Forecasting the crowd: An effective and efficient neural network for citywide crowd information prediction at a fine spatio-temporal scale

Xucai Zhang[a], Yeran Sun[b], Fangli Guan[a, c], Kai Chen[d], Frank Witlox[a, e], Haosheng Huang[a*]

[a] Department of Geography, Ghent University, Ghent, 9000, Belgium
[b] Department of Geography, Faculty of Science and Engineering, Swansea University, Swansea, SA2 8PP, United Kingdom
[c] The State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, 430000, China.
[d] Guangdong Key Laboratory for Urbanization and Geo-simulation, School of Geography and Planning, Sun Yat-sen University, Guangzhou, 510275, China
[e] Department of Geography, University of Tartu, Tartu, 51014, Estonia

Abstract: Modelling and forecasting citywide crowd information (e.g., crowd volume of a region, the inflow of crowds into a region, outflow of crowds from a region) at a fine spatio-temporal scale is crucial for urban and transport planning, city management, public safety, and traffic management. However, this is a challenging task due to its complex spatial and temporal dependences. This paper proposes a novel and efficient model to reduce the training time cost while maintaining predictive accuracy in forecasting citywide crowd information at a fine spatio-temporal scale. Our model integrates Gated Recurrent Unit (GRU), convolutional neural network (CNN), and k-nearest neighbors (k-NN) to jointly capture the spatial and temporal dependences between two regions in a city. The evaluation with two different datasets in two different cities shows that compared to the state-of-the-art baselines, our model has better predictive accuracy (reducing the mean absolute errors MAEs by 20.99% on average) and a lower training time cost (reducing the time cost to only 26.16% on average of that of the baselines). Our model also has better abilities in making accurate predictions with low time cost under the influences of large-scale special events (when massive crowds of people are gathering in a short time) and for regions with high and irregular crowd changes. In summary, our model is an effective, efficient, and reliable method for forecasting citywide crowd information at a fine spatio-temporal scale, and has a high potential for many applications, such as city management, public safety, and transportation.

Keywords: Crowd Information, Convolutional Neural Network; k-Nearest Neighbor; Gated Recurrent Unit; Training Time Cost

1. Introduction

Modelling and predicting fine-grained spatial-temporal crowd information in citywide environments, e.g., crowd volume (i.e., the number of people presented) in each region of a city, crowd flows into or out of each region, is of great importance to urban planning, public safety, transport planning, and traffic management (Ahas et al. 2015, Demissie, Correia and Bento 2015, Liu et al. 2020). For instance, modelling the fine-grained crowd distribution in a city provides a scientific basis for city management to reasonably allocate city resources dynamically and optimally. Knowing and predicting (near) real-time crowd mobility in a city also helps to prevent

* Correpsonding author: Haosheng Huang (haosheng.huang@ugent.be)

catastrophic accidents caused by massive crowds of people gathering in a short time. For example, many tourists and citizens gathered into Bund in Shanghai to celebrate the New Year of 2015, causing many people to fall and overlap and resulting in a stampede that killed 36 and injured 49. Additionally, such information also helps traffic managers and transportation operators to optimize their mobility services and improve the efficiency of the transportation system. Accordingly, much research has been conducted in recent years to develop methods for modeling and predicting such fine-grained spatial-temporal crowd information in cities. Various data have been employed as a proxy to represent such crowd information, including mobile phone network data (such as call detail records CDRs and signaling data) (Jarv, Ahas and Witlox 2014), bike-sharing data (Zhang et al. 2016, Zhang et al. 2018), taxi GPS trajectory data, location-based social media data, and location-based services usage/log data.

Forecasting crowd information at a fine spatial-temporal resolution in a citywide environment (e.g., the total volume of crowds in each 1km*1km region of a city in an hourly interval), however, has always been a challenging task due to its complex spatial and temporal dependences. In terms of spatial dependences, due to human mobility between different regions in a city, the crowd volume in a region is affected by the inflow and outflow of nearby regions and vice versa. With the development of transportation infrastructure, such as subways and other high-speed transportation (e.g., light rail), that more efficiently connect different regions within a city, such spatial dependences exist not only between nearby regions, but also between distant regions. In other words, such spatial dependences often present nonlinear characteristics and are influenced by many factors. In terms of *temporal dependences*, the distribution of crowds in a city changes dynamically over time and can be generally characterized by periodicity (e.g., the crowd distribution at 9am might be similar on consecutive workdays, repeating every 24 hours) and recent trends (e.g., the citywide crowd distribution at 9am will affect that of 10am, or even longer). Meanwhile, time of the day, weekdays/weekends, and special events might also significantly affect the crowd distribution in a city. Therefore, to effectively forecast fine-grained crowd information, it is essential to consider both the spatial and temporal dependences.

Existing methods for predicting the volume of crowds and traffic flows can be mainly divided into two groups: (i) statistical and machine learning, and (ii) deep learning algorithms (For an extensive overview, please refer to Section 2). Examples of the first group include the applications of k-nearest neighbors (k-NN), autoregressive integrated moving average (ARIMA) and its extensions, random forest (RF), support vector regression (SVR) (Smith, Williams and Oswald 2002, Hamner 2010, Xia et al. 2016). Although these statistical and conventional machine learning algorithms are easier to train, they are often limited in their predictive accuracy. Recent years have seen an increasing interest of developing deep learning-based methods. Such studies typically combine (graph) convolutional neural network (CNN, GCNN), recurrent neural network (e.g., long short-term memory networks LSTM and gated recurrent units GRU), and other neural networks to capture the spatio-temporal dependences between the data (Zhang et al. 2018, Wu et al. 2019, Zheng et al. 2020, Xu, Kang and Cao 2022). Although more complex models (e.g., with more hidden layers or sophisticated structures) can potentially achieve better predictive accuracy, much more time must be spent on the training phase, and thus more computational resources are required and more energy is consumed. How to reduce training time cost while maintaining excellent predictive accuracy is still an open research challenge.

To tackle this open research challenge, this study proposes a novel and *efficient* neural network model for forecasting crowd information in citywide environments, with the aims to reduce the training time cost while maintaining a better predictive accuracy than the baseline models. The proposed model combines recurrent neural networks (i.e., GRU) and convolutional neural network (CNN) to jointly capture the complex spatio-temporal dependences of crowd information. More importantly, a k-nearest neighbors (k-NN) module, which is shown to be an effective and efficient conventional predictive method in the literature, is added to the model to further capture more 'neighborhoods' features and accelerate the convergence of the loss function, thus reducing the training time cost of the proposed model and improving its predictive accuracy. In short, we term the proposed neural network model as ST-RCNet-knn. Our contributions are three-fold:

1) Our proposed ST-RCNet-knn model integrates two DL approaches GRU and CNN and a conventional ML method k-NN to jointly model spatial and temporal dependences between nearby grid cells in a city. This combination allows to significantly reduce the training time cost, while still ensuring an excellent predictive accuracy.
2) We comprehensively evaluate our ST-RCNet-knn model with state-of-the-art predictive models using two different datasets in two different cities. The evaluation results show that our ST-RCNet-knn model can better capture both temporal dependences (via the GRU part) and spatial dependences (via the CNN and k-NN part), despite a very simple and shallow network structure. Compared to the state-of-the-art baselines, our model reduces the mean absolute errors (MAEs) by 20.99% on average (minimum: 4.00%; maximum: 63.56%), while its training time cost is only 26.16% on average (minimum: 1.07%; maximum: 57.98%) of that of the baselines. In short, our model significantly outperforms the state-of-the-art models in terms of both the predictive accuracies and the training time costs.
3) The results also illustrate that compared to the state-of-the-art baselines, our model is able to make more accurate prediction with low time cost under the influences of large-scale special events (when massive crowds gather in short time), as well as more robust to make prediction for regions with high and irregular crowd changes.

2. Related Work

With regard to prediction of the volume of crowds and traffic flows, two groups of methods can be identified in the literature: (i) statistical and machine learning-based method, and (ii) deep learning algorithms. This section summaries the related works from these two perspectives.

2.1 Traditional method for crowd flow prediction

For statistical and conventional machine learning-based algorithms, many researchers have applied k-nearest neighbors (k-NN) to predict flow volume for a short time (Chen et al. 2018, Xia et al. 2016). Other predictive methods include Kalman filtering model (Okutani and Stephanedes 1984), support vector regression (Wu, Ho and Lee 2004, Semanjski et al. 2017), Bayesian model (Sun, Zhang and Yu 2006), and autoregressive integrated moving average (ARIMA) (Smith et al. 2002). With the improvement of prediction techniques, many extended models were proposed to enhance the predictive accuracy, including spatial-temporal weighted k-nearest neighbors (Xia et al. 2016), Kohonen ARIMA (Van Der Voort, Dougherty and Watson 1996), seasonal ARIMA (Williams and Hoel 2003), seasonal SVM (Hong 2011), and online SVM (Castro-Neto et al. 2009). Additionally, random forest (RF) has been also employed in traffic flow prediction and achieved a good performance by consider more contextual information (Hamner 2010). Although the statistical and machine learning algorithms have a lower cost in training time, they are often limited in capturing complex and dynamic spatio-temporal dependences to obtain better predictive accuracy.

2.2 Deep learning method for crowd flow prediction

In recent years, DL has grown as one of the best techniques in application fields such as computer vision (Vinyals et al. 2015) and natural language processing (Gu et al. 2018). As an excellent tool being capable of modelling complex dependences between data, DL is very promising for addressing the problems of predicting crowd information (Xie et al. 2020). For example, there were researchers who focused on applying the convolutional neural network (CNN) (Ma et al. 2017) to capture the spatial characteristics for forecasting traffic flow. Additionally, other researchers proposed an online gated recurrent unit (GRU) model to consider periodicity characteristics for improving prediction accuracy (Fan et al. 2018). Likewise, Liu, Liu and Jia (2019) combined fully connected network and long short-term memory (LSTM) to predict metro passenger flow, wherein fully connected network is used to extract the external features, and LSTM is applied to portray the temporal dependency. The studies mentioned above only considered either spatial or temporal dependences and failed to jointly consider both aspects.

To address this issue, Zhang et al. (2018) proposed a DL-based model (called ST-ResNet), which combines CNN and residual convolutional network, to capture spatiotemporal dependences based on three units of temporal closeness, period, and trend of crowd flow. To further portray the temporal features, many works integrate CNN and RNN to exploit the capability of the latter to address temporal patterns (Luca et al. 2021). For instance, Yao et al. (2019) proposed a spatial-temporal dynamic network which applied CNN module to capture the spatial features and LSTM to portray temporal features. Extended from ST-ResNet, Xu et al. (2022) developed a high-resolution spatiotemporal transformer network, which employed multi-head attention mechanism's transformer to capture the spatiotemporal features in closeness, period and trend patterns . Apart from portraying the spatiotemporal dependences, other features are also added to enhance the prediction accuracy. For example, Geng et al. (2019) proposed a multi-graph convolutional network, in which the relationship of neighborhoods, their connective and function were represented into three graphs. A multi-view residual attention network additionally applied node2vec to encode the transition probability and transition distance between urban functional areas, and the crowd flows patterns of the functional areas, which effectively portrays the mobility pattern to enhance prediction accuracy (Yuan et al. 2020). Additionally, Zhang et al. (2020) used the Euclidean distance between two regions and the Pearson correlation coefficient of historical data as distance to construct two k-NN graphs that are encoded with graph convolutional neural network (GCNN).

For the irregular grids and topological construction, Zhao et al. (2020) proposed a temporal graph convolutional network, which combines GCNN and GRU, to describe the spatio-temporal characteristics of traffic flow. Diffusion convolutional recurrent neural network was also developed to model diffusion process of graph signals, which is proved to be effective in spatio-temporal modelling (Li et al. 2017b). Inspired by that, Wu et al. (2019) designed a gating mechanism with diffusion convolutional layer, which is helpful for aggregating and transforming the neighborhood information, along with GCNN to predict the traffic flow. Additionally, Guo et al. (2019) developed an attention-based GCNN to capture the spatio-temporal attributes of traffic flow. Similarly, Zheng et al. (2020) introduced the attention mechanism into GMAN, that includes spatio-temporal embedding layer, ST-attention blocks and transformer attention layer, to forecast the traffic flow at intersections. They further used node2vec to capture the topological relationships between intersections. Meanwhile, to capture the time series characteristics more completely, researchers introduced different ways to encode the temporal dependences of crowd flow integrating GCNN and transformer model (Xu et al. 2020, Cai et al. 2020).

While more complex models (e.g., with more hidden layers, sophisticated structures, or supplement information) can potentially achieve better predictive accuracy, much more time must be spent on the training phase, and thus more computational resources are required and more energy is consumed, which is unfriendly to the limited computational resources or users who just expect a good accuracy under limited time cost. How to reduce training time cost while maintaining excellent predictive accuracy is still an open research challenge. Thereby, this study addresses this issue by proposing ST-RCNet-knn, which integrates DL approaches (specifically GRU and CNN) and conventional ML (specifically k-NN).

3. Methodology

3.1 Problem Definition

This study aims to predict crowd information at a given time in each region of an entire city, using historical observations. In our methodology, crowd information is seen as a general concept. It can be crowd volume of a region (i.e., the number of people presented in the region), crowd density of a region, inflow crowds into a region (i.e., the total traffic of crowds entering the region), and outflow crowds from a region (i.e., the total traffic of crowds leaving a region). Without loss of generality, in the following we mainly use crowd volume as an example of crowd information.

Various ways can be used to define a region, in terms of different granularities and semantic meanings. Similar to Zhang et al. (2018), this study partitions a city into a $I \times J$ grid map based on latitude and longitude, where each grid cell represents a region, as shown in Figure 1 (left).

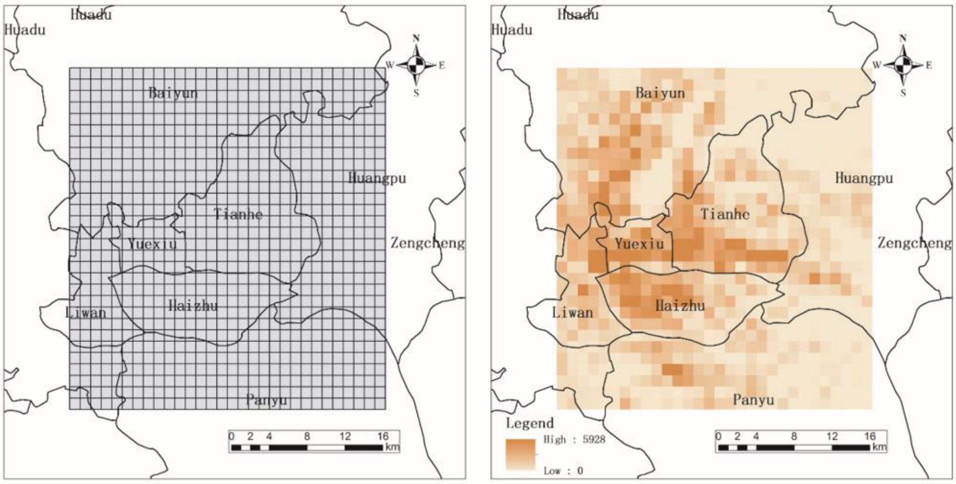

Figure 1. Regions of Guangzhou. Left: grid map with 900 cells, each of which has a size of 1km x 1km; Right: an example of a cell-level crowd distribution

At the t[th] time interval, the crowd volume in all $I \times J$ regions/cells can be denoted as a matrix $X_t \in \mathbb{R}^{I \times J}$. An example of such a matrix is shown in Figure 1 (right). It shows the crowd distribution in Guangzhou (China) during the time from 18:00 to 19:00 on a weekday. Grid cells with high crowd volumes (those with dark orange colors) are mainly around the urban villages in the neighboring area of Liwan, Haizhu, and Yuexiu District (old city center) and the CBD located in Tianhe District (new city center). Owing to relatively low housing expenses and proximity to workplace, numerous younger people live in these areas, making these areas socially active.

Therefore, the problem of fine-grained crowd information prediction can be defined as: given the historical observations $X_0, \cdots, X_{t-2}, X_{t-1}$, predict X_t.

$$\widehat{X_t} = f(X_0, \cdots, X_{t-2}, X_{t-1}) \tag{1}$$

3.2 Overview of the ST-RCNet-knn model

Figure 2 presents the framework of the proposed ST-RCNet-knn model, which consists of four main components, namely 'weekly pattern', 'daily pattern', 'recent hourly trend', and 'nearest neighbors'. As illustrated in the top part of Figure 2, we first consider the crowd distribution throughout a city at each time interval as a gray-scale image-like matrix. We then extract different subsets of historical time series, denoting the recent hourly trend (time series data of the last several hours immediately before t: $S^H = [X_{t-a}, \cdots, X_{t-2}, X_{t-1}]$), daily pattern (time series of data of the specific hour of the last several days: $S^D = [X_{t-b*24}, \cdots, X_{t-2*24}, X_{t-24}]$), and weekly pattern (time series of data of the specific hour and the specific day of the last several weeks: $S^W = [X_{t-c*24*7}, \cdots, X_{t-2*24*7}, X_{t-24*7}]$). a, b, c are hyperparameters describing the input time series lengths considered in our prediction model.

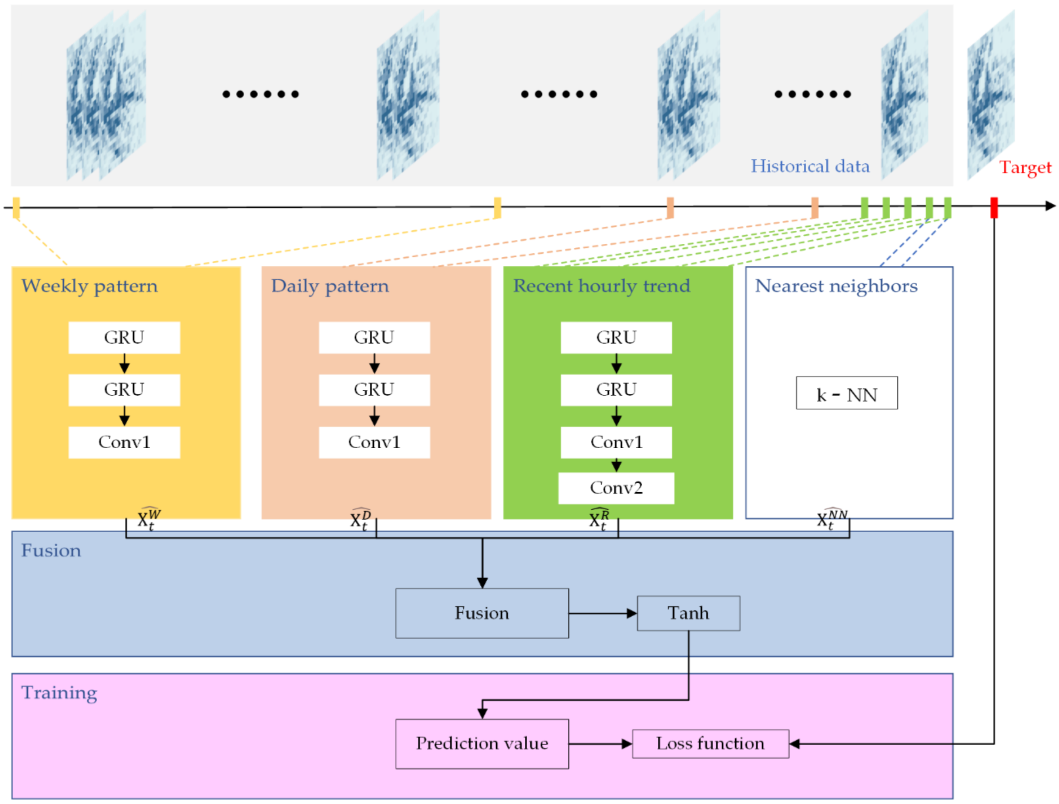

Figure 2. ST-RCNet-knn architecture. GRU: gated recurrent unit; Conv: Convolution; k-NN: k-nearest neighbors.

These three subsets are then fed into the first three components respectively, with the aim to model the spatial-temporal characteristics of crowd information at the hourly, daily, and weekly levels. The 'weekly pattern' and 'daily pattern' components share the same network structure with two gated recurrent unit (GRU) layers followed by a convolutional layer (Conv). Such structure captures the temporal dependences between historical observations of individual regions (via GRUs), and the spatial dependences between nearby regions (via Conv). For the 'recent hourly trend', the importance of spatial dependences increases (e.g., the inflow/outflow of other regions, nearby or distant, will more *directly* influence the crowd volume of a region). Therefore, apart from two GRU layers in the 'recent hourly trend', two convolutional layers are integrated to capture the hourly spatial-temporal characteristics, since two conventional layers can capture the spatial dependences over a wider range of areas.

The efficiency of k-NN model in traffic flow prediction have been empirically proved (Chen et al. 2018, Smith et al. 2002). Essentially, it is a nonparametric regression model without accounting for specific training in advance. To reduce the time cost in the training phase and further capture more 'neighborhoods' features, we introduce a 'nearest neighbors' component based on the k-NN model, which can accelerate the convergence of the loss function to some extent. We feed the historical time series data of the past two hours and the latitude/longitude location information of a grid cell into the k-NN model to obtain preliminary results.

The outputs of the four components are then fused into a single matrix based on parameters, which assign different weights to the results of these components in different grid cells. An activation function of Tanh (hyperbolic tangent) is then applied to the fused matrix to output the final forecasting values $\widehat{X_t}$.

3.3 Gated Recurrent Unit

The crowd volume in each region of a city changes dynamically over time, and is generally characterized by periodicity (e.g., the crowd distribution at 3 pm might be similar on consecutive workdays, repeating every 24 hours) and recent trends (e.g., the citywide crowd distribution at 3pm will affect that of 4pm, or even longer). Considering such temporal dependence is vital for forecasting fine-scale crowd information in a city. Currently, LSTM and GRU are two state-of-the-art neural network models for processing sequence data, and they avoid the gradient vanishing and explosion problems of the traditional recurrent neural network (RNN). Both LSTM and GRU use a gated mechanism to decide how much information from the previous stages should be passed to the output. Compared to LSTM, GRU has a relatively simpler structure, fewer parameters and is easier to train (Cho et al. 2014). Therefore, this study employs GRU to model the temporal dependences from the citywide crowd distribution data.

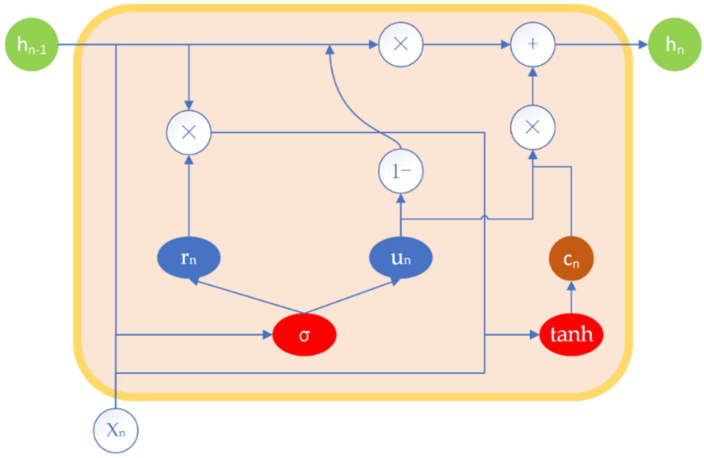

Figure 3. The workflow of a GRU unit

A single GRU unit (Figure 3) takes the current input data X_n and state h_{n-1} which holds the useful information of the previous $n-1$ GRU units, and outputs the new state h_n. It has two gates: an update gate that determines how much of the previous information needs to be passed along to the future (i.e., h_n); a reset gate deciding how much of the previous information to forget. The calculation formula of each GRU unit is shown below:

$$r_n = \sigma(W_r \cdot [h_{n-1}, X_n] + b_r) \tag{2}$$

$$u_n = \sigma(W_u \cdot [h_{n-1}, X_n] + b_u) \tag{3}$$

$$c_n = tanh(W_c \cdot [r_n \circ h_{n-1}, X_n] + b_c) \tag{4}$$

$$h_n = (1 - u_n) \circ h_{n-1} + z_n \circ c_n \tag{5}$$

$$\hat{X}_{n+1} = \sigma(W_o * h_n) \tag{6}$$

Where X_n denotes the current input data (e.g., the citywide crowd distribution at time n), W represents the learnable parameters, σ and $tanh$ refer to sigmoid and hyperbolic tangent activation functions which add nonlinearities to the model, operator $\circ$ denotes Hadamard product (i.e., element-wise multiplication). r_n and u_n denote the reset gate and update gate respectively, which control how much previous information h_{n-1} (gained from previous time steps) and current information gained from X_n will be passed along to the new state h_n, which is then used to forecast the $\hat{X}_{n+1}$. c_n is a candidate state.

As shown in Figure 2, we employ two GRU layers to extract the temporal dependences at the hourly, daily, and weekly levels respectively. Take 'recent hourly trend' as an example, the time series data of the last several hours immediately before t (i.e., $S^H = [X_{t-a}, \cdots, X_{t-2}, X_{t-1}]$) are taken one-by-one as input for the first GRU layer. In total, there are a GRU units in the first layer, with the first data input as X_{t-a}, the second as X_{t-a+1}, …, and until X_{t-1}. Each of these a GRU units outputs a state h. Following the common practices in GRU stacking, the GRU units in the 2nd layer take each output state h of the first layer as input step-by-step, and finally output the final state h. Again, there are a GRU units in the 2nd layer. Using equation (6), the final output state h is then used to forecast $\widehat{X_t^{H}}'$ (in terms of recent hourly trend). Similarly, we can obtain $\widehat{X_t^{D}}'$ (in terms of daily pattern), and $\widehat{X_t^{W}}'$ (in terms of weekly patterns). Note that $\widehat{X_t^{H}}'$, $\widehat{X_t^{D}}'$ and $\widehat{X_t^{W}}'$ are all $I \times J$ matrices. They will be then taken as inputs and fed into the convolutional layer(s) in the 'recent hourly trend', 'daily pattern', and 'weekly pattern' components, respectively.

3.4 Convolutional Neutral Network

Intuitively, due to human mobility between different regions in a city, the crowd volume in a region interacts with each other in nearby areas. To model such spatial dependences, we employ convolutional neural network (CNN) (Lecun et al. 1998), which has been shown to be an effective method to hierarchically capture the spatial structural information (Zhang et al. 2018). Different from the classical CNN, we only include convolutional layers, each of which consists of a convolution operation and activation function (Figure 4). The output of a single convolutional layer can be described as:

$$X_{l+1} = f(W_l * X_l + b_l) \tag{7}$$

where * represents the convolution operation; W_l are learnable parameters; l denotes the l-th layer in whole CNN model; X_l denotes the input of the layer; X_{l+1} refers to the output of the convolutional layer, and it also can be the input of l+1 -th layer; f is the activation function.

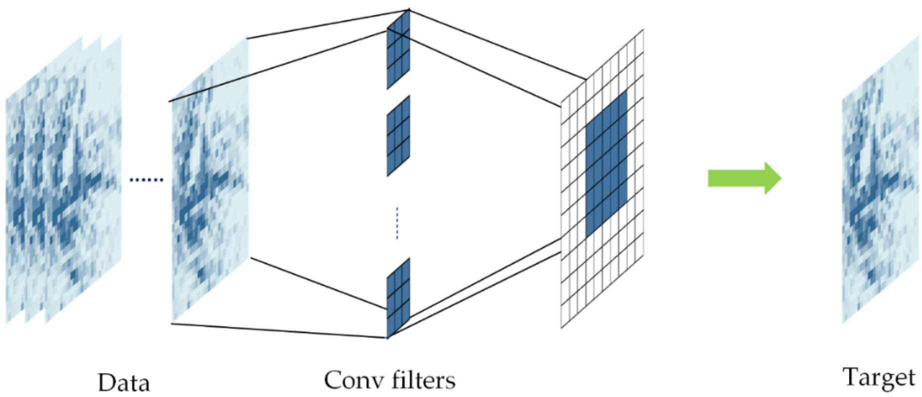

Figure 4. Simplistic graphic concept of CNN.

In this study, we use filters (i.e., kernels) with size of 3×3, which means that a node in a higher-layer feature map depends on nine nodes of its previous layer (i.e., a lower-level feature map). This means that one convolution layer is able to capture spatial dependences across immediately adjacent regions, while several convolution layers together can further capture spatial dependences over distant regions (Zhang et al. 2018). To ensure that the input and output have the same size (i.e., $I \times J$), we employ a same padding approach, by padding each area outside the border with a zero and setting stride = 1.

For the 'weekly pattern' and 'daily pattern' components in Figure 2, a single convolutional layer with 1 filter is added after the GRU part. In other words, the outputs of the GRU parts (i.e., $\widehat{\mathrm{X}_t^{W}}'$ and $\widehat{\mathrm{X}_t^{D}}'$, respectively) are fed into the convolutional layer. Using equation (7) and setting $X_l = \widehat{\mathrm{X}_t^{W}}'$ (or $X_l = \widehat{\mathrm{X}_t^{D}}'$ for daily patterns), they are then used to forecast $\widehat{\mathrm{X}_t^{W}}$ (in terms of weekly pattern) and $\widehat{\mathrm{X}_t^{D}}$ (in terms of daily pattern). The combination of the two GRUs and the convolutional layer helps to model both temporal and spatial dependences of the time series data.

For the 'recent hourly trend', considering the importance of spatial dependences increases, two convolutional layers, the first with 30 filters and the 2nd with 1 filter, are added after the GRU parts, with the aims to capture the spatial dependence over a wider range of areas. Using equation (7), the output of the GRU part (i.e., $\widehat{\mathrm{X}_t^{R}}'$) is fed into the first convolutional layer, whose outputs are then fed into the 2nd convolutional layer to forecast $\widehat{\mathrm{X}_t^{R}}$ (in terms of recent hourly trend).

3.5 K-Nearest Neighbors (k-NN)

K-Nearest Neighbors (k-NN) is one of the most popular classic machine learning models. For a new point whose value is to be predicted, k-NN first calculates the distance between the new point and each training point, e.g., using Euclidian or Manhattan distances (for continuous features) or Hamming distance (for categorical features). It then identifies the k nearest neighbors (i.e., the training points) of this new point. For classification tasks, the new point will be assigned the most common class label among its k nearest neighbors. For regression tasks, the value of the new point is the (weighted) average of the values of its k nearest neighbors.

In this study, we introduce the k-NN model to quickly capture more 'neighborhoods' features, with the aims to accelerate the convergence of the loss function and thus to reduce the time cost in the training phase. The feature space of each data point (i.e., each region in Figure 1) is the combination of its latitude/longitude and the past 2-hour crowd volumes:

$$S_i^{NN} = (lat, lon, x_{t-2}, x_{t-1}) \tag{8}$$

Where lat and lon are the latitude and longitude of the center of the i-th region; x_{t-2} and x_{t-1} are the crowd volumes of the last two hours at this region.

The reason of selecting such a feature space is to balance the importance between location information and crowd volumes. After normalizing (using Min-Max normalization) each feature dimension of all data points, Euclidian distance is then used to compute the similarity/distance between each data point. Consequently, the similarity value not only considers the location distance but also the similarity of the varying trend of crowd volumes. For each region whose value (i.e., its crowd volume at time t) is to be predicted, we then select the top-24 nearest regions based on the similarity value. The crowd volume of the region at time t is then forecasted as the average volume of these nearest regions at time t. This step is repeated for each region in the city. By aggregating these predicted values as a matrix, we can then obtain $\widehat{\mathrm{X}_t^{NN}}$, i.e., the predicted outcome of the 'nearest neighbor' component.

3.6 Fusion

The outputs of the four components 'weekly pattern', 'daily pattern', 'recent hourly trend', and 'nearest neighbors', i.e., $\widehat{\mathrm{X}_t^{W}}$, $\widehat{\mathrm{X}_t^{D}}$, $\widehat{\mathrm{X}_t^{R}}$, and $\widehat{\mathrm{X}_t^{NN}}$, respectively, are then fused into a single matrix based on parameters, which assign different weights to the results of different components in different regions. An activation function of Tanh (hyperbolic tangent) is then applied to the fused matrix to output the final forecasting values $\widehat{X_t}$.

$$\widehat{X_t} = \tanh\left(W^W \circ \widehat{\mathrm{X}_t^W} + W^D \circ \widehat{\mathrm{X}_t^D} + W^R \circ \widehat{\mathrm{X}_t^R} + W^{NN} \circ \widehat{\mathrm{X}_t^{NN}}\right) \tag{9}$$

where $\circ$ is element-wise multiplication (i.e., Hadamard product); $\widehat{\mathrm{X}_t^W}$, $\widehat{\mathrm{X}_t^D}$, $\widehat{\mathrm{X}_t^R}$, and $\widehat{\mathrm{X}_t^{NN}}$ are the output of the weekly, daily, recent hourly, and nearest neighbor components, respectively; W^W, W^D, W^R, and W^{NN} are the learnable parameters that adjust the degrees affected by these individual components, respectively.

In this study, the predicted target is continuous data instead of discrete data, we thus utilize the mean absolute error (MAE) loss function as evaluation standard to minimize the error and train this model. The loss function can be calculated as:

$$loss(\theta) = MAE = \left\| X_t - \widehat{X_t} \right\| = \frac{1}{N}\sum_{k=1}^{N} |x_{t_k} - \widehat{x_{t_k}}| \tag{10}$$

Where θ are all learnable parameters in the proposed model, x_{t_k} denotes the actual value, $\widehat{x_{t_k}}$ represents the predicted value, and N is the total number of values needed to be predicted, i.e., the total number of regions (= $I \times J$).

3.7 Algorithm and Optimization

Similar to the training algorithm in Zhang et al. (2018), Algorithm 1 outlines the training process of the proposed ST-RCNet-knn model. At the first step, we construct the training instances from the original time series data. The proposed model is then trained via back-propagation using a batch size of 10. Moreover, since the Adam optimizer (Kingma and Ba 2014) has been widely used in machine learning and deep learning models, we train the model using this method to learn the learnable parameters.

Algorithm 1: ST-RCNet-knn Training Algorithm

Input: Historical observations: $\{X_0, \dots, X_{t-1}\}$;
lengths of the recent hourly, daily, and weekly sequence: a, b, c;
Output: Learned ST-RCNet-knn model

//construct training instances
$I \leftarrow \emptyset$
for all available time interval i ($c * 24 * 7 \le i \le t-1$) do
$S^H = [X_{i-a}, \cdots, X_{i-2}, X_{i-1}]$
$S^D = [X_{i-b*24}, \cdots, X_{i-2*24}, X_{i-24}]$
$S^W = [X_{i-c*24*7}, \cdots, X_{i-2*24*7}, X_{i-24*7}]$
//LAT and LON are the latitude and longitude vectors of all grids
$S^{NN} = [LAT, LON, X_{i-2}, X_{i-1}]$
//X_i is the target at time i
put a training instance ($\{S^H, S^D, S^W, S^{NN}\}, X_i$) into I

//train the model
initialize all learnable parameters θ in the ST-RCNet-knn
repeat
randomly select a batch of instances I_b from I
find θ by minimizing the loss function (i.e., equation (10)) with I_b
until stopping criteria is met

4. Evaluation

4.1 Evaluation settings

Datasets. Two datasets are used for the evaluation: a Tencent location dataset in Guangzhou (China), and a Taxicab dataset in Beijing (China).

- **TencentGZ**: The Tencent location dataset collected from the big data platform of Tencent (https://heat.qq.com) is adopted as the proxy of fine-scale crowd volume data. The data recorded all users' location requests through location-based services across a variety of Tencent products, including social media, gaming, travel, online shopping, communications and payment tools. Given the ubiquity of Tencent users in China, the Tencent location data have a better user representativeness than other alike data (e.g., Twitter data, Weibo data, taxi trace data, and bike-sharing data), and therefore they can better reflect the real crowd volume in a city. In this study, we use a Tencent location dataset generated from 1 to 27, November 2018 for the city Guangzhou in China. However, due to the absence of the data in one Wednesday, we remove all the data on Wednesdays. The dataset divided Guangzhou into 900 (=30*30) grid cells, each of which has a size of 1km x 1km. The total number of people in each grid cell was recorded every hour (See Figure 1 (right) for an example on a weekday at 18:00-19:00). The dataset is further divided into two parts: the data before 22 November are viewed as the training set, the rest as the testing set. Besides, we employ Min-Max normalization method to scale the data.
- **TaxiBJ**: To test the robustness of the proposed model, this study further employs the Beijing Taxicab dataset shared by Zhang et al. (2018), which divided Beijing into 1024 (= 32*32) grid cells. For each cell, the dataset recorded both crowd inflow (i.e., the total traffic of crowds entering this grid) and crowd outflow (i.e., the total traffic of crowds leaving this grid) in each hour. We select the inflow dataset from 42 consecutive days, wherein the first 35 days are the training set, the rest as the testing set. Again, we employ Min-Max normalization method to scale the data.

Baselines. We compare our proposed ST-RCNet-knn model with the following five baselines. The first five baselines are selected considering their popularity and publication dates.

- RF: It is a classic and popular machine learning model for regression task. RF is selected mainly because it is found to be easy to train, to have high performances and not to over-fit the data (Breiman 2001, Cutler, Cutler and Stevens 2012).
- ST-ResNet: It is a popular deep learning model proposed by Zhang et al. (2018), which integrates the residual units for enabling the model to have a deeper network and further capture spatiotemporal characteristics.
- Graph WaveNet: It integrates graph convolution layer and dilated casual convolution (Wu et al. 2019).
- GMAN: It includes a spatio-temporal embedding layer, ST-attention blocks and a transformer attention layer. It furtheruses node2vec to capture the topological relationships betweenintersections (Zheng et al. 2020).
- HRSSTs: It applies multi-head attention mechanism's transformer to portray the spatio-temporal features in closeness, period and trend patterns (Xu et al. 2022). It is a latest extension of Zhang et al. (2018).

Evaluation metrics. We are interested in both the predictive accuracy and the training time cost of the seven models.

- **Predictive accuracy**. The Mean Absolute Error (MAE), R^2, and Root Mean Squared Error (RMSE) are used to assess the predictive accuracy of the proposed model and the baselines.

$$MAE = \frac{1}{N}\sum_{k=1}^{N}|x_k - \widehat{x_k}| \tag{10}$$

$$R^2 = 1 - \frac{\sum_{k=1}^{N}(x_k - \widehat{x_k})^2}{\sum_{k=1}^{N}(x_k - \bar{x}_k)^2} \tag{11}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(x_k - \widehat{x_k})^2} \tag{12}$$

Where x and $\hat{x}$ are the ground truth and the predicted value, respectively; N is the number of all predicted values (i.e., the number of all regions; $N = 900$ for TencentGZ dataset; $N = 1024$ for TaxiBJ dataset); $\bar{x}$ represents the mean value of the ground truth.

- **Training time cost**. We measure the training time cost needed to allow a model to achieve a good result (i.e., when the test MAEs reach a minimal; by plotting training and test MAEs over different epochs). For a single epoch, the proposed model and the baselines have different execution time (due to different computational complexity). Therefore, we measure the training time cost as the total execution time in seconds. The hardware environment is based on Intel Core i7-8700K CPU and NVIDIA GeForce GTX 1070Ti. The software environment is Python 3.6 and tensorflow-gpu 2.0.0.

Main objectives of the evaluation. For the evaluation, we would like to answer the following questions:

1) How do our ST-RCNet-knn model and the five baselines perform in terms of predictive accuracy and the training time cost (Section 4.2)?

2) How does each component of our ST_RCNet-knn contribute to its prediction(Section 4.3)?

3) How are the predictive accuracies of our ST-RCNet-knn and the baselines influenced by the variation of the crowd volume (Section 4.4)?

4) How do our ST-RCNet-knn and the baselines perform when predicting crowd information during a large-scale special event (when the situation of massive people gathering in short time was happening) (Section 4.5)?

5) How does the predictive accuracy of our ST-RCNet-knn model vary spatially and temporally (Section 4.6)?

4.2 Model Comparison Results

4.2.1 Model comparison on different input lengths

Table 1 compares the predictive accuracies (MAE, R^2, and RMSE) and the training time costs of our proposed ST-RCNet-knn model and the baselines, under different input data lengths (i.e., lengths of the recent hourly, daily, and weekly sequence). We did not include RF, Graph WaveNet and GMAN in this comparison, since they employ a different architecture: They do not rely on weekly and daily sequences, but instead only make use the sequence from the last several hours. Table 1 show that for both datasets, our ST-RCNet-knn model outperforms the baselines in both the predictive accuracies and the training time costs under most of the input data lengths.

Predictive accuracy. From the perspective of predictive accuracy, regardless of how the input data lengths and dataset change, our ST-RCNet-knn model achieves almost the best accuracy among these three models via the three evaluation metrics.

For both datasets, ST-RCNet-knn greatly outperforms the ST-ResNet model. Compared to ST-ResNet, our ST-RCNet-knn model reduces the MAEs by 12.9% (minimum: 9.7%; maximum: 19.7%) on average for the TencentGZ dataset, and by 15.4% (minimum: -3%; maximum: 45.5%) on average for TaxiBJ. Similar advantages of ST-RCNet-knn against HRSSTs (which is a latest extension of ST-ResNet) can be also observed, with MAEs being reduced by average 14.1% (minimum: 7.3%; maximum: 24.3%) in the TencentGZ dataset and average 19.1% (minimum: 3.2%; maximum: 43%) in the TaxiBJ dataset. All these demonstrate that the proposed model structure might have a better ability in capturing the spatio-temporal dependences than the baseline models in these two datasets.

Furthermore, by analyzing the varied predictive accuracies of the three models under different combination of input data lengths, we found that the ST-RCNet-knn model has the smallest variation in predictive accuracy, while the other two baseline models show an obvious fluctuation. This suggests that, to some extent, our proposed ST-RCNet-knn model structure is more robust than the other two baseline models.

Training time cost. From the perspective of training time cost, the performance of the proposed ST-RCNet-knn model is extremely better than the other two models. In the TencentGZ dataset, the training time cost of our ST-RCNet-knn model is only average 16.8% and 43.3% of the baselines ST-ResNet and HRSSTs respectively. Similarly, in the TaxiBJ dataset, the training time cost of our ST-RCNet-knn model is only 28.6% and 33.8% on average of the baselines ST-ResNet and HRSSTs respectively. The training time cost of our ST-RCNet-knn model is only average 68.4% of ST-RCNet for the TencentGZ dataset, and average 71.8% of ST-RCNet for the TaxiBJ dataset.

Summary. The above results show that our ST-RCNet-knn model outperforms ST-ResNet and HRSSTs in terms of both the predictive accuracies and the training time cost under all input data lengths. This suggests that the proposed model, compare to these two baselines, is more suitable for large scale prediction task due to the low training cost under premise of excellent predictive accuracy.

Table 1. The predictive accuracies and the training time costs of the proposed ST-RCNet-knn model and the baselines, under different input data lengths (i.e., lengths of the weekly, daily, and recent hourly sequence (c, b, a))

Input lengths (c, b, a)	Metrics	TencentGZ			TaxiBJ		
		ST-RCNet-knn	ST-ResNet	HRSSTs	ST-RCNet-knn	ST-ResNet	HRSSTs
(1,1,1)	MAE	**8.253×10^{-3}**	9.601×10^{-3}	1.046×10^{-2}	6.164×10^{-3}	6.791×10^{-3}	7.936×10^{-3}
	R^2	**96.25%**	95.2%	94.8%	**98.86%**	98.65%	98.13%
	RMSE	**1.718×10^{-2}**	1.943×10^{-2}	2.023×10^{-2}	**1.161×10^{-2}**	1.264×10^{-2}	1.485×10^{-2}
	Time(s)	**90.3**	557.7	215.3	**219.3**	820.8	692.5
(1,1,2)	MAE	**8.254×10^{-3}**	9.92×10^{-3}	9.783×10^{-3}	**5.981×10^{-3}**	7.093×10^{-3}	9.432×10^{-3}
	R^2	**96.25%**	95.22%	95.03%	**98.9%**	98.47%	97.53%
	RMSE	**1.719×10^{-2}**	1.94×10^{-2}	1.978×10^{-2}	**1.113×10^{-2}**	1.346×10^{-2}	1.707×10^{-2}
	Time(s)	**92.9**	559.9	215.6	**222.9**	823.6	692.9
(1,1,3)	MAE	**8.295×10^{-3}**	1.003×10^{-2}	9.352×10^{-3}	**6.068×10^{-3}**	7.377×10^{-3}	7.653×10^{-3}
	R^2	**96.23%**	94.97%	95.38%	**98.90%**	98.41%	98.42%
	RMSE	**1.722×10^{-2}**	1.989×10^{-2}	1.907×10^{-2}	**1.139×10^{-2}**	1.37×10^{-2}	1.365×10^{-2}
	Time(s)	**95.9**	557.1	215	**232.3**	818.7	694.3

(1,1,4)	MAE	**8.358×10^{-3}**	9.78×10^{-3}	9.697×10^{-3}	**6.006×10^{-3}**	6.767×10^{-3}	6.659×10^{-3}
	R^2	**96.25%**	95.10%	95.23%	**98.93%**	98.63%	98.75%
	RMSE	**1.719×10^{-2}**	1.965×10^{-2}	1.937×10^{-2}	**1.125×10^{-2}**	1.274×10^{-2}	1.216×10^{-2}
	Time(s)	**97.1**	555.6	215.2	**235.0**	822.3	693.5
(1,2,1)	MAE	**8.212×10^{-3}**	9.49×10^{-3}	9.817×10^{-3}	**6.217×10^{-3}**	7.027×10^{-3}	7.326×10^{-3}
	R^2	**96.27%**	95.3%	95.08%	**98.83%**	98.56%	98.48%
	RMSE	**1.715×10^{-2}**	1.923×10^{-2}	1.968×10^{-2}	**1.174×10^{-2}**	1.305×10^{-2}	1.341×10^{-2}
	Time(s)	**92.9**	555.3	214.4	**222.9**	821.5	693
(1,2,2)	MAE	**8.165×10^{-3}**	9.633×10^{-3}	1.025×10^{-2}	**6.055×10^{-3}**	6.556×10^{-3}	7.315×10^{-3}
	R^2	**96.29%**	95.40%	94.93%	**98.9%**	98.83%	98.35%
	RMSE	**1.709×10^{-2}**	1.902×10^{-2}	1.998×10^{-2}	**1.139×10^{-2}**	1.177×10^{-2}	1.395×10^{-2}
	Time(s)	**95.4**	557.5	214	**228.6**	820.5	695.1
(1,2,3)	MAE	8.34×10^{-3}	9.55×10^{-3}	1.053×10^{-2}	**5.998×10^{-3}**	7.276×10^{-3}	6.901×10^{-3}
	R^2	**96.21%**	95.33%	94.52%	**98.94%**	98.48%	98.66%
	RMSE	**1.727×10^{-2}**	1.918×10^{-2}	2.078×10^{-2}	**1.121×10^{-2}**	1.341×10^{-2}	1.258×10^{-2}
	Time(s)	**97**	569.4	215.8	**236.9**	808.3	696
(1,2,4)	MAE	**8.307×10^{-3}**	9.630×10^{-3}	9.506×10^{-3}	**5.939×10^{-3}**	8.489×10^{-3}	6.701×10^{-3}
	R^2	**96.2%**	95.29%	95.29%	**98.96%**	97.85%	98.73%
	RMSE	**1.715×10^{-2}**	1.925×10^{-2}	1.926×10^{-2}	**1.109×10^{-2}**	1.594×10^{-2}	1.226×10^{-2}
	Time(s)	**98.9**	562.9	216.4	**239.0**	822.3	695.4
(1,3,1)	MAE	**8.215×10^{-3}**	1.024×10^{-2}	9.393×10^{-3}	**6.194×10^{-3}**	6.878×10^{-3}	9.067×10^{-3}
	R^2	**96.29%**	95.1%	95.41%	**98.83%**	98.67%	97.98%
	RMSE	**1.709×10^{-2}**	1.965×10^{-2}	1.9×10^{-2}	**1.176×10^{-2}**	1.255×10^{-2}	1.546×10^{-2}
	Time(s)	**96.0**	564.2	214.8	**232.6**	822.1	695.3
(1,3,2)	MAE	8.305×10^{-3}	9.357×10^{-3}	9.253×10^{-3}	6.029×10^{-3}	6.58×10^{-3}	6.499×10^{-3}
	R^2	**96.24%**	95.49%	95.41%	**98.93%**	98.77%	98.82%
	RMSE	**1.721×10^{-2}**	1.884×10^{-2}	1.902×10^{-2}	**1.125×10^{-2}**	1.207×10^{-2}	1.118×10^{-2}
	Time(s)	**98.1**	566.8	215.2	**237.3**	822	696.7
(1,3,3)	MAE	**8.260×10^{-3}**	9.411×10^{-3}	9.818×10^{-3}	**5.938×10^{-3}**	6.542×10^{-3}	7.844×10^{-3}
	R^2	**96.27%**	95.32%	95.17%	**98.97%**	98.77%	98.34%
	RMSE	**1.713×10^{-2}**	1.919×10^{-2}	1.951×10^{-2}	**1.105×10^{-2}**	1.206×10^{-2}	1.398×10^{-2}
	Time(s)	**97**	565.6	216	**244.9**	825.4	697.1
(1,3,4)	MAE	**8.222×10^{-3}**	9.322×10^{-2}	9.399×10^{-3}	**6.02×10^{-3}**	6.905×10^{-3}	7.606×10^{-3}
	R^2	**96.28%**	95.43%	95.43%	**98.94%**	98.59%	98.41%
	RMSE	**1.712×10^{-2}**	1.897×10^{-2}	1.898×10^{-2}	1.112×10^{-2}	1.291×10^{-2}	1.372×10^{-2}
	Time(s)	**95.8**	565.2	216.7	**249.4**	825.5	697.3
(2,1,1)	MAE	**8.402×10^{-3}**	9.54×10^{-3}	9.77×10^{-3}	**6.656×10^{-3}**	7.699×10^{-3}	8.42×10^{-3}
	R^2	**96.2%**	95.46%	93.78%	**98.44%**	97.89%	97.49%
	RMSE	**1.729×10^{-2}**	1.891×10^{-2}	2.212×10^{-2}	**1.359×10^{-2}**	1.579×10^{-2}	1.723×10^{-2}
	Time(s)	**85.3**	563.5	214.1	**222.4**	821	693
(2,1,2)	MAE	**8.285×10^{-3}**	9.417×10^{-3}	9.76×10^{-2}	**6.382×10^{-3}**	1.164×10^{-2}	1.112×10^{-2}
	R^2	**96.24%**	95.42%	95.08%	**98.64%**	91.58%	93.13%
	RMSE	**1.721×10^{-2}**	1.899×10^{-2}	1.969×10^{-2}	**1.269×10^{-2}**	3.153×10^{-2}	2.849×10^{-2}
	Time(s)	**88.9**	556.3	215.5	**229**	825.7	696
(2,1,3)	MAE	8.416×10^{-3}	9.413×10^{-2}	9.367×10^{-3}	**6.964×10^{-3}**	8.088×10^{-3}	8.0×10^{-3}
	R^2	**96.19%**	95.35	95.2%	**98.37%**	97.46%	97.72%
	RMSE	**1.731×10^{-2}**	1.913×10^{-2}	1.944×10^{-2}	**1.386×10^{-2}**	1.733×10^{-2}	1.643×10^{-2}
	Time(s)	**90.1**	547.6	215.5	**235.9**	824	697
(2,1,4)	MAE	**8.572×10^{-3}**	9.596×10^{-3}	9.462×10^{-3}	**7.518×10^{-3}**	8.422×10^{-3}	7.777×10^{-3}
	R^2	**96.1%**	95.31%	95.17%	**97.8%**	97.11%	97.69%

	RMSE	**1.751×10^{-2}**	1.922×10^{-2}	1.951×10^{-2}	**1.611×10^{-2}**	1.848×10^{-2}	1.653×10^{-2}
	Time(s)	**93.4**	544.6	216.8	**241.2**	823.4	695
(2,2,1)	MAE	**8.291×10^{-3}**	9.338×10^{-3}	1.095×10^{-3}	**6.718×10^{-3}**	7.199×10^{-3}	1.021×10^{-2}
	R^2	**96.27%**	95.56%	94.76%	**98.45%**	98.27%	94.92%
	RMSE	**1.714×10^{-2}**	1.87×10^{-2}	2.031×10^{-2}	**1.352×10^{-2}**	1.429×10^{-2}	2.449×10^{-2}
	Time(s)	**87.4**	545.5	214.8	**227.2**	823.9	694.3
(2,2,2)	MAE	**8.633×10^{-3}**	9.763×10^{-3}	9.317×10^{-3}	**6.683×10^{-3}**	1.225×10^{-2}	8.222×10^{-3}
	R^2	**96.19%**	95.32%	95.46%	**98.50%**	89.95%	97.38%
	RMSE	**1.732×10^{-2}**	1.919×10^{-2}	1.891×10^{-2}	**1.333×10^{-2}**	3.445×10^{-2}	1.758×10^{-2}
	Time(s)	**88.8**	547.5	215.7	**231.9**	823.7	695.8
(2,2,3)	MAE	**8.307×10^{-3}**	9.357×10^{-3}	9.874×10^{-3}	6.638×10^{-3}	7.363×10^{-3}	7.47×10^{-3}
	R^2	**96.22%**	95.49%	95.03%	98.51%	98.04%	98%
	RMSE	**1.725×10^{-2}**	1.884×10^{-2}	1.978×10^{-2}	**1.327×10^{-2}**	1.523×10^{-2}	1.536×10^{-2}
	Time(s)	**94.1**	546.3	216.2	**243.5**	818.5	697
(2,2,4)	MAE	**8.325×10^{-3}**	9.625×10^{-3}	9.612×10^{-3}	**6.759×10^{-3}**	7.897×10^{-3}	8.278×10^{-3}
	R^2	**96.19%**	95.33%	95.39%	**98.44%**	97.51%	97.46%
	RMSE	**1.731×10^{-2}**	1.917×10^{-2}	1.906×10^{-2}	**1.357×10^{-2}**	1.715×10^{-2}	1.733×10^{-2}
	Time(s)	**92.8**	549.9	217.5	**244**	826.9	699.1
(2,3,1)	MAE	**8.23×10^{-3}**	9.517×10^{-3}	9.46×10^{-3}	**6.969×10^{-3}**	7.314×10^{-3}	8.93×10^{-3}
	R^2	**96.25%**	95.41%	95.33%	**98.25%**	98.1%	96.69%
	RMSE	**1.719×10^{-2}**	1.902×10^{-2}	1.919×10^{-2}	**1.437×10^{-2}**	1.5×10^{-2}	1.976×10^{-2}
	Time(s)	**89.9**	545.3	215.7	**235.9**	828.2	697.3
(2,3,2)	MAE	**8.228×10^{-3}**	9.233×10^{-3}	9.471×10^{-3}	7.727×10^{-3}	**7.465×10^{-3}**	8.919×10^{-3}
	R^2	**96.25%**	95.67%	95.33%	97.91%	**98.03%**	96.58%
	RMSE	**1.714×10^{-2}**	1.846×10^{-2}	1.943×10^{-2}	1.571×10^{-2}	**1.526×10^{-2}**	2.009×10^{-2}
	Time(s)	**94.4**	548.1	215.3	**243.4**	828.3	698.9
(2,3,3)	MAE	**8.40×10^{-3}**	9.299×10^{-2}	9.68×10^{-3}	**6.879×10^{-3}**	7.853×10^{-3}	8.216×10^{-3}
	R^2	**96.23%**	95.51%	95.02%	**98.36%**	97.68%	97.47%
	RMSE	**1.722×10^{-2}**	1.881×10^{-2}	1.979×10^{-2}	**1.392×10^{-2}**	1.654×10^{-2}	1.728×10^{-2}
	Time(s)	**94.5**	538.4	216.6	**249.4**	828.9	700.4
(2,3,4)	MAE	**8.327×10^{-3}**	9.257×10^{-3}	8.983×10^{-3}	**6.818×10^{-3}**	1.041×10^{-2}	7.86×10^{-3}
	R^2	**96.24%**	95.59%	95.6%	**98.41%**	93.8%	97.7%
	RMSE	**1.72×10^{-2}**	1.864×10^{-2}	1.856×10^{-2}	**1.372×10^{-2}**	2.708×10^{-2}	1.649×10^{-2}
	Time(s)	**95.6**	547.4	217.9	**253.5**	830.6	701.4

4.2.2 Model comparison with all baselines

To further evaluate the performances between our ST-RCNet-knn model and all baselines, we chose the best model (in terms of predictive accuracy) after turning the hyperparameters for each of all baselines. Specifically, for GMAN, we used the recent 5 hours and 7 hours for training TencentGZ and TaxiBJ respectively. As in the Zheng et al. (2020), we set the number of layers, attention heads and dimensions to (3, 4, 6) and (2, 3, 8) for GMAN on TencentGZ and TaxiBJ respectively. For Graph WaveNet, we set the numbers of recent hours, sequences of dilation factors and diffusion steps to (12, 8, 1) and (15, 8, 1) on TencentGZ and TaxiBJ respectively. According to Table 1, we selected the inputs combination that resulted in the best MAE performance. Therefore, we selected the (1, 2, 2), (2, 3, 2) and (2, 3, 4) for ST-RCNet-knn, ST-ResNet and HRSSTs on TencentGZ, along with (1, 3, 3), (1, 3, 3) and (1, 3, 2) for them on TaxiBJ. Additionally, we add one of variants of ST-RCNet-knn, ST-RCNet (i.e., without the k-NN part), in this comparison. We selected the inputs of (2,1,1) and (1,1,2) on TencentGZ and TaxiBJ respectively for ST-RCNet. By setting these values, we ensure that the comparison is based on the best predictive performance of all models, making it a fair comparison.

The comparison results are presented in Table 2. It shows that our ST-RCNet-knn model outperforms all baselines on both datasets in terms of predictive accuracy and training time cost, followed by its trimmed version ST-RCNet (i.e., without the k-NN part). Compared to the other baselines (ST-ResNet, GMAN, Graph WaveNet, HRSSTs, and RF), our ST-RCNnet-knn reduces their MAEs by 18.76% on average (minimum: 4.00%; maximum: 51.08%) in the TencentGZ dataset, and more importantly, the training time cost of our ST-RCNnet-knn is only about 25.65% (minimum: 1.07%; maximum: 57.96%) of their training time costs. Similar results can be found for the TaxiBJ dataset. Additionally, the GMAN and Graph WaveNet are worse than the other models, probably because these two models make predictions only based on the last several hours, without making use of the weekly and daily patterns. In summary, compared to the baselines, our model significantly reduces the training time cost, while still maintaining a better predictive accuracy.

Table 2. The performance of all models

Data	Metric	ST-RCNet-knn	ST-RCNet	ST-ResNet	GMAN	Graph WaveNet	HRSSTs	RF
Tencent GZ	MAE	**8.17×10^{-3}**	8.24×10^{-3}	9.23×10^{-3}	9.99×10^{-3}	1.67×10^{-2}	8.98×10^{-3}	8.51×10^{-3}
	R^2	**96.29%**	96.19%	95.67%	94.66%	84.43%	95.6%	95.67%
	RMSE	**1.71×10^{-2}**	1.73×10^{-2}	1.85×10^{-2}	2.05×10^{-2}	3.5×10^{-2}	1.86×10^{-2}	1.85×10^{-2}
	Time(s)	**95.4**	129	548.1	8934.2	1185.02	217.9	164.6
TaxiBJ	MAE	**5.94×10^{-3}**	5.95×10^{-3}	6.54×10^{-3}	7.09×10^{-3}	1.63×10^{-2}	6.50×10^{-3}	7.29×10^{-3}
	R^2	**98.97%**	98.9%	98.77%	98.42%	90.6%	98.82%	98.31%
	RMSE	**1.11×10^{-2}**	1.12×10^{-2}	1.21×10^{-2}	1.37×10^{-2}	3.33×10^{-2}	1.12×10^{-2}	1.41×10^{-2}
	Time(s)	**244.9**	310.2	825.4	16212.8	2713.6	696.7	422.4

Figure 5 shows the learning curves of all models on the two datasets. Since each model has different computational complexity, Table 3 presents their computation time for each epoch. For both the training and test curves, our ST-RCNet-knn model has better performance of training convergence on the two datasets. This suggests that the proposed combination of GRU, CNN and k-NN together with the hourly, daily, and weekly sequences is more efficient in capturing the spatio-temporal dependences of the data, and therefore leads to faster training convergence.

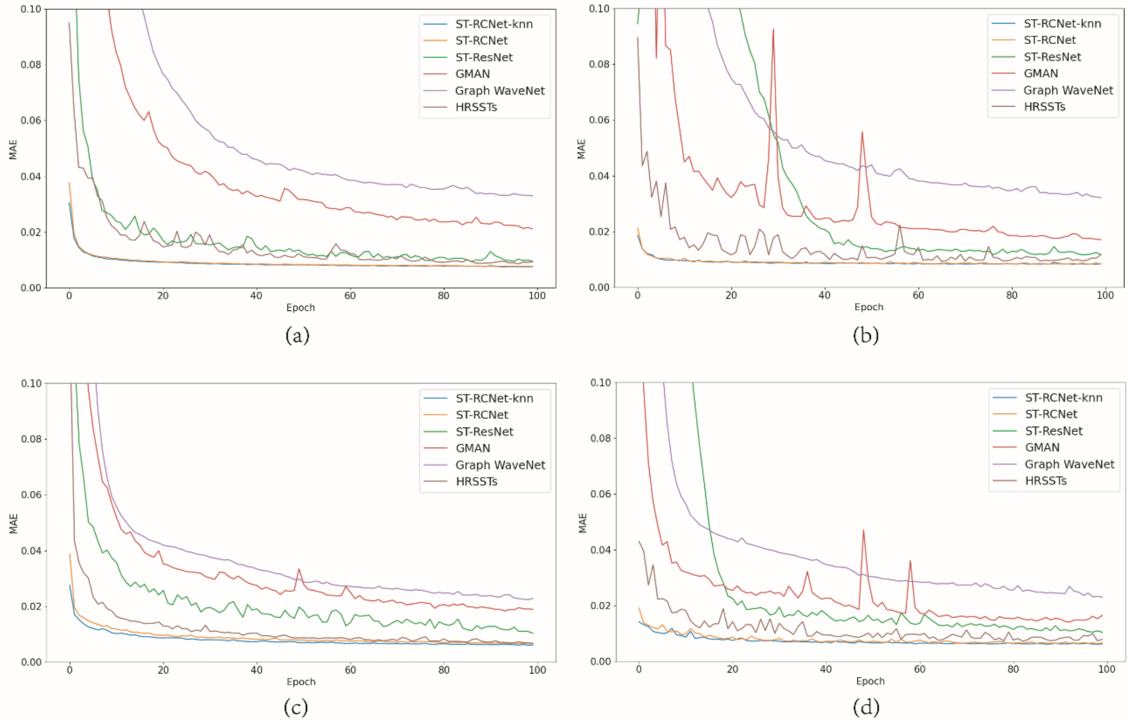

Figure 5. The training curve of all DL-based models: (a) – training curves on TencentGZ; (b) – test curves on TencentGZ; (c) – training curves on TaxiBJ; (d) – test curves on TaxiBJ. Note that different models have different computational time in an epoch (See Table 4).

Table 3. The computation time (in second) for each training epoch.

Data	ST-RCNet-knn	ST-RCNet	ST-ResNet	GMAN	Graph WaveNet	HRSSTs
TencentGZ	0.6	0.5	0.5	10.5	0.4	1.2
TaxiBJ	1	1.1	0.9	16.2	0.9	3.1

4.3 Ablation study

To investigate how effective each component contributes to the predictive performances of our ST_RCNet_knn, we compare the four variants by removing the parts of k-NN, weekly, daily, and recent hourly pattern from ST-RCNet-knn separately. For each of these four variants, we select the inputs combination with best performance from all inputs combinations. Table 4 shows that ST-RCNet-knn (i.e., with all four components) outperforms all variants in terms of accuracy, indicating that considering all these four components together (k-NN part, weekly, daily, and recent hourly pattern) can better model the complex spatio-temporal dependencies. Additionally, compared to removing k-NN part, the models with k-NN lead to a much lower training time. This suggests that the k-NN part is helpful to accelerate the convergence of loss function, and thus reduce the training time cost.

Table 4. The comparison between variants.

Data	Metric	ST-RCNet-knn	without k-NN	without weekly	without daily	without hourly
	MAE	**8.17×10^{-3}**	8.24×10^{-3}	8.34×10^{-3}	8.2×10^{-3}	8.2×10^{-3}

TencentGZ	R^2	**96.29%**	96.19%	96.14%	96.26%	96.27%
	RMSE	**1.71×10^{-2}**	1.73×10^{-2}	1.74×10^{-2}	1.72×10^{-2}	**1.71×10^{-2}**
	Time(s)	95.4	129	88	88.5	**82.6**
TaxiBJ	MAE	**5.94×10^{-3}**	5.95×10^{-3}	5.97×10^{-3}	5.97×10^{-3}	7.32×10^{-3}
	R^2	**98.97%**	98.9%	98.95%	98.9%	98.3%
	RMSE	**1.11×10^{-2}**	1.12×10^{-2}	**1.11×10^{-2}**	**1.11×10^{-2}**	1.41×10^{-2}
	Time(s)	244.9	310.2	**203.1**	235.6	237.4

4.4 Influences of the crowd volume variation on the model predictions

This section investigates how the variation of the hourly crowd volumes in each grid cell influences the predictive accuracies of the seven models. The same hyperparameter values as in Section 4.2.2 were used. We focused on the TencentGZ dataset, and filtered out the grid cells where the hourly crowd volume (i.e., the number of people presented in the region) has never exceeded 100. We therefore removed 197 cells (out of the total 900 ones).

Figure 6 shows the results of the MAEs. The x-axis represents the standard deviation of the hourly crowd volumes in each grid cell, which is then classified into three groups. The y-axis shows the MAEs of the seven models. As shown in Figure 6, the MAEs of the seven models increase greatly with the increase of the standard deviation of the crowd volumes. This is expected as grid cells with frequently changing crowd volumes are more difficult to predict than cells with less changes. More importantly, our ST-RCNet-knn model is better than the baselines in general, and the MAE gaps between ST-RCNet-knn and the baselines become bigger with the increase of the standard deviation of crowd volumes. Figure 7 shows the results of R^2. Similarly, our ST-RCNet-knn model is better than the baselines, especially when the standard deviation of crowd volume increase. The above results illustrate that compared to the baselines, our ST-RCNet-knn model is generally able to provide more accurate prediction of crowd volumes for both grid cells that are highly changing and cells that have less changes.

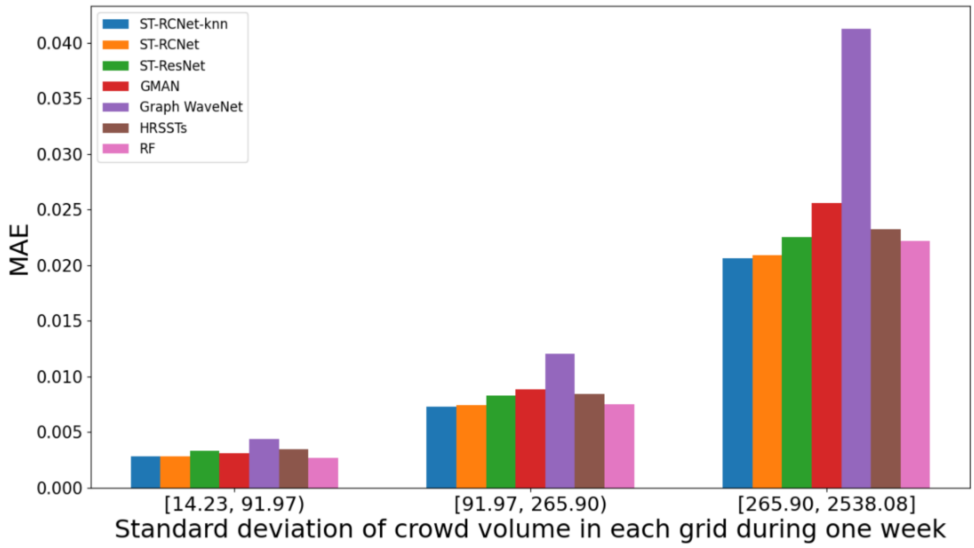

Figure 6. The MAEs of our ST-RCNet-knn model and the baselines on grid cells with different degrees of crowd change.

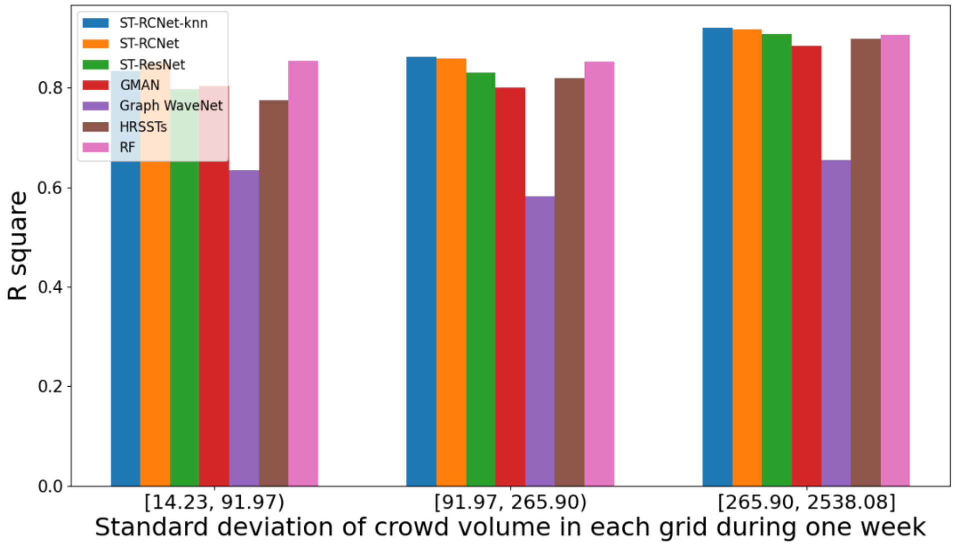

Figure 7. The R^2 of our ST-RCNet-knn model and the baselines on grid cells with different degrees of crowd change.

4.5 Prediction of crowd information influenced by a large-scale special event

This section aims to study how our ST-RCNet-knn model and the baselines perform when predicting crowd information during a large-scale special event. An international light festival of Guangzhou held on the time included in the TencentGZ dataset was found. The international light festival held on Monday, 26 November 2018 led to the large gathering of people at Zhujiang New Town which is the CBD of Guangzhou. It covers an area of 6.44 km^2. Figure 8 shows how the crowd volumes of the relevant grids differ on the light festival Monday and other Mondays. As shown in Figure 8, these two hourly trend lines were similar to each other before 17:00, while the obvious differences took place after 17:00. The international light festival was held at 19:00, hence the crowd volumes reached the peak at 19:00. People began to gradually leave after 22:00 when the light show was over. As shown in the literature, predicting crowd information during such a large-scale special event is very challenging (Ni, He and Gao 2017, Li et al. 2017a).

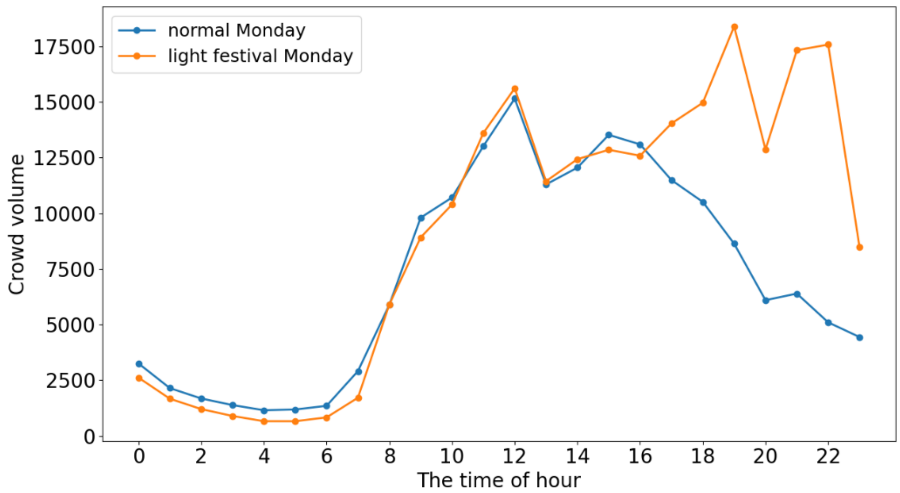

Figure 8. Varying trends of the crowd volumes of the relevant grid cells on the light festival Monday (i.e., 26 November 2018) and other Mondays.

In the following, we compare the performances of all models in predicting crowd volumes during the international light festival. Here, the same hyperparameter values as in Section 4.2.2 were used. The MAEs of different models changing over 24 hours on the light festival day are shown in Figure 9. As expected, for all models, the MAEs quickly increased during the period of the light festival. Importantly, the MAEs of our ST-RCNet-knn model (i.e., the blue line with circle) is below the MAEs of the baselines during the period of the light festival, except those of GMAN. Specifically, during the period from 17:00 to 23:00, our ST-RCNet-knn model reduces the MAEs by 21.9%, 15.7%, 14.2%, 22.7%, and 22.8% on average compared to ST-RCNet, ST-ResNet, Graph WaveNet, HRSSTs, and RF, respectively. Note that while GMAN model outperforms our ST-RCNet-knn model during the period, its training time cost is about 93 times more than our model. All these demonstrate that our ST-RCNet-knn model is able to achieve good predictive results with extremely low time cost, when the situation of large people gathering in short time was happening.

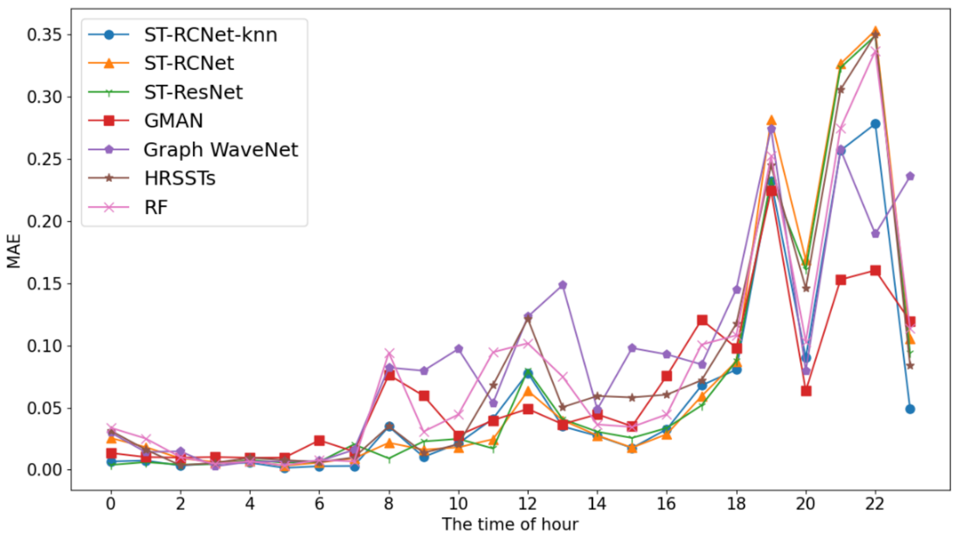

Figure 9. The predictive performance of our ST-RCNet-knn model and the baselines on the light festival day in Guangzhou

4.6 Spatial and temporal distributions of the predictive performance

Across the city, prediction errors are likely to vary spatially and temporally for a variety of reasons. To further examine the characteristics of our ST-RCNet-knn model, this section investigates how its predictive performance (focusing on MAE) varies spatially and temporally, using the TencentGZ dataset. Again, the input sequence lengths (1,2,2) are selected for the ST-RCNet-knn model.

Figure 10 shows the spatial distribution of the MAEs, comparing weekdays and weekends. According to the MAEs, we classify all the grid cells in Guangzhou into five groups using an equal interval classification scheme. As shown in Figure 6, 95% of the grids have low-level errors (i.e., the 2 categories with the lowest MAEs in the legends) for both weekday and weekend, which illustrates that our ST-RCNet-knn model has a high potential to be used in city management. Grid cells with a relatively high level of errors (i.e., the 2 categories with the highest MAEs in the legends) are mainly located around the urban villages in the neighboring area of Liwan, Haizhu, and Yuexiu District (old city center), as well as around the CBD located in Tianhe District (new city center). Specifically, one of the dark-orange cells is always located at an urban village near the most prosperous area of Guangzhou no matter on weekday or weekend. This is probably because a large percentage of population lives in narrow urban villages there (due to relatively low housing expenses and proximity to workplaces), and the high and irregular human mobility happens in these areas.

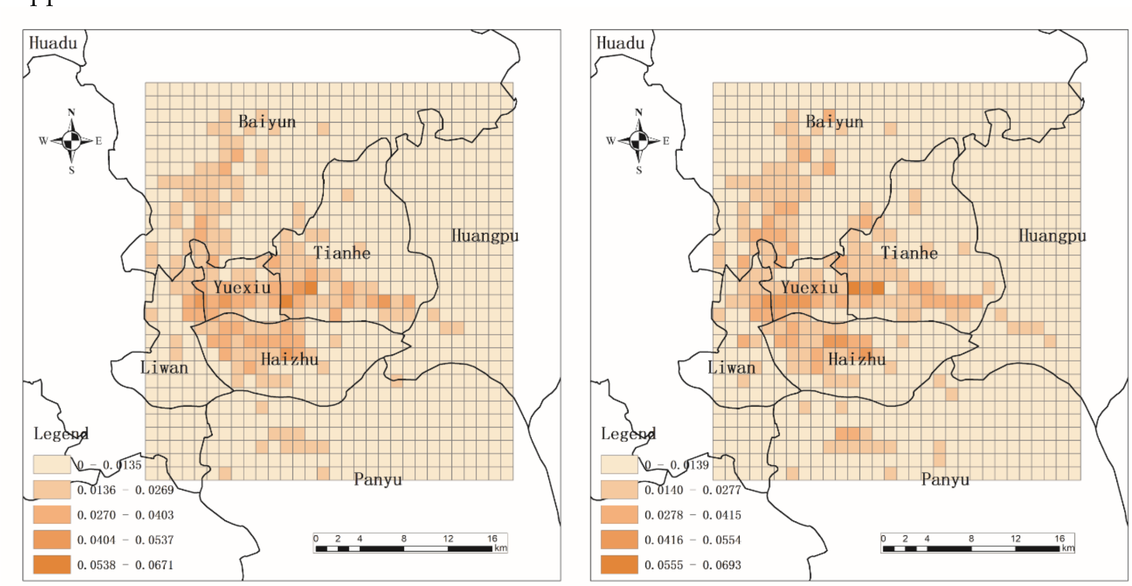

Figure 10. Spatial distribution of the predicting errors (MAEs) of our ST-RCNet-knn model, comparing weekdays (left) and weekends (right).

Figure shows the temporal distribution of the predicting errors. As shown in Figure 11, the MAEs of our ST-RCNet-knn model are relatively low from 1 am to 11 am, and start to increase and fluctuate for the rest of the day. This can be explained by the fact that most people often have relatively regular activity patterns (e.g., either resting at home or commuting) from 1 am to 11 am. And starting from the middays, people start to be involved in various activities over different areas in a city, which then significantly influences the crowd distribution in the city and makes it more difficult to achieve an accurate prediction.

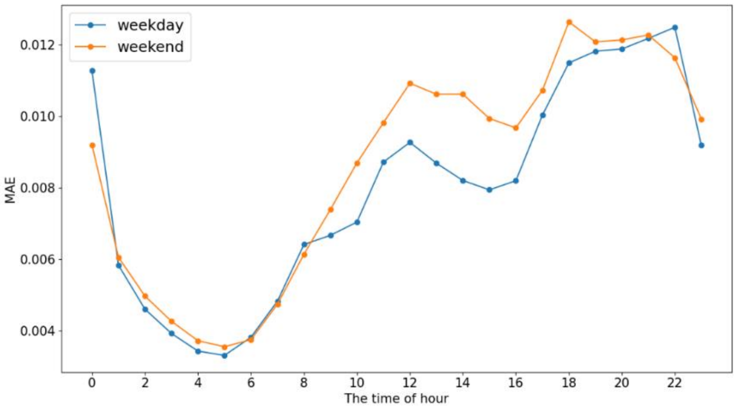

Figure 11. Temporal distribution of the predicting errors of our ST-RCNet-knn model.

5. Discussion

The main aim of this paper is to explore a model to reduce the training time cost while maintaining an excellent predictive accuracy in forecasting citywide crowd information at a fine spatio-temporal scale. To this end, we propose ST-RCNet-knn, which integrates two DL approaches (i.e., GRU and CNN) and a conventional ML method k-NN to jointly model the spatial and temporal dependences between any two regions in a city. Using two different datasets in two different cities, we show that, compared to the state-of-the-art models, our ST-RCNet-knn model performed better in terms of MAEs and R^2 for both datasets. More importantly, the training time costs of ST-RCNet-knn model are just about 26.16% on average (minimum: 1.07%; maximum: 57.98%). In other words, compared to the baselines, our model significantly reduces the training time cost, while still maintaining a better predictive accuracy. Comparison between ST-RCNet-knn and its trimmed version ST-RCNet (i.e., with the k-NN part) shows that adding k-NN reduces the training time costs to approximately 76.45% of ST-RCNet. The above results demonstrate that compared to the baselines, our ST-RCNet-knn model can better capture both temporal dependences (via the GRU part) and spatial dependences (via the CNN and k-NN part). Meanwhile, due to the relatively simpler and shallower structure, our ST-RCNet-knn model leads to a significant less training time cost. Furthermore, the adding of k-NN to our model also helps to capture more spatial information and to accelerate the convergence of loss function, thus reducing the training time cost of the proposed model and further improving its predictive accuracy.

Checking the model performance when predicting crowd information during a large-scale special event (when massive crowds are gathering in short time), we found that our ST-RCNet-knn model outperforms almost all baselines and only slightly worse than GMAN in such a case (note that the training time cost of GMAN is about 93 times more than that of our model). This is desirable, as accurately predicting sudden massive crowds gathering is of vital importance to applications related to public safety and helps to prevent potential catastrophic accidents. The evaluation results also show that the MAEs of all models increase with the increase of the standard deviation of the crowd volumes, and R^2 of all models increase with the increase of the standard deviation of the crowd volumes. However, our ST-RCNet-knn model is generally able to provide more accurate prediction for both regions that are highly changing and those with less changes. Such advantages of our model under large-scale events and crowd volume variation might be due to the proposed combination of GRU, CNN and k-NN together with the hourly, daily, and weekly sequences.

In summary, the evaluation results show that our ST-RCNet-knn model significantly outperforms the state-of-the-art models in terms of both the predictive accuracies and the training time costs. Meanwhile, ST-RCNet-knn is able to make accurate prediction under the influences of large-scale special events with lowest training time cost, as well as robust to regions with various degrees of variations. All these suggest that our proposed model has a high potential in many applications (e.g., city management and transportation), in which forecasting citywide crowd information at a fine spatio-temporal scale is a key.

Several limitations of this study (and thus future work) should be noted. Firstly, in the evaluation, data were at an 1km x 1km spatial and 1 hour temporal resolution, and from two large cities. It would be interesting to investigate how our proposed model performs across different spatial and temporal scales, as well as in medium and small cities. Secondly, while our model allows to capture some spatio-temporal dependences, more explicit considerations of the underlying geographic features, e.g., land use, distance to city center, POI categories and their distribution, road/transportation network configuration, are still missing. Considering such geographic features might improve the "transferability" of a predictive model. Thirdly, despite being significantly better than the state-of-the-art model, the MAEs of our model still increase greatly with the increase of the standard deviation of the crowd volumes, as well as when the situation of massive crowd of people gathering in short time was happening. Further research attentions should be paid to such issues. Again, explicitly considering the underlying geographic contexts might be a potential solution.

6. Conclusion

This paper proposes a novel and efficient model (i.e., ST-RCNet-knn) to reduce the training time cost while maintaining an excellent predictive accuracy in forecasting citywide crowd information at a fine spatio-temporal scale. ST-RCNet-knn seamlessly integrates GRU, CNN and k-NN to jointly capture the spatial and temporal dependences in a citywide. The evaluation with two different datasets in two different cities shows that our ST-RCNet-knn significantly outperforms the state-of-the-art models in terms of predictive accuracy, training time cost, and abilities in making accurate prediction with lowest training time cost under the influences of large-scale special events and for regions with various degrees of variations. All these suggest that ST-RCNet-knn is an *effective*, *efficient*, and *reliable* method for forecasting citywide crowd information at a fine spatio-temporal scale, and has a high potential for many applications, such as city management, public safety, and transportation.

Further research attentions should be paid to improve the prediction under the influences of large-scale short-time and irregular events and for regions with high degrees of variations. Considering the underlying geographic features (e.g., land use, road/transportation network configuration), as well as external aspects (e.g., weather) might be a potential solution. Meanwhile, we are also interested in developing explainable AI techniques to better understand the capacities and limitations of the prediction models.

Acknowledgement

Xucai Zhang and Fangli Guan are supported by CSC (China Scholarship Council).

Declarations of interest

none

References

Ahas, R., A. Aasa, Y. Yuan, M. Raubal, Z. Smoreda, Y. Liu, C. Ziemlicki, M. Tiru & M. Zook (2015) Everyday space-time geographies: using mobile phone-based sensor data to monitor urban activity in Harbin, Paris, and Tallinn. *International Journal of Geographical Information Science,* 29**,** 2017-2039.

Breiman, L. (2001) Random Forests. *Machine Learning,* 45**,** 5-32.

Cai, L., K. Janowicz, G. Mai, B. Yan & R. Zhu (2020) Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting. *Transactions in GIS,* 24**,** 736-755.

Castro-Neto, M., Y. S. Jeong, M. K. Jeong & L. D. Han (2009) Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions. *Expert Systems with Applications,* 36**,** 6164-6173.

Chen, L., S. Wu, J. Chen, M. Li & F. Lu (2018) The near-real-time prediction of urban population distributions based on mobile phone location data. *Journal of Geo-information Science,* 20**,** 523-531.

Cho, K., B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk & Y. Bengio (2014) Learning phrase representations using RNN encoder-decoder for statistical machine translation. *preprint arXiv:1406.1078.*

Cutler, A., D. R. Cutler & J. R. Stevens. 2012. Random Forests. In *Ensemble Machine Learning: Methods and Applications,* eds. C. Zhang & Y. Ma, 157-175. Boston, MA: Springer US.

Demissie, M. G., G. Correia & C. Bento (2015) Analysis of the pattern and intensity of urban activities through aggregate cellphone usage. *Transportmetrica A-Transport Science,* 11**,** 502-524.

Fan, Z., X. Song, T. Xia, R. Jiang, R. Shibasaki & R. Sakuramachi (2018) Online Deep Ensemble Learning for Predicting Citywide Human Mobility. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies,* 2**,** 1-21.

Geng, X., Y. G. Li, L. Y. Wang, L. Y. Zhang, Q. Yang, J. P. Ye, Y. Liu & Aaai. 2019. Spatiotemporal Multi-Graph Convolution Network for Ride-Hailing Demand Forecasting. In *33rd AAAI Conference on Artificial Intelligence / 31st Innovative Applications of Artificial Intelligence Conference / 9th AAAI Symposium on Educational Advances in Artificial Intelligence*, 3656-3663. Honolulu, HI.

Gu, J. X., Z. H. Wang, J. Kuen, L. Y. Ma, A. Shahroudy, B. Shuai, T. Liu, X. X. Wang, G. Wang, J. F. Cai & T. Chen (2018) Recent advances in convolutional neural networks. *Pattern Recognition,* 77**,** 354-377.

Guo, S., Y. Lin, N. Feng, C. Song & H. Wan (2019) Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence,* 33**,** 922-929.

Hamner, B. 2010. Predicting Travel Times with Context-Dependent Random Forests by Modeling Local and Aggregate Traffic Flow. In *2010 IEEE International Conference on Data Mining Workshops*, 1357-1359.

Hong, W. C. (2011) Traffic flow forecasting by seasonal SVR with chaotic simulated annealing algorithm. *Neurocomputing,* 74**,** 2096-2107.

Jarv, O., R. Ahas & F. Witlox (2014) Understanding monthly variability in human activity spaces: A twelve-month study using mobile phone call detail records. *Transportation Research Part C: Emerging Technologies,* 38**,** 122-135.

Kingma, D. P. & J. Ba (2014) Adam: A Method for Stochastic Optimization. *preprint arXiv:1412.6980.*

Lecun, Y., L. Bottou, Y. Bengio & P. Haffner (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE,* 86**,** 2278-2324.

Li, Y., X. Wang, S. Sun, X. Ma & G. Lu (2017a) Forecasting short-term subway passenger flow under special events scenarios using multiscale radial basis function networks. *Transportation Research Part C: Emerging Technologies,* 77**,** 306-328.

Li, Y., R. Yu, C. Shahabi & Y. Liu (2017b) Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *preprint arXiv:1707.01926*.

Liu, Y., Z. Liu & R. Jia (2019) DeepPF: A deep learning based architecture for metro passenger flow prediction. *Transportation Research Part C: Emerging Technologies,* 101**,** 18-34.

Liu, Z., Y. Y. Du, J. W. Ya, F. Y. Liang, T. Ma & T. Pei (2020) Quantitative estimates of collective geo-tagged human activities in response to typhoon Hato using location-aware big data. *International Journal of Digital Earth,* 13**,** 1072-1092.

Luca, M., G. Barlacchi, B. Lepri & L. Pappalardo (2021) A Survey on Deep Learning for Human Mobility. *ACM Comput. Surv.,* 55**,** Article 7.

Ma, X., Z. Dai, Z. He, J. Ma, Y. Wang & Y. Wang (2017) Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction. *Sensors,* 17.

Ni, M., Q. He & J. Gao (2017) Forecasting the subway passenger flow under event occurrences with social media. *IEEE Transactions on Intelligent Transportation Systems,* 18**,** 1623-1632.

Okutani, I. & Y. J. Stephanedes (1984) Dynamic prediction of traffic volume through kalman filtering theory. *Transportation Research Part B-Methodological,* 18**,** 1-11.

Semanjski, I., S. Gautama, R. Ahas & F. Witlox (2017) Spatial context mining approach for transport mode recognition from mobile sensed big data. *Computers Environment and Urban Systems,* 66**,** 38-52.

Smith, B. L., B. M. Williams & R. K. Oswald (2002) Comparison of parametric and nonparametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies,* 10, 303-321.

Sun, S. L., C. S. Zhang & G. Q. Yu (2006) A Bayesian network approach to traffic flow forecasting. *Ieee Transactions on Intelligent Transportation Systems,* 7, 124-132.

Van Der Voort, M., M. Dougherty & S. Watson (1996) Combining Kohonen maps with ARIMA time series models to forecast traffic flow. *Transportation Research Part C: Emerging Technologies,* 4, 307-318.

Vinyals, O., A. Toshev, S. Bengio, D. Erhan & Ieee. 2015. Show and tell: a neural image caption generator. In *2015 IEEE Conference on Computer Vision and Pattern Recognition*, 3156-3164.

Williams, B. M. & L. A. Hoel (2003) Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of Transportation Engineering,* 129, 664-672.

Wu, C. H., J. M. Ho & D. T. Lee (2004) Travel-time prediction with support vector regression. *Ieee Transactions on Intelligent Transportation Systems,* 5, 276-281.

Wu, Z., S. Pan, G. Long, J. Jiang & C. Zhang (2019) Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*.

Xia, D. W., B. F. Wang, H. Q. Li, Y. T. Li & Z. L. Zhang (2016) A distributed spatial-temporal weighted model on MapReduce for short-term traffic flow forecasting. *Neurocomputing,* 179, 246-263.

Xie, P., T. Li, J. Liu, S. Du, X. Yang & J. Zhang (2020) Urban flow prediction from spatiotemporal data using machine learning: A survey. *Information Fusion,* 59, 1-12.

Xu, M., W. Dai, C. Liu, X. Gao, W. Lin, G.-J. Qi & H. Xiong (2020) Spatial-temporal transformer networks for traffic flow forecasting. *preprint arXiv:2001.02908*.

Xu, Z., Y. Kang & Y. Cao (2022) High-Resolution Urban Flows Forecasting with Coarse-Grained Spatiotemporal Data. *IEEE Transactions on Artificial Intelligence*.

Yao, H., X. Tang, H. Wei, G. Zheng & Z. Li (2019) Revisiting Spatial-Temporal Similarity: A Deep Learning Framework for Traffic Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence,* 33, 5668-5675.

Yuan, H., X. Zhu, Z. Hu & C. Zhang (2020) Deep multi-view residual attention network for crowd flows prediction. *Neurocomputing,* 404, 198-212.

Zhang, J., Y. Zheng, D. Qi, R. Li & X. Yi. 2016. DNN-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '16*, 1-4.

Zhang, J. B., Y. Zheng, D. K. Qi, R. Y. Li, X. W. Yi & T. R. Li (2018) Predicting citywide crowd flows using deep spatio-temporal residual networks. *Artificial Intelligence,* 259, 147-166.

Zhang, X., R. Cao, Z. Zhang & Y. Xia. 2020. Crowd Flow Forecasting with Multi-Graph Neural Networks. In *2020 International Joint Conference on Neural Networks (IJCNN),* 1-7.

Zhao, L., Y. J. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng & H. F. Li (2020) T-GCN: A Temporal Graph Convolutional Network for Traffic Prediction. *IEEE Transactions on Intelligent Transportation Systems,* 21, 3848-3858.

Zheng, C., X. Fan, C. Wang & J. Qi (2020) GMAN: A Graph Multi-Attention Network for Traffic Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence,* 34, 1234-1241.