# Deep Learning Models for Population Flow Generation from Aggregated Mobility Data

**Can Rong**[1], **Jie Feng**[2], **Yong Li**[2]
[1]Department of Computer Science, Peking University, Beijing 100084, China
[2]Beijing National Research Center for Information Science and Technology,
Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
liyong07@tsinghua.edu.cn

## ABSTRACT

Population flow data (traffic sets of crowds from one regions to another) is of great value in a wide range of fields from urban traffic resource allocation to public security protection. Since the data of the individual-level mobility requires privacy protection, it's hard to collect detailed population flow data. There have been published works on generating population flow from aggregated data. But they have the limitations of considering the flow between neighbors only or modeling the mapping from aggregated population data to flow by a simple physical model without taking regionally diversity into account. However, long-range dependencies and regionally diversity is very important. Since population flow contains more information than that in aggregated population variation, generating the detailed former from the aggregated latter is quite difficult. In this paper, we proposed an end-to-end structure deep learning based model to generate population flow from aggregated historical population variation data. We use real-world datasets to compare the performance of our model with several baselines, which shows the superiority of our model. This proves the potential of using deep learning in population flow generation.

**ACM Reference Format:**
Can Rong[1], Jie Feng[2], Yong Li[2] . 2019. Deep Learning Models for Population Flow Generation from Aggregated Mobility Data. In *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and the 2019 International Symposium on Wearable Computers (UbiComp/ISWC '19 Adjunct), September 9–13, 2019, London, United Kingdom.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3341162.3349319


*UbiComp/ISWC '19 Adjunct, September 9–13, 2019, London, United Kingdom*

ACM ISBN 978-1-4503-6869-8/19/09…$15.00
https://doi.org/10.1145/3341162.3349319

## 1 INTRODUCTION

The population flow data, which means statistics on the number of people moving between regions, has a great value on applications in urban traffic resource allocation, public security protection and so on. Obtaining population flow data is quite a hindrance due to the high cost of existing approaches and privacy leakage. However, getting aggregated population variation data is relatively simple. Thus, it's important to find a way of generating population flow data from aggregated population variation data.

Some works [11, 17, 18, 22] have been done to generate population flow or individuals' trajectories from aggregated population data. Iwata et al. [11, 18] generated flow data from neighbors. But the long-range dependencies is also important, because modern transportation, such as subway and expressway, brings long-distance transportation in a short time interval. Xu et al. [17, 22] consider long-range dependencies but have other problems. Xu et al. [22] use a simple physical model without dealing with the complex interaction between all regions and dynamic complexity in time. Tanaka et al. [17] ignore that different regions have different flow patterns.

In summary, it's not easy to generate the population flow from aggregated data due to 2 challenges. First, the interactions between a region with all others are a lot in quantity and variety. Second, the interactions are dynamic in time. Traditional models can't express this kind of complexity of population flow but we can extract the spatial-temporal features of it from data using deep learning model. What's more, the mapping from population variation to population flow is regionally diverse. So, every region should have its own model to extract the distinctive pattern.

Due to the reasons above, we propose to solve the population flow generation problem with deep learning. Deep learning models have been widely used in many applications and given excellent performance, especially CNN (convolutional neural network)[14]. CNN performs well on extracting the interactions between regions from data. And CNN with multiple layers is good at hierarchically extracting spatial features. Deep layers in the CNN can get the long-range spatial dependencies. Additionally, Zhang et al. [16, 23, 24]

bring the knowledge that CNN carries the ability of spatial-temporal prediction. The diversity of interregional population flow makes it hard to extract different patterns from a single model. So, we propose to use multiple deep learning models to generate the respective region's population flow with all others.

In this paper, we propose a deep learning based model to generate population flow from aggregated population variation data. First, we use CNN to extract the interactions between regions from data. Second, we use multiple models to generate the population flow of the corresponding region respectively. Our contributions can be summarized as follows:

- We design a deep learning based model to generate population flow from aggregated population variation data.
- We conduct extensive experiments with several baselines. Compared with the baselines, results demonstrate that our model has considerable advantages in generating population flow.

The rest of this paper is organized as follows. We first formulate the problem in Section 2. Following the problem formulation, in Section 3, we describe the motivation of our works and the details of the whole framework of our model. In Section 4, we apply our model on real-world mobility datasets and conduct extensive experiments. After systematically reviewing the related works in Section 5, we finally conclude our paper in Section 6.

## 2 PROBLEM DEFINITION

The several definitions of generating population flow from aggregated population variation data will be introduced as follow.

**Definition 1 (Region)** Based on longitude and latitude, the city is divided into (H x W) grid regions $\mathbb{S}$.

**Definition 2 (Population Variation)** Population variation is the aggregated data counting all people flow into or out of one region and we define them as inflow and outflow, respectively. The inflow/outflow of region $(h, w)$ at $t^th$ time are expressed as follow,

$$x_k^{h,w,in} = \sum_{T_{r_k} \in \mathbb{P}} |\{j > 1 | g_{j-1} \notin (h,w) \ \& \ g_j \in (h,w)\}|,$$

$$x_k^{h,w,out} = \sum_{T_{r_k} \in \mathbb{P}} |\{j \geq 1 | g_{j-1} \in (h,w) \ \& \ g_j \notin (h,w)\}|.$$

The $\mathbb{P}$ up here means trajectories collection at the $k^{th}$ time interval. $T_r : g_1 \rightarrow g_2 \rightarrow \cdots g_{|T_r|}$ is a trajectory in $\mathbb{P}$, and $g_j$ represents the geospatial coordinate; $g_j \in (h, w)$ when $g_j$ belong to region $(h, w)$, while $g_j \notin (h, w)$ not. $|\cdot|$ means the cardinality of a set.

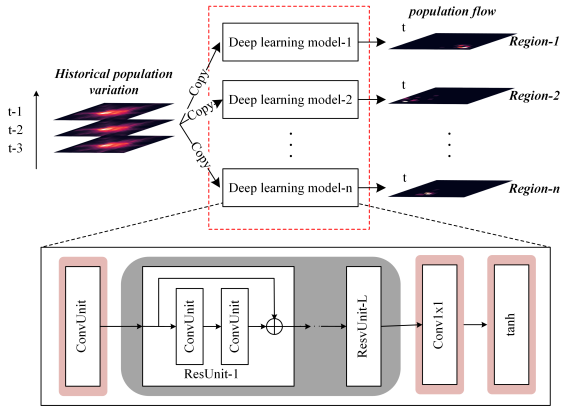



**Figure 1: Main architecture.**

**Definition 3 (Population Flow)** Population flow provides the information of how many people move from one region to another. We named one region $s$, $s \in \mathbb{S}$. Population flow is defined as follow,

$$f_k^{s_m, s_n} = \sum_{T_{r_k} \in \mathbb{P}} |\{j > 1 | g_{j-1} \in s_m \ \& \ g_j \in s_n \ \& \ m \neq n\}|.$$

Here $f_k^{s_m, s_n}$ means the population flow from region $s_m$ to region $s_n$ at $k^{th}$ timestamps.

**Population Flow Generation** We generate $\{f_k\}$ with given historical population variation data $\{x_t | t < k\}$, where $\{f_k\}$ means the population flow set between regions at time $k$.

## 3 METHODS

### Motivations

The difficulties of the population flow generation from aggregated data lie in that interactions between a region with all others are a lot in quantity and variety. And the interactions are dynamic in time. This means the complexity of population flow makes traditional models hardly model the spatial-temporal features. Even worse, the patterns of the flow between different regions are various. A single model can't express this kind of diversity.

Deep learning models [14] have shown its great power to extract the interactions between regions from data, and [16, 23, 24] have shown that CNN carries the ability of spatial-temporal prediction.

The output data of population flow contains important information. Because of the different locations of regions and the uniqueness of the mode of transportation in each region, the population flow of each region with all others has a distinctive pattern. Neural networks are powerful in learning this kind of pattern from the output data. In this way, the generation is not based entirely on input population

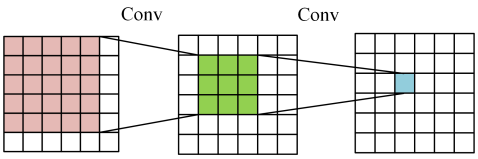



**Figure 2: Convolutions for capturing different level dependencies.**

variation, and transition patterns in population flow data are also playing a role in influencing the model.

### Model

Our model's framework is shown in Figure 1. It takes stacked historical aggregated population variations of all regions $\{X_i \in \mathbb{R}^{2\times H\times W} \mid i = k-l, ..., k-2, k-1\}$ as input, where $l$ is the length of dependent timestamps of historical data. Here is the formulation of historical population map,

$$(X_i)_{0,h,w} = x_i^{h,w,in},$$

$$(X_i)_{1,h,w} = x_i^{h,w,out}.$$

The $l$ historical population maps are stacked onto different channels of one training instance and Min-Max normalized to $[-1, 1]$. Then, the instance input the network and the output is population flow of one region.

The flows of one selected region $(h^*, w^*)$ $\{\hat{F}_k \in \mathbb{R}^{2\times H\times W}\}$ include population of all other regions flowing into this region, and the number in the opposite direction. The formulation is as follow,

$$(\hat{F}_i)_{0,h,w} = f_i^{(h,w),(h^*,w^*)},$$

$$(\hat{F}_i)_{1,h,w} = f_i^{(h^*,w^*),(h,w)}.$$

Here we use convolutional operators to extract hierarchical spatial features, shown as Figure 2. People on the subway will have a high speed and move a far distance in a short time interval. Hence, the long-range spatial dependencies is important in population flow generation. And CNN should have multiple convolutional layers to get that information. The architecture of ConvUnit in Figure 1 includes a convolutional layer and a batch normalization layer[10] followed by Relu function[6]. The formulation of ConvUnit is shown as below,

$$X^{(l)} = F_{Relu}(F_{BN}(W^{(l)} * X^{(l-1)} + b^l)),$$

where $X^{(l)}$ means the feature map of layer $l$, $F_{Relu}$ means the Relu activation function, $F_{BN}$ means the batch normalization operation, $W^{(l)}$ and $b^l$ represents weights and bias of the convolutional layer.

The ConvUnit before ResUnits is used to adjust the number of tensors' channels to matches with ResUnits at first, this step will also extract a certain degree of features. Batch normalization layers working here have a great effect. Population flow data across a wide range of areas is often sparse. Batch normalization layers can eliminate the impact of data particularity and prevent our model converging at a local optimum.

It's well known that very deep convolutional networks will compromise its training effectiveness. But we need many consecutive convolutional layers to extract the long-range dependencies. ResNet[8] is famous for that it will remain its performance even if the architecture goes very deep. So we use ResUnits to extract the spatial-temporal features. The formulation of ResUnit is as followed,

$$X^{l+1} = X^l + \mathcal{F}(X^l),$$

where $\mathcal{F}(\cdot)$ represents the function of calculation that passes two successive ConvUnits, and $X^l$ represents the feature map of $l^{th}$ layer. ResUnit can be regarded as a feature extractor unit with a shortcut connection. We use identity mapping as the shortcut, and its output is element-wise added to the output of 2 stacked ConvUnit layers, shwon in Figure 1. The output of ResUnits then passes through a $(1 \times 1)$ convolutional operator and is reshaped to $(2\times H\times W)$. We use $tanh$ function[15] to limit the output into $[-1, 1]$ which can be re-normalized to the level of population number.

**Training** Algorithm 1 tells the process of training our model. We construct training instances, $D_k$, from the historical population variation data maps $\{X_i | k < i\}$, shown in lines 1-5, where $k$ means the timestamps of the target population flow generation. We train the model by back-propagation and using Adam[13] as the optimizer.

**Algorithm 1** Model Training Algorithm

**Input:**
Dependent $l$ historical population variation data of timestamp $k$, $\{X_i | i = k-1, k-2, ..., k-l\}$
**Output:**
Generated population flows $\{\hat{F}_k\}$ at timestamp $k$ of selected region $(h^*, w^*)$.
//construct training set
1: $\mathbb{D} \Leftarrow \emptyset$
2: **for** all timestamps of training set **do**
3: $D_i \Leftarrow [X_{i-1}, X_{i-2}, ..., X_{i-l}]$
4: put $D_i$ into $\mathbb{D}$
5: **end for**//$X_i$ is the variation instance at time $i$.
//train model
6: initialize the learnable parameters $\theta$ of model
7: **repeat**
8: choose a batch of $D$ from $\mathbb{D}$
9: optimize $\theta$ by Adam based on the choosen batch of $D$
10: **until** model converge

## 4 EXPERIMENTS

In this section, we conduct extensive experiments on a real-world dataset to demonstrate the superiority of our model.

### Dataset

This dataset is collected from the most popular social network in China from Apr. 1st to Apr. 30th located in Beijing. It records the location of users when they used the location service in the application. We first partition the grids as **Definition 1**. And then we extract population variation as **Definition 2**, and extract population flow as **Definition 3**. We choose data from the last week as the testing data, and all data left as the the training data.

### Baselines

- **HA**: It utilizes the periodical historical average population flow value to generate the respectively period population flow.
- **Gravity Model**[3]: Gravity model is a well known and widely used classical model that motivated by Newton's law of Gravitation. In population flow generation, the population flow is represented by the gravity produced between celestial bodies and the population number represents the mass of celestial bodies. It includes two versions: singly constrained and globally constrained. We test them all on our data.
- **Individual Model**: It computes the aggregated regional individual's probability distribution of flowing into all other regions based on historical data. And the probability is then used to generate population flow by aggregated population variation at target timestamps on individual level.

### Metrics

We select Root Mean Squared Error (RMSE) and Normalized Root Mean Squared Error (NRMSE) and Mean Absolute Error (MAE) as metrics,

$$RMSE = \sqrt{\frac{1}{T}\sum_{i=1}^{T}||\mathbf{X}_i - \hat{\mathbf{X}}_i||_2^2},$$

$$NRMSE = \frac{RMSE}{\sqrt{\frac{\sum_{i=1}^{T}||\mathbf{X}_i - \overline{\mathbf{X}}||_2^2}{T}}},$$

$$MAE = \frac{1}{T}\sum_{i=1}^{T}|\mathbf{X}_i - \hat{\mathbf{X}}_i|,$$

where $\mathbf{X}_i$ and $\hat{\mathbf{X}}_i$ means the ground-truth and the generation at the $i^th$ time interval.

### Performance

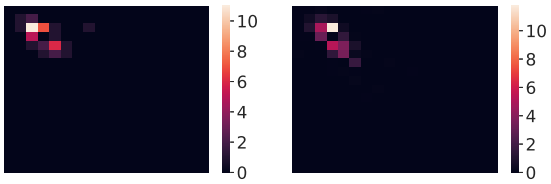
**Overall Performance** The performances of our model and baselines are shown in Table 1. The parameters of our model are given in Table 2. The ResNum means the number of ResUnits in the model. The len_closeness means length of historical data of one training instance. We use $(7 \times 7)$ kernels which have 64 channels as the convolutional operator in ConvUnits. As shown in Table 1, deep learning has considerable advantages in solving the problem of generating population flow from aggregated data. Even though the architecture of CNN in Table 1 includes a ConvUnit of $(1 \times 1)$ and 3 ConvUnits of $(7 \times 7)$ only, it exceeds the best traditional baseline, individual model, 10 percent in performance based on NRMSE. And we test the model which has 6 ResUnits, as the ResNet shown in Table 1. ResNet with 6 ResUnits exceeds the CNN by 5 percent based on NRMSE. From the view of MAE, basic CNN doesn't perform well enough. Because the high deviation generated instances have been destroyed but not a large number of low deviation generated instances during training. This can be solved by increasing the number of layers of the neural network. We select one region and visualize its generated population flow and ground truth of one same time via heatmap, shown in Figure 3. Every cell in the heatmap represents a grid region in city. And the region we select is at $(3, 3)$ on shown heatmap of Figure 3, whose top-left cell's coordinate is $(0, 0)$. We can see that the population flow generated is very close to the ground truth.

| Model | RMSE | NRMSE | MAE |
|---|---|---|---|
| gravity-globally | 4.0876 | 1.0042 | 0.3890 |
| gravity-singly | 3.6046 | 1.0049 | 0.3315 |
| HA | 1.6172 | 0.4018 | 0.1762 |
| Individual-Model | 1.2893 | 0.3204 | 0.1590 |
| CNN | 1.1649 | 0.2880 | 0.1823 |
| ResNet | 1.1028 | 0.2726 | 0.1522 |

**Table 1: Comparison with baselines**

| | Value |
|---|---|
| ResNum | 6 |
| len_closeness | 3 |
| kernel_size | 7 |
| channelsNum | 64 |

**Table 2: Parameters**

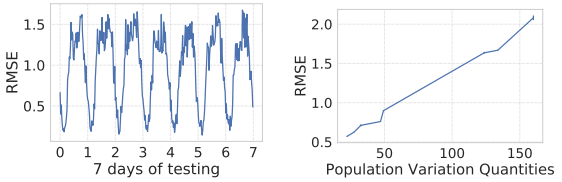



(a) generation (b) groundtruth

**Figure 3: Comparison visualization of generated population flow with groundtruth.**

**Results Analysis** We analyze the generated results of our deep residual model from spatial and temporal perspectives. As shown in Figure 4(a), the fluctuations in the error during the testing week showed a significant periodicity. During the peak traffic period (daytime and midnight before), the generation error will increase significantly. This shows that the greater the change of population variation, the more difficult it is to generate flow. The similar results are shown in Figure 4(b). Regions with larger population changes(higher population variation) has a larger error.

**Effect of Parameters** There are four important parameters, shown in Table 2. The effects of these parameters are shown in Figure 5. Figure 5(a) shows the effect of length of historical data. The results give knowledge that the information isn't enough to generate population flow if the historical length is too short. Too much noise brings out if it's too long. The same as kernel size, moderate kernels give better performance, as shown in Figure 5(b). The population flow depends on not only the temporal patterns but also the spatial patterns. The $(1 \times 1)$ kernel can't extract the interaction of the regions. During the forward propagation, we use zero paddings to fit the size of intermediate feature maps. If the kernel size is too large, there will be too many zeros adding to the feature maps as noise which will make our model perform worse. Figure 5(c) give the experiment results of the survey on the number of ResUnits used in model. We can see that the deeper the network is, the better performance will the model brings. The element in a feature map of the deeper layer will get information from a larger scale compared with the shallow one, shown in Figure 2. That's maybe the reason for better performance of the deeper network. Figure 5(d) shows the relationship between performance and the number of channels of ConvUnits used in ResUnits. In a certain range, increasing the complexity of the model by add channels will improve the performance. However, too many channels will make the network overfitting. That's what we can see from Figure 5(d).

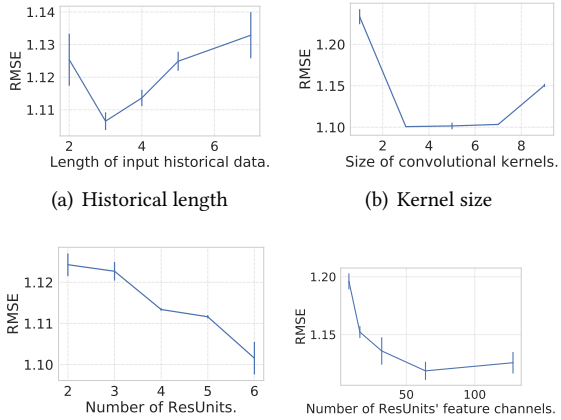



(a) Analysis from temporal view. (b) Analysis from spatial view.

**Figure 4: Results analysis from spatial and temporal view.**

## 5 RELATED WORK

The work in the following three fields is related to our work.




(a) Historical length (b) Kernel size

(c) ResUnits (d) Channels

**Figure 5: Effect of importance parameters.**

**Population Variation Prediction.** There are some published studies working on prediction crowd flow from historical data. Some used traditional methods. Xia et al. [20] usde KNN (K-nearest neighbor algorithm) to improve forecasting accuracy based on spatial-temporal correlation. Fan et al. [5] used random Markov chains to model the naive movement. Hoang et al. [9] proposed the seasonal and trend models based on Gaussian Markov random fields. Many deep learning-based models have been proposed as the development of neural networks. ConvLSTM[21], hybrid deep learning framework[4], STRCNs[12], Periodic-CRN[26], were all in form of combining CNN with RNN. Deep-ST[24] is the first model to extract spatial features based on CNN. Further, ST-ResNet[23] replaced the general convolutional operation with the residual framework. DeepST+[16] used ConvPlus, SemanticPlus operation to get long-range spatial dependence and POI (point of interest) information and improved the model. Works above are all CNN-based. They all predict aggregated crowd flow.

**Origin-Destination Prediction.** Calabrese et al. [2] probed trajectory from location data first, then recovered the flow data from trajectories. Gong et al. [7] predicted the next needed OD matrix from previous OD snapshots based on Online Non-negative Matrix Factorization. Toqué et al. [1, 19, 25] predicted time series of TM (traffic matrix ) by LSTM from historical OD data.

**Population Flow Generation from Aggregated Data.** Some works focused on population flow generation from aggregated flow data. Xu et al. [22] recovered trajectory from aggregated population data and transition probabilities. Iwata et al. [11] used collective graphical models as a framework to estimate the population flow spatial-temporal population data. Tanaka et al. [17] estimated latent population flow from inflow and outflow. Iwata et al. [18] was proposed by adding the elements of a simple neural network

to get the probability distribution used to generate population flow based on [11].

## 6 CONCLUSION

Population flow data have a great value but it's difficult to get the data. In this paper, we propose a deep learning based model to generate detailed population flow given aggregated population variation data. With our model, the patterns that exist in population flow data can be learned by the neural network which is of great use in generation. Compared with several baselines, the extensive experiments confirm the superiority of our model.

What is more, we have a more concise and effective research direction. The first is merging the POI and extra support information into our model to enhance its performance. The second is using transfer learning to get similarity between regions to accelerate the training process.

## 7 ACKNOWLEDGMENTS

This work was supported in part by The National Key Research and Development Program of China under grant 2017YFE0112300, the National Nature Science Foundation of China under 61861136003, 61621091 and 61673237, Beijing National Research Center for Information Science and Technology under 20031887521, and research fund of Tsinghua University - Tencent Joint Laboratory for Internet Innovation Technology.

## REFERENCES

[1] Abdelhadi Azzouni and Guy Pujolle. 2017. A Long Short-Term Memory Recurrent Neural Network Framework for Network Traffic Matrix Prediction. (2017).

[2] Francesco Calabrese, Giusy Di Lorenzo, Liang Liu, and Carlo Ratti. 2011. Estimating Origin-Destination flows using opportunistically collected mobile phone location data from one million users in Boston Metropolitan Area. (2011).

[3] Henry Charles Carey. 1867. *Principles of social science*. Vol. 3. JB Lippincott & Company.

[4] Shengdong Du, Tianrui Li, Xun Gong, Yan Yang, and Shi Jinn Horng. 2017. Traffic flow forecasting based on hybrid deep learning framework. In *2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*. IEEE, 1–6.

[5] Zipei Fan, Xuan Song, Ryosuke Shibasaki, and Ryutaro Adachi. 2015. CityMomentum: an online approach for crowd behavior prediction at a citywide level. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 559–569.

[6] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 315–323.

[7] Yongshun Gong, Zhibin Li, Jian Zhang, Wei Liu, Yu Zheng, and Christina Kirsch. 2018. Network-wide Crowd Flow Prediction of Sydney Trains via customized Online Non-negative Matrix Factorization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1243–1252.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[9] Minh X Hoang, Yu Zheng, and Ambuj K Singh. 2016. Forecasting citywide crowd flows based on big data. *ACM SIGSPATIAL* 2016 (2016).

[10] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[11] Tomoharu Iwata, Hitoshi Shimizu, Futoshi Naya, and Naonori Ueda. 2017. Estimating people flow from spatiotemporal population data via collective graphical mixture models. *ACM Transactions on Spatial Algorithms and Systems (TSAS)* 3, 1 (2017), 2.

[12] Wenwei Jin, Youfang Lin, Zhihao Wu, and Huaiyu Wan. 2018. Spatio-Temporal Recurrent Convolutional Networks for Citywide Short-term Crowd Flows Prediction. In *Proceedings of the 2nd International Conference on Compute and Data Analysis*. ACM, 28–35.

[13] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[15] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. 2012. Efficient backprop. In *Neural networks: Tricks of the trade*. Springer, 9–48.

[16] Ziqian Lin, Jie Feng, Ziyang Lu, Yong Li, and Depeng Jin. 2019. DeepSTN+: Context-aware Spatial-Temporal Neural Network for Crowd Flow Prediction in Metropolis. AAAI.

[17] Yusuke Tanaka, Tomoharu Iwata, Takeshi Kurashima, Hiroyuki Toda, and Naonori Ueda. 2018. Estimating Latent People Flow without Tracking Individuals.. In *IJCAI*. 3556–3563.

[18] Iwata Tomoharu and Shimizu Hitoshi. 2019. Neural Collective Graphical Models for Estimating Spatio-temporal Population Flow from Aggregated Data. AAAI.

[19] Florian Toqué, Mohamed Khalil El Mahrsi, Etienne Côme, and Latifa Oukhellou. 2016. Forecasting Dynamic Public Transport Origin-Destination Matrices with Long-Short Term Memory Recurrent Neural Networks. In *IEEE International Conference on Intelligent Transportation Systems*.

[20] Dawen Xia, Binfeng Wang, Huaqing Li, Yantao Li, and Zili Zhang. 2016. A distributed spatial–temporal weighted model on MapReduce for short-term traffic flow forecasting. *Neurocomputing* 179 (2016), 246–263.

[21] Shi Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*. 802–810.

[22] Fengli Xu, Zhen Tu, Yong Li, Pengyu Zhang, Xiaoming Fu, and Depeng Jin. 2017. Trajectory recovery from ash: User privacy is not preserved in aggregated mobility data. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1241–1250.

[23] Junbo Zhang, Zheng Yu, and Dekang Qi. 2016. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. (2016).

[24] Junbo Zhang, Zheng Yu, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. 2016. DNN-based prediction model for spatio-temporal data. In *Acm Sigspatial International Conference on Advances in Geographic Information Systems*.

[25] Jianlong Zhao, Hua Qu, Jihong Zhao, and Dingchao Jiang. 2018. Towards traffic matrix prediction with LSTM recurrent neural networks. *Electronics Letters* 54, 9 (2018), 566–568.

[26] Ali Zonoozi, Jung-jae Kim, Xiao-Li Li, and Gao Cong. 2018. Periodic-CRN: A Convolutional Recurrent Model for Crowd Density Prediction with Recurring Periodic Patterns.. In *IJCAI*. 3732–3738.