed to address this problem in the literature. Traditional methods [6][7] utilize time series models such as Auto-Regressive Integrated Moving Average (ARIMA) and its variants to predict traffic. These methods only consider temporal correlation. However, recent studies [2][3][8] have revealed that a region's passenger demand is also related to other regions demand and thus utilizing the spatial relationship between regions could positively help predict future passenger demand. There are two ways of utilizing the spatial relationships in literature: (1) Treating the whole city as an image (a two-dimensional matrix) and applying CNN [2][12] or Convolutional Long-Short Term Memory (ConvLSTM) [13][1] directly to this image to capture relationships among all regions. Although this method can find all possible relationships, it may also introduce weak or negative correlations. As a result, this method may adversely impact the prediction outcomes. Also, processing the data in this manner for a large city requires several CNN layers, which consumes significant resources. (2) The second focuses on discovering local relationships [3]. This method treats the target region, and it's surrounding regions as an image. The foundation of this method is that "near things are more related than distant things". However, it neglects the fact that remote regions could also share strong similarities in passenger demand patterns if they have similar properties (for example they have similar PoIs such as schools or hospitals). Besides, both of these two approaches can only be applied if the city is partitioned by a grid-based method. Furthermore, the integration of ubiquitous technologies makes it possible to collect a vast amount of multi-modal data from urban spaces (e.g., historical crowd flow, weather, holidays), many of which may influence taxi demand and thus promote better prediction. However, previous works either don't take these external features into account [6][8] or directly map external features to future passenger demand [1][4][13], which can lead to large biases because the influence of external factors is not uniform to all regions.

In this work, we propose an end-to-end unified deep learning framework to predict passenger demand. Our model readily scales to large urban areas and also incorporates insights from urban data sources of different types. More specifically, the model takes historical passenger demand, historical crowd outflow data, PoI information, weather data, air quality data and time meta (time of day, day of week, and holidays) as inputs for predicting the passenger demand in future for all regions. We first select similar regions for each target region by their PoI information or historical demand, then utilize CNN and LSTM to extract their spatial-temporal relationships. Our method is more flexible than

the two approaches listed above which either consider all regions or co-located regions, as it can both filter weakly-related adjacent regions and find similar remote regions. Moreover, our method operates without knowledge of how the city is partitioned, i.e., either using road networks or relying on grids. Besides, to better utilize external features, we design an auxiliary task under the multi-task learning framework which predicts the demand level (e.g., high, medium or low) for each region in next time interval to further improve the passenger demand prediction task. Besides, our framework does not need hand-crafted features and is extensible to other datasets. The contributions of our work include the following:

- We propose a multi-task deep learning based framework for passenger demand prediction to solve the supply-demand imbalance problem in urban transportation systems. The framework takes features from multiple urban datasets into consideration and incorporates their joint influence on future passenger demands.
- We propose a similarity-based CNN model to capture the spatial similarity exclusively with similar regions. Our model emphasizes highly correlated regions, while simultaneously filtering out the influence of weakly related regions.
- We propose to predict and classify future passenger demand level as an auxiliary task under the multi-task learning framework to better utilize the power of external features and enhance the prediction accuracy of passenger demand value. None of the above aspects have been examined thoroughly in previous works.
- We conduct extensive experiments on a large-scale real-world dataset collected from a major city in China covering 1.5 million people, and demonstrate that our method outperforms a series of baselines and state-of-the-art methods.

2 Proposed Approach

2.1 Problem Formulation

In this section, we will first give some notations and definitions used to formalize the passenger demand prediction problem.

Notation 1: Region We utilize the road networks based partition [4] to divide the entire city into blocks as it is more flexible and can integrate semantic meanings into regions. The entire city is divided into N regions, represented as a set: $\{r_1, r_2, ..., r_i, ...r_N\}$.

Notation 2: External Features These represent the following information: weather data, air quality data, PoI information and time meta (e.g., time of day, day of week, holidays).

Definition 1: Passenger Demand The passenger demand of region $r_i (i \in [1, N])$ in a given period t is defined as the number of taxi requests originating in this region during this time period, which can be represented as $D_t(r_i)$.

Definition 2: Crowd Outflow [11] We use $P_T(r_i)$ to denote the set of people in region r_i at time T. The crowd outflow of region r_i during time interval t can be defined as $C_t(r_i) = P_T(r_i) \setminus P_{T+\Delta T}(r_i)$.

Passenger Demand Prediction Let $S_t(r_i)$ denotes all the historically observed data (passenger demand, crowd outflow) for region r_i in time period t, $E_{t+1}(r_i)$ denotes all external features in time interval $t+1$ (since the weather in time interval $t+1$ is unknown, we can use the predicted weather or the weather in time t), passenger demand prediction aims to to predict:

$$D_{t+1}(r_i) = \mathcal{F}(S_t(r_i), S_{t-1}(r_i), S_{t-2}(r_i), ..., S_{t-h}(r_i), E_{t+1}(r_i))$$

where $D_{t+1}(r_i)$ is the passenger demand for region r_i in time interval $t+1$, h is the historical window of time that is used for prediction. We define our prediction function $\mathcal{F}(\cdot)$ on all regions and previous time period up to $t-h$.

2.2 Multi-Task CRNN (MT-CRNN) framework

We design a multi-task deep learning framework (shown in Fig.1) that contains two tasks: (1) The main task involves predicting the precise passenger demand in the next time interval by capturing the spatial-temporal relationships within historical observed data from selected regions; (2) The auxiliary task is to predict and classify the level of passenger demand (e.g., whether it is high, low or medium in the next time interval) to get a better representation of external features for the main task.

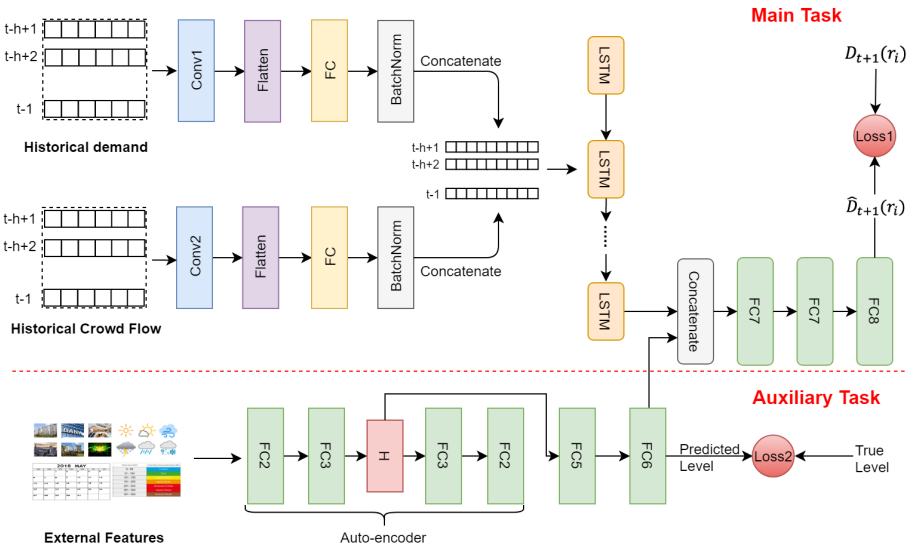

Fig. 1. Multi-Task Convolutional Recurrent Neural Networks (MT-CRNN)

Auxiliary Task: Predict Future Passenger Demand Level For passenger demand prediction and related tasks (such as crowd prediction, air quality prediction, rainfall prediction and so on), it is not apparent how external features can be used. External features from different domains may have different

attributes, i.e., they may vary from dynamic to static and from continuous to categorical. Misusing external features may adversely impact the accuracy of the final prediction. The state-of-the-art methods [1][4][13] usually map external features to the value of the passenger demand directly, which can result in significant errors and thus doesn't make the best use of external features. Our intuition is that external features have a closer relationship with a more granular measure of the passenger demand rather than the precise value. Mapping external features to passenger demand level could lead to a better representation of external features. Here, we use categorical values to denote levels. For example, level 0, 1 and 2 map to low, medium and high passenger demand respectively.

Table 1. Passenger Demand Level Generation

Level	Condition	Label
Extreme	$D_{t+1}(r_i) >= 3 * Ar_i$	3
High	$2 * Ar_i =< D_{t+1}(r_i) < 3 * Ar_i$	2
Medium	$1 * Ar_i =< D_{t+1}(r_i) < 2 * Ar_i$	1
Low	$D_{t+1}(r_i) < Ar_i$	0

Label Generation Predicting passenger demand level of the next time interval is a classification task. Given the external features as input, the classifier outputs the label of the corresponding passenger demand level. To train a classifier, we organize external features as a vector. We use $E_{t+1}(r_i)$ to represent the vector of region r_i in time interval $t+1$ and then label $E_{t+1}(r_i)$ by comparing $D_{t+1}(r_i)$ with the average passenger demand of region r_i in a day. In this paper, labels are generated according to Table 1, where A_{r_i} is the average passenger demand of region r_i in the corresponding day. Considering the high imbalance of passenger demand in large cities, region-specific average demands are more meaningful than a comprehensive average demand for the entire city. We emphasize that predicting demand level would not involve unavailable data because we only need to generate demand level when training the model. We do not need to generate demand level in the predicting phase. While some external features (such as weather) of the next time interval are also not obtainable in testing, we can use the predicted value or the value in the last period.

Classification As shown in Fig.1, the classification portion of the auxiliary task is composed of an auto-encoder and two fully-connected layers. $E_{t+1}(r_i)$ is fed into an auto-encoder at first to fuse features from different domains together while keeping most of the useful information. The encoding and decoding processes are implemented with two-layer fully-connected neural networks: $H_{t+1}(r_i) = encoder(E_{t+1}(r_i))$ and $\hat{E}_{t+1}(r_i) = decoder(H_{t+1}(r_i))$. Then the hidden representation $H_{t+1}(r_i)$ is fed into two fully connected layers for classification.

Main Task: Predicting Future Passenger Demand As introduced in Section 1, previous works are inapplicable to non-grid based city partition datasets. When applied to grid-based datasets, they either introduce excessive data from weakly correlated regions or miss out on exploiting correlations from spatially distant but similar regions. To overcome these problems, we propose to extract

spatial correlations only from regions that are similar to the target region. We propose two strategies to measure similarities between different regions.

Measuring by historical order sequence A direct way to measure the similarity between different regions is to calculate the correlations (e.g., Pearson Coefficient) with historical demand. Let $D_{0\sim t}(r_i)$ represent historical order sequence of region r_i from time 0 to t in the training data. Then the similarity of region r_i and r_j can be defined as:

$$Similarity_{r_i,r_j} = Pearson(D_{0\sim t}(r_i), D_{0\sim t}(r_j)) \tag{1}$$

Measuring by PoI similarity PoI information can also be used to measure the region similarity. Our motivation is that the existence of certain PoIs in a region can directly influence the passenger demand patterns in that region. For example, if a region has many shopping malls, then passenger demand of that region would significantly increase on weekends and holidays. PoI data can thus be used to characterize a region, analyze the region's passenger demand patterns and find regions that have similar characteristics. Consequently, regions with similar categories of PoI are likely to share similar patterns of passenger demand. Considering that different regions are of different size, we normalize the PoI information of each region with the area of that region. For region r_i and region r_j, similarity between r_i and r_j is :

$$Similarity_{r_i,r_j} = \| \frac{poi_{r_i}}{Area_{r_i}} - \frac{poi_{r_j}}{Area_{r_j}} \|_1 \tag{2}$$

where $Area_{r_i}$ and $Area_{r_j}$ represent the area of region r_i and region r_j respectively, $\| \cdot \|_1$ represents the L1 norm.

After obtaining the pairwise similarities between all regions, we select the m most similar regions for region r_i, represented as $r_{i\_s1}, r_{i\_s2}, ..., r_{i\_sm}$. We organize their passenger demand and crowd outflow data of the same time interval as a vector separately. The main task (predicting precise passenger demand) treats the passenger demand and crowd outflow data in the previous h time intervals of the target region and m similar regions as input to model the spatial and temporal correlations.

Convolutional Neural Networks As shown in Fig.1, historical passenger demand and crowd outflow are fed into two CNN networks separately to extract spatial relationships. In the following, we will omit the descriptions for crowd outflow data transformation as they are the same with processing passenger demand data. For each time interval in the previous h time intervals, we only use demand data during this period. Consider region r_i as the target region, we have the most similar m regions $r_{i\_s1}, r_{i\_s2}, ..., r_{i\_sm}$ of r_i. At time interval t, we treat and mix these $1 + m$ region's passenger demand as a $1 \times (2 * m)$ image respectively, represented as:

$$\begin{aligned} D_t(r_i, r_{i\_s1}, r_{i\_s2}, ..., r_{i\_sm}) = (&D_t(r_i), D_t(r_{i\_s1}), D_t(r_i), D_t(r_{i\_s2}), \\ &..., D_t(r_i), D_t(r_{i\_sm})) \end{aligned} \tag{3}$$

For time interval t in the previous h time intervals, the CNN takes $D_t(r_i, r_{i\_s1}, r_{i\_s2}, ..., r_{i\_s3})$ as input and feeds it into a convolutional layer. After the convolutional operation, a flatten layer is used to transfer the output of the convolutional

layer into a vector. Next we use a fully-connected layer to lower the dimension and get $R_d^t(r_i)$. Using the same approach, we can also get $R_c^t(r_i)$ for crowd outflow in time t of region r_i. Before feeding these two representations into the LSTM layer, we concatenate them together:

$$R_{dc}^t(r_i) = R_d^t(r_i) + R_c^t(r_i) \tag{4}$$

LSTM Layer The representations extracted from the CNN are fed into an LSTM layer to capture the temporal relationships between future passenger demand and previous h time interval's passenger demand and crowd outflow. Notice that we use previous h time intervals' passenger demand and crowd outflow as input to CNN and extract representations for each time interval separately, so we get h representations. We only save the output of the last LSTM cell for further processing:

$$Q_{t-h+1}^t(r_i) = lstm(R_{dc}^{t-h+1}(r_i), R_{dc}^{t-h+2}(r_i), ..., R_{dc}^t(r_i)) \tag{5}$$

where $lstm$ represents the transformation of all cells in LSTM layer, $Q_{t-h+1}^t(r_i)$ is the output of the last LSTM cell, it represents the captured spatial-temporal information of region r_i and corresponding top m most similar regions from time interval $t-h+1$ to t.

Combination and Prediction To fuse the information from spatial, temporal and external part together, we concatenate $Q_{t-h+1}^t(r_i)$ with $\hat{H}_{t+1}(r_i)$ together to form $U_{t+1}(r_i)$:

$$U_{t+1}(r_i) = Q_{t-h+1}^t(r_i) + \hat{H}_{t+1}(r_i) \tag{6}$$

Finally $U_{t+1}(r_i)$ is fed into three fully-connected layers to get the final predicted passenger demand $\hat{D}_{t+1}(r_i)$. Up to this point, the objective function of the proposed network is composed of three parts: (1) Constraint of auto-encoder in auxiliary task part $\mathcal{L}_1$; (2) Loss of passenger demand level prediction in auxiliary task part $\mathcal{L}_2$; (3) Loss of final passenger demand prediction in main task part $\mathcal{L}_3$:

$$\mathcal{L}_1 = MSE(\hat{E}_{t+1}(r_i) - E_{t+1}(r_i)) \tag{7}$$

$$\mathcal{L}_2 = Cross\_entropy(\hat{L}_{t+1}(r_i) - L_{t+1}(r_i)) \tag{8}$$

$$\mathcal{L}_3 = MSE(\hat{D}_{t+1}(r_i) - D_{t+1}(r_i)) \tag{9}$$

where MSE is the mean square error, $Cross\_entropy$ is the cross entropy loss, and $L_{t+1}(r_i)$ is the true label of passenger demand level for region r_i in $t+1$. Then the overall loss is:

$$\mathcal{L}(\theta) = \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3 \tag{10}$$

where θ represents all learnable parameters in the network. It is obtained via back-propagation and Adadelta optimizer.

3 Experiments

3.1 Dataset

We use real-world collected datasets to evaluate our method. There are five datasets collected from Dec 5th, 2016 to Feb 4th, 2017 in Shenyang, a big city in China [11]:

– **Passenger Demand Data**: This dataset contains taxi request data of Didi Chuxing. Each item contains the time and location (latitude and longitude) of a request. We pre-process this dataset to map requests to related regions and time intervals. In our experiment, we set the time interval to 1 hour.
– **Crowd Outflow Data**: This data is extracted from the cellular networks of the same city which covers more than 1.5×10^6 mobile users. They are also mapped to related regions and time intervals, with the time interval set to 1 hour.
– **Meteorological Data**: The meteorological dataset contains information about weather and air quality, including temperature, wind speed, visibility, weather, and air quality level. Temperature, wind speed, and visibility readings are continuous and updated every one hour. Weather and air quality level are categorical data.
– **PoI data**: We collected PoI data of 12 categories, including offices, entertainment facilities, hotels, shopping malls, residences (i.e., apartments), schools, banks, restaurants, government facilities, bus stations, tourist attractions, and hospitals. Each PoI item contains name and location (latitude and longitude). We pre-process this dataset to map PoI data to related regions.
– **Time Meta**: Time meta includes hour of day, day of week, and holiday information.

3.2 Experiment Settings

The model is implemented in TensorFlow 1.8. Due to the page limitations, we have excluded the discussion on parameter tuning. In our experiments, the length of the time interval used is 1 hour. We set the number of the most similar regions m to 3 and the historical time window h to 8, which means previous 8 hours passenger demand and crowd outflow of the most similar three regions are used to predict the passenger demand in next hour. We use 32 kernels with size 1×3 in CNN, and the stride is 1×1. The output dimension of CNN is re-scaled to 32 by FC layer. The hidden layer of auto-encoder is 24 dimensions. We set the learning rate to 0.02, batch size to 140, and use previous 80% of the data for training and the rest 20% for testing. To evaluate the model, we use Root Mean Square Error $RMSE = \sqrt{\frac{1}{\epsilon}\sum_i (\hat{D}_{t+1}(r_i) - D_{t+1}(r_i))^2}$ and Mean Absolute Error $MAE = \frac{1}{\epsilon}\sum_i \|\hat{D}_{t+1}(r_i) - D_{t+1}(r_i)\|$ of **all regions** to evaluate our model, where ϵ is the number of total time intervals in testing data.

3.3 Experimental Results

Overall Comparison. To validate our model, we compare it with the following methods.

– HA: The historical average model predicts future passenger demand by calculating the average value of previous passenger demand in the same related time interval in the same region.
– ARIMA: The Auto-Regressive Integrated Moving Average model is a widely used time series prediction model which is a generalization of Auto-Regressive Moving Average (ARMA) model.

– SARIMA: The Seasonal Auto-Regressive Integrated Moving Average model is a variance of the ARIMA model, which can capture the seasonality in a time series data.
– OLSR: The Ordinary Least Square Regression model is a kind of linear regression model, it can estimate the relationship between multiple variables.
– MLP: The Multiple Layer perceptron is a typical class of feed-forward neural network. It has multiple layers and non-linear activation function.
– LSTM: As introduced in section 4, LSTM is a variation of recurrent neural networks, which is prominent in sequence data processing.
– XGBoost [14]: XGBoost is a boosting tree-based machine learning method, which is used to achieve state-of-the-art results on many data mining challenges.
– DMVST-Net [2]: DMVST-Net is a state-of-the-art method for predict passenger demand. It is a deep learning based method which considers both spatial and temporal correlations.

Table 2. Overall Comparison with Different Methods

Index	Method	RMSE	MAE
1	HA	25.028	95.573
2	ARIMA	23.702	93.829
3	SARIMA	23.293	91.682
4	OSLR	22.003	87.348
5	MLP	21.889	85.265
6	LSTM	21.799	88.049
7	XGBoost [14]	20.497	79.489
8	DMVST-Net [2]	20.231	80.753
9	**MT-CRNN (PoI)**	19.602	76.469
10	**MT-CRNN (Order)**	19.467	74.438

HA only considers the historical demand as input, while all other aforementioned models employ all features to predict future passenger demand. MLP contains four fully connected layers, while LSTM only has one layer. As described in Section 1, DMVST-Net [2] can only be used when the city is partitioned to grids. In order to compare with DMVST-Net, we fed inputs of our model to DMVST-Net.

We show the experimental results in Table.2. From the table, we can observe that the performance of simple neural networks such as MLP and LSTM is not good. They don't show much improvement over traditional methods such as SARIMA and OSLR. In contrast, state-of-the-art methods XGBoost and DMVST-Net achieve 18.17% and 19.57% improvement in RMSE over HA, respectively. However, our model produces the lowest RMSE (19.602 when measuring similarity by PoI and 19.467 when measuring similarity by historical order sequence) among all the methods. Furthermore, our method achieves 3.80% (RMSE) and 7.82% (MAE) relative improvement over DMVST-Net.

Component Analysis. We also evaluated some variations of our MT-CRNN model to study the effect of different components and our auxiliary task setting, including:

– ST-D: This model only performs the main task of the MT-CRNN model and uses historical demand data as input. Similar region selection, CNN and LSTM layers are the same with MT-CRNN. The loss function is $\mathcal{L}_1$.
– ST-DC: Similar to ST-D, ST-DC further integrates crowd flow data as additional input.
– ST-DE: ST-DE is a single task model with the same design as MT-CRNN, but it doesn't take crowd flow data as input. The loss function is $\mathcal{L}_1 + \mathcal{L}_3$.
– ST-DCE: In this model, the final loss function is $\mathcal{L}_1 + \mathcal{L}_3$, which transforms MT-CRNN to a single task model.
– MT-DE: MT-DE is a multi-task model with the same design as MT-CRNN, but it doesn't utilize crowd outflow data.
– MT-FC: In this model, the final loss function is $\mathcal{L}_2 + \mathcal{L}_3$, which means the decoder part of the auto-encoder is not trained. Thus, the auto-encoder is transformed into a two-layer fully-connected neural network.
– MT-GA: Instead of labeling the passenger demand level by the target region's average demand, this model labels the passenger demand with the entire region's average demand.

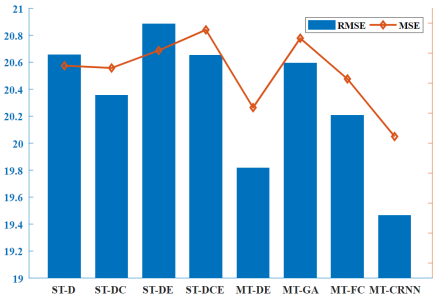

Fig. 2. Component Analysis

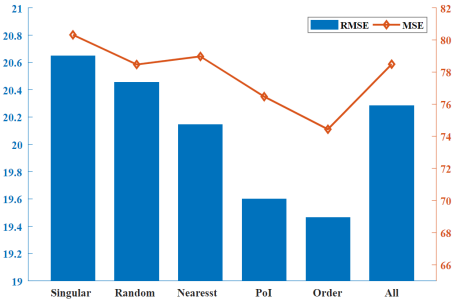

Fig. 3. Prediction with different correlated regions

From Fig.2, we can observe that MT-CRNN outperforms ST-DCE and MT-DE outperforms ST-DE, which justify the importance of our multi-task setting. Secondly, ST-DC outperforms ST-D, ST-DCE outperforms ST-DE and MT-CRNN outperforms MT-DE, which shows that prediction accuracy is better when crowd outflow data is included in addition to historical passenger demand data. Besides, ST-DCE performs worse than ST-DC and ST-DE performs worse than ST-D, which demonstrates that the improper use of external features adversely impacts prediction accuracy. Thirdly, the results for MT-FC shows that an auto-encoder is better than fully-connected neural networks in extracting hidden features from external data. Finally, by comparing MT-GA with MT-CRNN, we can show that considering region-specific average demand is better than average demand over the entire city.

Spatial Correlation Analysis. We also evaluated the performance of our model with data from different regions to analyze spatial correlations. As shown in Fig.3, we included the following:

– Singular: Only consider the target region's historical data;
– Random: Randomly select correlated regions for the target region. We randomly select regions five times and present the average prediction results.
– Nearest: Similar with DMVST-Net, it only considers spatially nearby regions to capture their spatial correlations;
– PoI: Select correlated regions by PoI similarity;
– Order: Select correlated regions by historical demand series similarity;
– All: Similar with DeepST [12], it captures the spatial correlations within the whole city.

We can observe that predicting only with selected similar region's data is better than all other strategies, which shows the advantages of our similarity-based CNN in capturing spatial correlations. Moreover, all strategies are better than One, which demonstrate the importance of spatial correlations in predicting passenger demand.

4 Related Works

One traditional method for passenger demand prediction is to consider passenger demand as time series data and applying time series models. Luis Moreira-Matias et al. [6] combined three time-series forecasting techniques (Time-Varying Poisson Model, Weighted Time-Varying Poisson Model, ARIMA model) to arrive at a prediction. Xiaolong Li et al. [7] proposed an improved ARIMA model to forecast the spatial-temporal variation of passengers in hotspots. These early works rely on GPS trajectories data from a subset of the entire taxis, which may not necessarily reveal the actual passenger demand. In recent years, some researchers have applied deep learning methods in smart transportation systems [10]. Rose Yu et al. [9] proposed to use Long-Short Term Memory (LSTM) network to capture the temporal relationship in historical observations and used auto-encoder to process static features. However, they didnt consider spatial correlations. Wang et al. [5] presented a neural network framework based on fully-connected layers and residual network to predict the gap between passenger demand and supply. Their approach cannot accurately capture the sequential relationship. Another way to capture the spatial correlation is treating the city as an image (a two-dimensional matrix) and applying CNN to it. Junbo Zhang et al. [3] propose a spatial-temporal model to predict citywide crowd flow. They represent city-wide crowd flow as a multi-dimensional image and use CNN and residual network to extract spatial relationships. Huaxiu Yao et al. [2] further designed "local CNN" to extract spatial relationship within surrounding regions and construct a weighted graph to represent similarity among regions. A positive aspect is that all these deep learning based methods take external features (weather, holiday, time meta) into consideration. However, they transform external features using fully-connected layers or auto-encoder, which are incapable of fully realising their potential.

5 Conclusions

In this paper, we proposed a Multi-Task Convolutional Recurrent Neural Network (MT-CRNN) framework to forecast the passenger demand with multiple

features from different domains. We captured the spatial-temporal correlations of historical passenger demand by the convolutional recurrent neural network based on the historical demand of selected similar regions. To better utilize external features, we designed an auxiliary task for predicting passenger demand level under the guideline of multi-task learning. Experimental results show that our model significantly outperforms a series of baselines and gains 3.8% improvement (RMSE) over state-of-the-art methods and that the auxiliary task can improve the final passenger demand prediction accuracy.

References

1. Ke, Jintao, et al. "Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach." Transportation Research Part C: Emerging Technologies 85 (2017): 591-608.
2. Yao, Huaxiu, et al. "Deep multi-view spatial-temporal network for taxi demand prediction." AAAI. 2018.
3. Zhang, Junbo, et al. "Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction." AAAI. 2017.
4. Deng, Dingxiong, et al. "Latent space model for road networks to predict time-varying traffic." Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2016.
5. Wang, Dong, et al. "DeepSD: supply-demand prediction for online car-hailing services using deep neural networks." 2017 IEEE 33rd International Conference on Data Engineering (ICDE). IEEE, 2017.
6. Moreira-Matias, Luis, et al. "Predicting taxipassenger demand using streaming data." IEEE Transactions on Intelligent Transportation Systems 14.3 (2013): 1393-1402.
7. Li, Xiaolong, et al. "Prediction of urban human mobility using large-scale taxi traces and its applications." Frontiers of Computer Science 6.1 (2012): 111-121.
8. Li, Yunxuan, et al. "Taxi booking mobile app order demand prediction based on short-term traffic forecasting." Transportation Research Record: Journal of the Transportation Research Board 2634 (2017): 57-68.
9. Yu, Rose, et al. "Deep learning: A generic approach for extreme condition traffic forecasting." Proceedings of the 2017 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2017.
10. Zheng, Yu, et al. "Urban computing: concepts, methodologies, and applications." ACM Transactions on Intelligent Systems and Technology (TIST) 5.3 (2014): 38.
11. Chu, Jing, et al. "Passenger Demand Prediction with Cellular Footprints." 2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON). IEEE, 2018.
12. Zhang, Junbo, et al. "DNN-based prediction model for spatio-temporal data." Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM, 2016.
13. Xingjian, S. H. I., et al. "Convolutional LSTM network: A machine learning approach for precipitation nowcasting." Advances in neural information processing systems. 2015.
14. Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016.