

Received October 1, 2019, accepted October 18, 2019, date of publication November 1, 2019, date of current version December 17, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2950956

ASTIR: Spatio-Temporal Data Mining for Crowd Flow Prediction

LABLACK MOURAD, HENG QI^{ID}, YANMING SHEN^{ID}, AND BAOCAI YIN

School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China

Corresponding author: Yanming Shen (shen@dlut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant U1811463 and in part by the Innovation Foundation of Science and Technology of Dalian under Grant 2018J11CY010 and Grant 2019J12GX037.

ABSTRACT The citywide crowd flow prediction is crucial for a city to ensure productivity, safety and management of its citizen. However, the crowd flow may be affected by many factors, such as weather, working times, events, seasons, and so on. In this paper, we proposed Attentive Spatio-Temporal Inception ResNet (ASTIR), which aims to address the difficulty of crowd flow prediction. The ASTIR is based on the Inception-ResNet structure combined with Convolution-LSTM layers and attention module to better capture pattern movement changes. We build our deep neural network framework consisting of four distinct parts, by which we can capture the short-term, long-term and period properties, as well as external factors that can affect crowd flow behaviors. To show the performance of the proposed method, we use the widely applied benchmarks for crowd flow prediction (Taxi Beijing and Bike New York), and obtain notable improvements over the state-of-the-art approaches.

INDEX TERMS Deep learning, crowd flow prediction, spatio-temporal modelling.

I. INTRODUCTION

Nowadays almost every person is connected via intelligent devices (smartphones, cars, smartwatches and so on), and those devices have led to an enormous amount of data (personal information, health information, GPS position, etc.) that is broadcasted in realtime to servers for the purpose of analysis. The crowd flow analysis is one of it. It is the interpretation of devices signals and GPS position data changes over time, i.e., analyzing the movements of individuals. This analysis will be used to make crowd flow predictions. This field has recently attracted many researchers because of its importance for traffic management and public safety. A better understanding of how crowds move and what factors are involved in patterns movement changes can lead to better infrastructure architecting and preventing crowd catastrophizes.

Many solutions have been proposed for crowd flow predictions. It first started with classical approaches like ARMA and VAR, and then the deep neural networks approaches. However, the accuracy of a deep neural network approach relies heavily on its architecture and hyper-parameters. The emergence of attentions in image

The associate editor coordinating the review of this manuscript and approving it for publication was Tie Qiu^{ID}.

segmentation and machine translation helped further decrease the error margin. We believe that such an approach can be implemented and adapted to achieve the state-of-the-art predictions.

In this paper, we focus on a better way to architect a deep neural network model for citywide crowd flow prediction. Our data is presented as positions (longitude and latitude) of pedestrians in a city over time. We first preprocess the data into two maps at each timestamp, inflow and outflow, where the inflow represents the flow entering an area and the outflow represents the flow leaving an area. Each area is represented as a square, e.g., a 1×1 (Km) in the real world. We then construct a 4D tensor (Timesteps, 2-Channels, Height, width), where the 2-channels represent the inflow and the outflow. This tensor matches the structure that a convolution-LSTM layer can process.

The crowd flow prediction is quite complex and is influenced by many factors, such as weather, holidays, economy, and so on. We group them into 4 categories:

1. Spatial dependency: the crowd is related by the nearby areas, what happened in a region may end up causing changes in the neighboring districts.
2. Temporal dependency: the crowd flow is also affected by what happened in the nearby time. For example, an accident at 11am can affect the traffic at 12am.

3. Periodical dependency: the flow is also affected by the periodical patterns. For example, on the workdays the rush time is caused by working time.
4. Externals factors: other factors such as weather, season of the year, holidays may also have an effect on the crowd flow.

Furthermore, the inflow of a zone is also affected by outflows of other regions and vice-versa.

To address the above challenges, we introduce ASTIR module that is capable to accurately capture these dynamics. Our main contributions can be summarized as follows:

- We architect ASTIR in a way that it can take into consideration three main properties related to crowd flow data (short-term, long-term, and periodical dependencies). ASTIR uses three modules to capture those characteristics.
- ASTIR applies Convolution-LSTM layers assembled in inception-resnet-v2 way [1], which allows us to have different sizes of filters to capture more accurately the spatio-temporal aspect of crowd flow dataset.
- We adapt ‘Squeeze and Excitation module’ [2] to support our spatio-temporal inception-resnet module, where the inflow and outflow are considered as channels and short-term, period, long-term lengths are considered as timesteps.
- ASTIR is extensively evaluated on two popular datasets (Taxi Beijing and Bike NYC) and outperforms the state-of-the-art frameworks by a significant margin.

II. RELATED WORK

The city-wide crowd flow prediction is a highly complex task that has attracted many researchers. The complexity of this task starts from the acquisition of data. The data arrives as a bunch of GPS coordinates and phone signals which cannot be directly analyzed, and a preprocessing is necessary. One of the earliest works was presented in [3], where they proposed a novel way to transform data into a graph that can be processed easily. Each node of the graph represents the recording time at t and the link between nodes represents the distance traveled between two-time intervals (t and $t + 1$). The work in [4] followed by proposing a slightly different approach, where a node represents an area or a station and the link between two nodes is the road. This approach helped to represent the city in a more static way and led to more accurate prediction. In our work we choose a grid map representation of the city crowd flow that was introduced by [5], and this representation was used in many researches [5]–[9] and will be discussed in Section 3.A. [5], [6] introduced a temporal separation (Short-term, Period and Long-term) that was very crucial to our work. In summary, the goal of these approaches is to leverage the temporal information and capture all the periodic tendency that may affect the crowd. We will discuss in more details the temporal separation in Section 3.B.

The crowd flow prediction problem can be solved by deep learning method, outperforming the classical methods like ARAMA or VAR. A deep convolution neural network

was introduced in [5]. Then, convolution layer was applied in different architectures including auto-encoder. However, a pure CNN based model was hard to scale up. The ResNet module is proposed for deeper network [6], where it was first used in image recognition [10] and offered a better way to train a very deep network. [11] introduced the inception module that showed improvement over ResNet. The Inception-ResNet, a combination of Inception module and ResNet network was proposed in [1], a natural evolution to Inception with a higher accuracy and a more efficient training process. Following this line, we have chosen to use the Inception-ResNet-v2 as a base structure for our model.

The downside of using convolution layer to process crowd flow data is that it lacks the capability to capture temporal aspect of the data. On the other side, RNN models are good for temporal data, and many attempts have been done [9], [12]–[14]. A convolutional recurrent auto-encoder [15] showed that using both convolution and recurrent at the same time gives a better result. Taking in consideration these improvements, we opted for a Convolution-LSTM layer [16]. The studies on attention mechanism [17] showed that for time series data, a simple fully connected network using attention can outperform recurrent neural networks. LSTM incorporating attention is very successful and gives more accurate prediction [9]. The attention was also used in the channel selection and helped recalibrate the feature responses towards the most informative and important components of the input, and it has also the role to investigate the interdependencies the channels of the convolutional features in a network. In our work, we not only focused on the relation between inflow and outflow, but also between timesteps that may be more relevant than others. Therefore, we propose a channel attention mechanism, which uses an idea similar to [2]. Our proposed attention can let the network construct an informative feature by fusing both the special and the channel-wise information.

III. PRELIMINARIES

In this section, we define some basics necessary to our study and describe the externals that may influence the crowd flow prediction.

A. CROWD FLOW PREDICTION

The crowd flow prediction problem can be defined as $M_d^{t+1} = f(M_d^t)$, where M_d^t is the map at time t for day d and M_d^{t+1} is the map at time $t + 1$ for the same day, and f is the predicting function (in our case ASTIR).

The M_d^t is a two channels grid map that is obtained from GPS positions or phone signals. The two channels are inflow and outflow. The inflow and outflow are defined as follows:

$$M_t^{in,h,w} = \sum_{Traj} |\{S > 1 | position_{t-1} \notin (h, w) \text{ and } position_t \in (h, w)\}|$$

$$M_t^{out,h,w} = \sum_{Traj} |\{S \geq 1 | position_t \in (h, w) \text{ and } position_{t+1} \notin (h, w)\}|$$

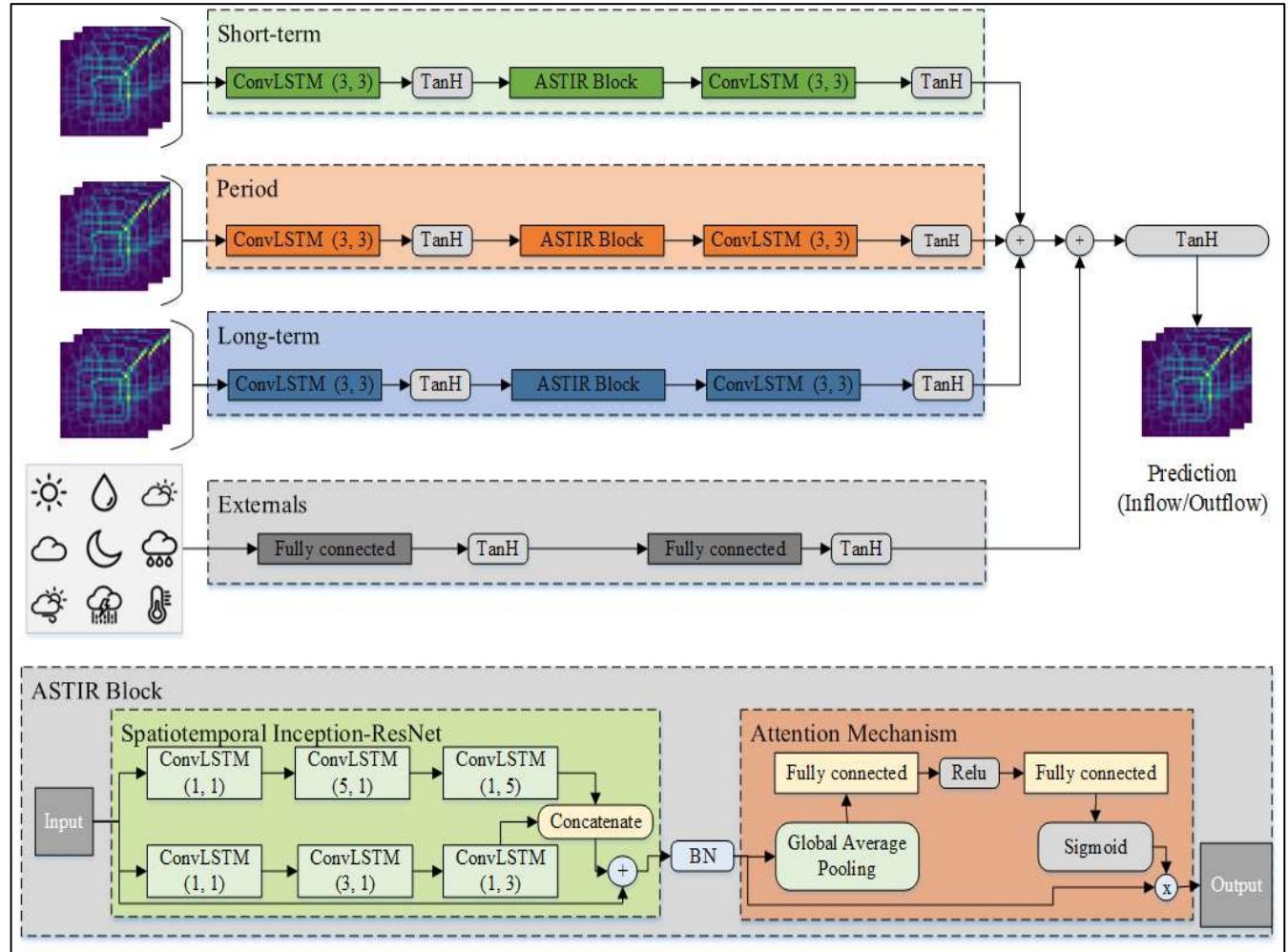


FIGURE 1. ASTIR architecture.

where Traj represents the trajectory of a pedestrian and $M_d^{in,h,w}, M_d^{out,h,w}$ are a specific area (h,w) in the map (inflow channel and outflow channel, respectively).

B. TEMPORAL SEPARATION

Following [6], we have chosen to separate our data into three categories: short-term, period and long-term.

$$\text{Short-term: } S = [X_{t-l_s}, X_{t-(l_s-1)}, X_{t-1}]$$

$$\text{Period: } P = [X_{t-l_p,p}, X_{t-(l_p-1),p}, X_{t-p}]$$

$$\text{Long-term: } L = [X_{t-l_q,q}, X_{t-(l_q-1),q}, X_{t-q}],$$

where l_s , l_p , and l_q are the corresponding parameters. As an example, if the smallest timestep is 1 hour, the short-term will be 1-hour length, the period will be 1-day, and the long-term will be 1-week.

C. EXTERNAL

People behave differently on a bad weather. For instance, on cold and rainy holiday, people have the tendency to stay at home or indoor public spaces, which translates into

less traffic. The externals are heterogeneous data and cannot be fed to a network directly, and a pre-processing is needed. We first separate the categorical attributes together and then apply a one-hot encoding. For the non-categorical attributes, we simply apply a min-max normalization to ensure that all the values are on the same scale. The holidays are presented as a date type, and we use those dates to construct a vector with the same size as the dataset, where the value of each element of the vector will be as follows:

- 0: if it is a working day
- 1: if it is a holiday

Finally, we concatenate all the externals to form a vector that contains all the externals attribute and feed it into a special branch of our network responsible for extracting all the external features and match them with each area of the map.

IV. ASTIR

Figure 1. shows the architecture of ASTIR. There are three main components that take M_d^{t-1} as input maps and output

three predictions, short-term, period and long-term respectively. The three predictions are then merged together to form $M_{d,tmp}^t$. If external data is available, then they are fed to the fourth component that will output N numbers of predictive externals information E_d^t , where N represents the number of areas in each map ($N = M_{Height} * M_{Width} * M_{Channel}$). We then reshape it to match $M_{d,tmp}^t$. E_d^t and $M_{d,tmp}^t$ are merged together to construct a precise map prediction based on externals and spatio-temporal information, which is fed into a tanh activation layer.

A. THE SPATIO-TEMPORAL COMPONENTS

The first three components are structured in the same way. They consist of three main parts. The first part is a convolution-LSTM layer that will extract the feature maps to feed the second component. The second component is the main component, which is the ASTIR block composed of our spatio-temporal Inception-ResNet and attention mechanism, and this block can be duplicated as needed (for more complex data that needs a deeper network). The output of the ASTIR block is fed to the third component, another convolution-LSTM layer that will output the desired prediction.

B. THE ASTIR BLOC

The ASTIR block is a two-stage block, where the first stage is responsible for learning the spatial and temporal dependency of the crowd flow data, and the second stage is responsible for learning the attention weight that helps focus on remembering the relations between not only the inflow and outflow, but also the most important temporal sequence feature to the prediction.

For the first sub-block (Spatio-temporal inception resnet), we use a Conv-LSTM [16] as a base layer. Since our data is both spatial and temporal, the Conv-LSTM layer is capable to catch the temporal dependency in our data. However, the matrix multiplication necessary for carrying the temporal dependency is too heavy when working with a 4D inputs (timesteps, channels, height, weight). Using convolution operations in the input-to-state and state-to-state transitions not only helps reduce the heavy workload, but also captures the underlying spatial features by convolution operations in multiple-dimensional data. Note that the timesteps in our case are defined by the short-term, long-term and period length. Also, using a Conv-LSTM layer instead of stacking convolution layers followed by LSTM layers not only helps performance, but also give us flexibility in designing our block to be reusable by avoiding the reshapes necessary for the stack.

Since our data is quite sparse, choosing the right kernel size for the convolution operation is difficult. Therefore, it is preferred to have a wide module that contains in parallel different sizes of kernels, where large kernel for far neighbors and small kernel for direct neighbors. Therefore, inspired by [6], we use three branches. The first branch contains three Conv-LSTM layers, where the first layer contains a kernel

(1, 1) that is mostly used to downgrade or limit the number of channels for performance purpose. The second and the third Conv-LSTM layer are basically a factorization of a Conv-LSTM layer with a (3, 3) kernel, which calculates the interaction between a specific area with nearby neighbors. The second branch contains the same set of Conv-LSTM layer except that we factorize a kernel (5, 5) which represents the bigger kernel responsible for far neighbors' influence. The output is concatenated with the output of the first branch and then fed to a final Conv-LSTM layer. The last branch is used to carry the residual information, which applies an element wise addition between the residual information and the output of the last Conv-LSTM to perform the ResNet system.

Before going to the next step, we apply a batch normalization to speed up the training, which helps achieve a better attention weights learning.

The second sub-block represents our attention module, which is inspired by [2]. In [2], the attention was used to output a ratio that indicates which channel is relevant in an image. In our model, instead of output a channel ratio, it outputs a set of ratios that indicate each timestep and channel. For example, if we are on the branch responsible for the short-term and the short-term length is “ n ”, knowing that our map contains two channels (inflow and outflow), the input to our attention will be in the form of [n , 2, height, weight] and the output set of ratios will be in the form of [n , 2]. This set of ratios represents how much each instance of the map is relevant to the final prediction.

To achieve this, we first perform a 2D global average pooling, and then use a fully connected layer to extract the feature representations of each channel/timestep of the map. Finally, we apply a ReLU activation to ensure that there is no negative number, and this representation is then fed to a second fully connected layer followed by a sigmoid activation which ensures that all the values are between 0 and 1. Finally, we perform an element wise multiplication with the initial input to apply the set of ratios to the map.

The part responsible of learning the externals features consists of two fully connected layers. The first layer extracts the feature representation of each external. This representation is then fed to the second fully connected layer, and the second layer will output $X_{d,ext}^t$, which indicates how much each area is affected by those externals (the number of factors match the number of areas in the map). In other words, if this factor is negative, then the traffic rate will be lower than predicted. Otherwise it will be higher. All the fully connected layers are followed by a tanh activation that ensures all the value are in $[-1, 1]$ range. Finally, we merge $X_{d,ext}^t$ with $M_{d,tmp}^t$ and apply the activation layer as follows:

$$M_d^t = \text{tanH} \left(X_{d,ext}^t \oplus M_{d,tmp}^t \right),$$

where M_d^t is the final prediction at timestep t and \oplus is the element-wise addition.

V. EXPERIMENTS

A. DATASE

The preprocessing for TaxiBJ and BikeNYC is done similarly to [6] for comparison purpose, where all the GPS data is transformed to two maps (inflow and outflow map). Each map is separated into 1km*1km areas, represented as a pixel. The externals are processed as follows:

- Weather condition: one-hot encoding for 16 types (sunny, rainy, windy, etc.)
- Wind speed: real numbers [0, 48.6]
- Temperature: real numbers [-24.6, 41]
- Holidays: date type

A min-max normalization is performed on Wind speed and Temperature.

1) TAXIBJ

This dataset groups all GPS coordinates of taxies of Beijing, and it also contains all the meteorological data (wind speed, weather condition, temperature and holidays). The times recorded are: from 01/07/2013 to 30/10/2013, from 01/03/2014 to 30/06/2014, and from 01/11/2015 to 10/04/2016. We take the last four weeks as test and the rest is used for training, where 90% for training and 10% for validation.

2) BIKENYC

This dataset groups all the GPS coordinate of Bikes in New York City. The time interval is from 01/04/2014 to 30/09/2014. We take the last ten days as test and the rest are for training. The training is further split, 10% for validation and 90% for training.

B. BASELINES

- SARIMA: Seasonal Auto Regressive Integrated Moving Average.
- VAR: Vector Auto-Regressive.
- ARIMA: Auto Regressive Integrated Moving Average.
- ST-ANN: It extracts 8 nearby spatial values and 8 previous time steps for the temporal features to be fed into an artificial neural network.
- DeepST: DNN-based prediction model for spatio-temporal data, using convolutional layers [5].
- ST-ResNet: ResNet-based prediction model for spatio-temporal data, where the ResNet operation is performed [6].
- AFCM: Attentive-LSTM based model for spatio-temporal data [9].

We use RMSE (Root Mean Squared Error) to evaluate our model,

$$RMSE = \sqrt{\frac{1}{n} \sum_i (x_i - y_i)^2}$$

C. COMPARISON RESULT

We compare our model (ASTIR) to 7 different models that are currently used as a benchmark in the literature. Table 1 shows

TABLE 1. Comparison results for TaxiBJ and BikeNYC datasets.

Model	TaxiBJ	BikeNYC
SARIMA	26.88	10.56
VAR	22.88	9.92
ARIMA	22.78	10.07
ST-ANN	19.57	-
DeepST	18.18	7.43
ST-ResNet	16.69	6.33
AFCM	15.4	5.64
ASTIR	13.57	4.18

that ASTIR outperforms the best baseline (AFCM) by 11.88% in TaxiBJ and 25.89% in BikeNYC.

D. TUNING HYPERPARAMETERS

In this Section, we run experiments to approve the effectiveness of each component.

Short-term, long-term and period: The temporal separation is very crucial in our architecture, which is shown in Tables 2, 3, and 4. The results show that the lengths of each branch are very important.

TABLE 2. Short-term dimension length impact on TaxiBJ and BikeNYC datasets.

Short-term length	TaxiBJ	BikeNYC
0	24.41	16.7
1	16.07	5.85
2	13.8	5.19
3	13.57	5.16
4	13.61	5.07

TABLE 3. Period dimension length impact on TaxiBJ and BikeNYC datasets.

Period length	TaxiBJ	BikeNYC
0	14.32	12.76
1	13.57	5.09
2	13.78	4.65
3	14.09	4.62
4	14.3	4.59

The network performance is highly impacted by the hyperparameters.

1) EXTERNALS

Table 5 shows how externals can be useful to improve the accuracy, as explained in Section 3.C, the crowd flow is affected by these factors (wind speed, weather, etc.).

TABLE 4. Long-term dimension length impact on TaxiBJ and BikeNYC datasets.

Long-term length	TaxiBJ	BikeNYC
0	14.71	10.36
1	13.57	4.59
2	13.72	4.24
3	13.8	4.23
4	14.84	4.24

TABLE 5. Externals impact.

Externals	TaxiBJ
With	13.61
Without	16.89

TABLE 6. Number of layer performance.

Number ASTIR Blocks	TaxiBJ	BikeNYC
1	13.61	4.18
2	13.38	4.24
3	13.92	4.28

TABLE 7. Filter size performance.

Filter size	TaxiBJ	BikeNYC
16	14.16	4.18
32	13.7	4.37
64	13.61	4.43

2) NUMBER OF LAYERS

If the network is too deep or too shallow, it can lead to overfitting or under-fitting respectively. Finding the right number of layers is a necessity for a good accuracy. Table 6 shows how the depth of our model can affect the performance.

3) FILTERS SIZE

Finding the right filter size is important for high accuracy. Table 7 shows how the filter size affects the performance.

4) BATCH NORMALIZATION POSITIONING

We tried different position for the batch normalization layer. Table 8 shows how the position can have a big effect on performance.

TABLE 8. Batch normalization position impact.

BN position	TaxiBJ	BikeNYC
A	24.94	3.56
B	19.29	3.51
C	24.47	5.11
D	13.61	4.18

- A: The BN layer is processed after each merging layer, entrance layer, exit layer and finally the Spatio-Temporal Inception-ResNet block (STIR).
- B: The BN layer is processed after each merging layer, entrance layer and finally the STIR block.
- C: The BN layer is processed after each merging layer and the STIR block.
- D: The BN layer is processed only after each STIR block.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose ASTIR, a deep neural network model for predicting citywide crowd flow. ASTIR uses convolution-LSTM layer as main component arranged in an inception-resnet architecture. We also apply a time-channel attention mechanism to accurately predict the crowd flow changes. We have proved the efficacy of our architecture on two benchmarks (TaxiBJ and BikeNYC).

In future work, we may add an attention for the hidden-states of the convolution-LSTM layer which may help to remember more efficiently the changes that matter to the prediction and help the network focus more on areas that have a bigger impact on the nearby regions.

The code and the datasets are available at <https://github.com/Mouradost/ASTIR>.

REFERENCES

- [1] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” in *Proc. AAAI*, 2017, pp. 1–7.
- [2] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proc. CVPR*, 2018, pp. 7132–7141.
- [3] Y. Zheng, “Trajectory data mining: An overview,” *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 3, p. 29, May 2015.
- [4] J. Sun, J. Zhang, Q. Li, X. Yi, and Y. Zheng, “Predicting citywide crowd flows in irregular regions using multi-view graph convolutional networks,” Mar. 2019, *arXiv:1903.07789*. [Online]. Available: <https://arxiv.org/abs/1903.07789>
- [5] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, “DNN-based prediction model for spatio-temporal data,” in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, Burlingame, CA, USA, 2016, p. 92.
- [6] J. Zhang, Y. Zheng, and D. Qi, “Deep spatio-temporal residual networks for citywide crowd flows prediction,” in *Proc. AAAI*, 2017, pp. 1655–1661.
- [7] J. Ke, H. Zheng, H. Yang, and X. M. Chen, “Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach,” *J. Transp. Res. C, Emerg. Technol.*, vol. 85, pp. 591–608, Dec. 2017.
- [8] L. Wang, X. Geng, X. Ma, F. Liu, and Q. Yang, “Cross-city transfer learning for deep spatio-temporal prediction,” May 2018, *arXiv:1802.00386*. [Online]. Available: <https://arxiv.org/abs/1802.00386>
- [9] L. Liu, R. Zhang, J. Peng, G. Li, B. Du, and L. Lin, “Attentive crowd flow machines,” in *Proc. ACM Multimedia Conf. Multimedia Conf.*, 2018, pp. 1553–1561.
- [10] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, 2016, pp. 770–778.
- [11] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 2818–2826.
- [12] X. Cheng, R. Zhang, J. Zhou, and W. Xu, “Deeptransport: Learning spatial-temporal dependency for traffic condition forecasting,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Rio de Janeiro, Brazil, 2018, pp. 1–8.
- [13] X. Dai, R. Fu, Y. Lin, L. Li, and F.-Y. Wang, “DeepTrend: A deep hierarchical neural network for traffic flow prediction,” Jul. 2017, *arXiv:1707.03213*. [Online]. Available: <https://arxiv.org/abs/1707.03213>

- [14] J. Xu, R. Rahmatizadeh, L. Bölöni, and D. Turgut, "Real-time prediction of taxi demand using recurrent neural networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2572–2581, Aug. 2018.
- [15] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. ICLR*, 2018, pp. 1–16.
- [16] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. NeurIPS*, 2015, pp. 1–9.
- [17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NeurIPS*, 2017, pp. 1–11.



LABBLACK MOURAD was born in Oran, Algeria, in 1993. He received the B.S. degree in science and technology, and the M.S. degree in information and communication from Oran 1 Ahmed Ben Bella University, Oran, Algeria, in 2014 and 2016, respectively. He is currently pursuing the Ph.D. degree in computer software and theory with the Dalian University of Technology, Liaoning, China. He is currently with the development of predicting system for city wide crowd flow using neural networks. His current research interests include data mining and deep neural networks.



HENG QI received the B.S. degree from Hunan University, in 2004, and the M.E. and Ph.D. degrees from the Dalian University of Technology, in 2006 and 2012, respectively. He was a JSPS Oversea Research Fellow with the Graduate School of Information Science, Nagoya University, Japan, from 2016 to 2017. He is currently an Associate Professor with the School of Computer Science and Technology, Dalian University of Technology, China. His current research interests include computer networks and multimedia computing.



YANMING SHEN received the B.S. degree in automation from Tsinghua University, Beijing, China, in 2000, and the Ph.D. degree from the Department of Electrical and Computer Engineering, the Polytechnic Institute of New York University, Brooklyn, in 2007. He is currently a Professor with the School of Computer Science and Technology, Dalian University of Technology, China. His current research interests include recommender systems, video streaming, and algorithm design, analysis, and optimization. He was a recipient of the 2011 Best Paper Award for Multimedia Communications (awarded by the IEEE Communications Society).



BAOCAI YIN received the Ph.D. degree from the Dalian University of Technology, in 1993. He is currently a Professor with the Faculty of Electronic Information and Electrical Engineering, College of Computer Science and Technology, Dalian University of Technology. He is also a Researcher with the Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology. His current research interests include multimedia, multifunctional perception, virtual reality, and computer graphics. He is a member of the China Computer Federation.

• • •