Multi-task Adversarial Spatial-Temporal Networks for Crowd Flow Prediction

Senzhang Wang[1,2], Hao Miao[1], Hao Chen[3], Zhiqiu Huang[1,2]
[1]Nanjing University of Aeronautics and Astronautics, Nanjing, China
[2]Collaboration Innovation Center of Novel Software Technology and Industrialization, Nanjing, China
[3]Beihang University, Beijing, China
{szwang,miaohao,zqhuang}@nuaa.edu.cn,chh@buaa.edu.cn

ABSTRACT

Crowd flow prediction, which aims to predict the in-out flows (e.g. the traffic of crowds, taxis and bikes) of different areas of a city, is critically important to many real applications including public safety and intelligent transportation systems. The challenges of this problem lie in both the dynamic mobility patterns of crowds and the complex spatial-temporal correlations. Meanwhile, crowd flow is highly correlated to and affected by the Origin-Destination (OD) locations of the flow trajectories, which is largely ignored by existing works. In this paper, we study the novel problem of predicting the crowd flow and flow OD simultaneously, and propose a multi-task adversarial spatial-temporal network model entitled MT-ASTN to effectively address it. As a multi-task learning model, MT-ASTN adopts a shared-private framework which contains private spatial-temporal encoders, a shared spatial-temporal encoder, and decoders to learn the task-specific features and shared features. To effectively extract high quality shared features, a discriminative loss on task classification and an adversarial loss on shared feature extraction are incorporated to reduce information redundancy. We also design an attentive temporal queue to automatically capture the complex temporal dependency without the help of domain knowledge. Extensive evaluations are conducted over the bike and taxicab trip datasets in New York. The results demonstrate that our approach significantly outperforms state-of-the-art methods by a large margin on both tasks.

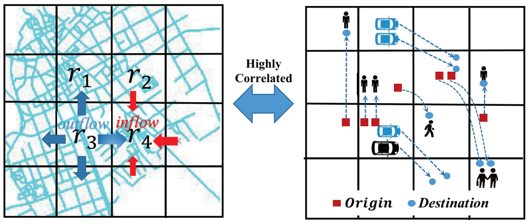
CCS CONCEPTS

• **Information systems** → **Spatial-temporal systems**.

KEYWORDS

Crowd flow prediction, multi-task learning, adversarial learning

ACM Reference Format:
Senzhang Wang, Hao Miao, Hao Chen, and Zhiqiu Huang. 2020. Multi-task Adversarial Spatial-Temporal Networks for Crowd Flow Prediction. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3340531.3412054



(a) Crowd flow (b) Flow OD

Figure 1: Illustration of crowd flow and flow OD

1 INTRODUCTION

As shown in Figure 1(a), crowd flows including in- and out- flows reflect the human mobility dynamics in different areas of a city. Such flows can be measured by various human mobility data such as taxi trajectories, sharing bike trips, subway check-in/out, pedestrians and all of them together if the data is available. Crowd flow prediction, which aims to build a fitting model to predict the sizes of crowd entering/leaving a particular region, is of great importance to various practical applications, and has attracted rising research interest recently [19]. For example, accurately forecasting the traffic flow of a transportation network can facilitate a more effective traffic management and help drivers better plan their travel routes in advance [5]; the passenger pickup/dropoff flow prediction is especially useful for more efficient vehicle distribution in the emerging mobility-on-demand (MOD) services [18].

Due to the importance of this problem, a considerable research effort has been devoted to developing effective models for city-wide crowd flow prediction [5, 20, 29]. Zhang et al. [26] proposed a deep learning model ST-ResNet to collectively forecast the in- and out-flows of crowds in each region of a city. To consider the crowd flow between a pair of regions, they further proposed a multi-task deep learning framework to simultaneously predict the node flow and edge flow in a constructed urban spatial-temporal network [28]. Yao et al. [24] proposed a Spatial-Temporal Dynamic Network model named STDN for road network based traffic flow prediction. Zhou et al. [30] proposed to use the attention-based neural network which combined encoder-decoder and ConvLSTM to predict the flows of passenger pickup/dropoff for the mobility-on-demand services. Existing approaches mostly model the crowd flow in city regions as "images", and then apply deep learning models that are effective in image processing for future crowd flow "image" prediction. However, as shown in Figure 1, in many real applications, people not only care about how many people will enter region r_4 (Figure 1(a)), but also need to know where the flows are from (Figure 1(b)), namely the Origin-Destination (OD) of the flows. Although

CIKM '20, October 19–23, 2020, Virtual Event, Ireland
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3412054

crowd flows are highly correlated to OD of flows, most existing works perform two prediction tasks separately without considering the mutual influence of them.

In this paper, we aim to predict the crowd flows and flow OD simultaneously under a unified multi-task learning framework. Our insight is that the two tasks are complementary to each other and share some common latent features, by combining which the prediction performance can be mutually enhanced. However, this problem is non-trivial to address due to the following challenges. First, due to the different data formats of crowd flows and flow OD, it is difficult to learn the shared data representations. There lacks an off-the-shelf method that can effectively decompose the two types of data into shared features and task-specific features for multi-task learning. Although some multi-task learning models are proposed [28], they still suffer from the issue that common features and task-specific features are somewhat blended, leading to information redundancy. Second, the spatial correlations of the crowd flows are complex, and sometimes do not follow the spatial smoothness [25]. With the development of public transportation, the First Law of Geography: "*near things are more related than distant things*" [15] may not fully reflect the spatial correlations of the crowd flows in urban areas. For example, a commercial area may have few crowd flows coming from a park, although they are geographically close to each other; while a residential district far away may have a large number of people flowing into the commercial area. Existing approaches model the crowd flows as "images", and use convolution operation for feature learning. The crowd flows with complex spatial correlations actually are different from the regular Euclidean data like images, and thus CNN is not enough for spatial feature learning. Third, the temporal correlation of the crowd flows is also complex and multi-scale, including temporal closeness, periodic and trend properties. Existing approaches need to manually extract multi-scale temporal correlations separately, and then fuse them together with a carefully designed information fusion component [26].

To tackle the aforementioned challenges, we propose a Multi-task Adversarial Spatial-Temporal Network entitled MT-ASTN to predict the crowd flows and OD of the flows simultaneously. To better capture the spatial correlations of the crowd flows, we propose to model the raw flow trajectories as two types of data representations, semantic spatial-temporal graphs and crowd flow images. The semantic spatial-temporal graph is built based on the historical crowd flows between each pair of regions, which can better reflect the global mobility patterns of crowds. To fuse the feature learning on semantic spatial-temporal graphs and crowd flow images, a heterogeneous spatial-temporal net is proposed to first encode the graphs and images separately, and then integrate them together. General shared-private model decomposes the task features into two feature spaces: one is task-specific features, and the other is the shared features. To address the issue that the shared feature space could contain some task-specific features, while some sharable features could also be mixed in private space [13], we propose to add a discriminative loss for private features and an adversarial loss for shared features to make private features of different tasks more distinguishable while shared features more similar. Finally, to automatically capture the complex temporal correlations, a temporal queue coupled with an attention mechanism is also designed. The attentive temporal queue component enables our model to store the latent feature representations of historical crowd flows in a long period (several months), from which the most useful ones for future prediction are attentively selected.

To summarize, our main contributions are as follows.

- We propose a novel adversarial multi-task learning framework to collectively and simultaneously predict the crowd flows and OD of the flows. Under a shared-private feature learning framework, the proposed approach can effectively couple the two highly correlated tasks to share complementary spatial-temporal knowledge.
- An adversarial loss and a discriminative loss are integrated into the multi-task learning framework to reduce feature redundancy, so that features in different latent spaces are more distinguishable.
- A heterogeneous spatial-temporal net is proposed to integrate the local and global spatial features from the crowd flow images and semantic spatial-temporal graphs. To automatically capture the complex temporal correlations, an attentive temporal queue is also designed to select the most relevant historical data representations for prediction.

The remainder of this paper is organized as follows. Section 2 will review related works. Section 3 will give a formal problem definition and show the model framework. Section 4 will introduce our methodology. Evaluations are given in Section 5. Finally, this paper is concluded in Section 6.

2 RELATED WORK

Traffic flow prediction. Traffic flow prediction, which is relevant to crowd flow prediction has been studied for many years in intelligent transportation systems [2, 8, 29]. The difference between traffic prediction and crowd flow prediction is that usually traffic prediction focuses on predicting the traffic on the road segments or a road network [8] rather than over cell regions. Traditional traffic prediction models are mostly statistics-based approaches such as ARIMA and SVR, and they mainly focus on predicting the traffic of one single road, a set of road segments or a road network. Williams [22] used ARIMA model to predict the short-term traffic flow. Mecit and Gurcan [2] put forward a regression model which included two kinds of traffic incident detection algorithms for traffic flow prediction. Hofleitner et al. [8] proposed a dynamic Bayesian network to predict the traffic flow of an entire arterial road network with hundreds of road links based on the sparse probe data. The major limitation of the above statistics-based traffic flow prediction models is that the complex temporal and spatial correlations of the traffic flow data are hard to be captured due to their limited learning ability.

Crowd flow prediction. Recently, with the advances of deep learning techniques, deep neural network models are widely used for urban crowd flow prediction tasks [18, 19]. A line of studies applied CNN to capture the spatial correlation by treating the traffic flow data of the entire city as images. Zhang et al. [26, 27] proposed a deep learning model ST-ResNet to collectively forecast the in- and out-flows of crowds in each cell region of a city. Another line of research is combining CNN model and RNN model to capture both spatial and temporal correlations. Yao et al. [24] proposed a Spatial-Temporal Dynamic Network (STDN) model for road network based

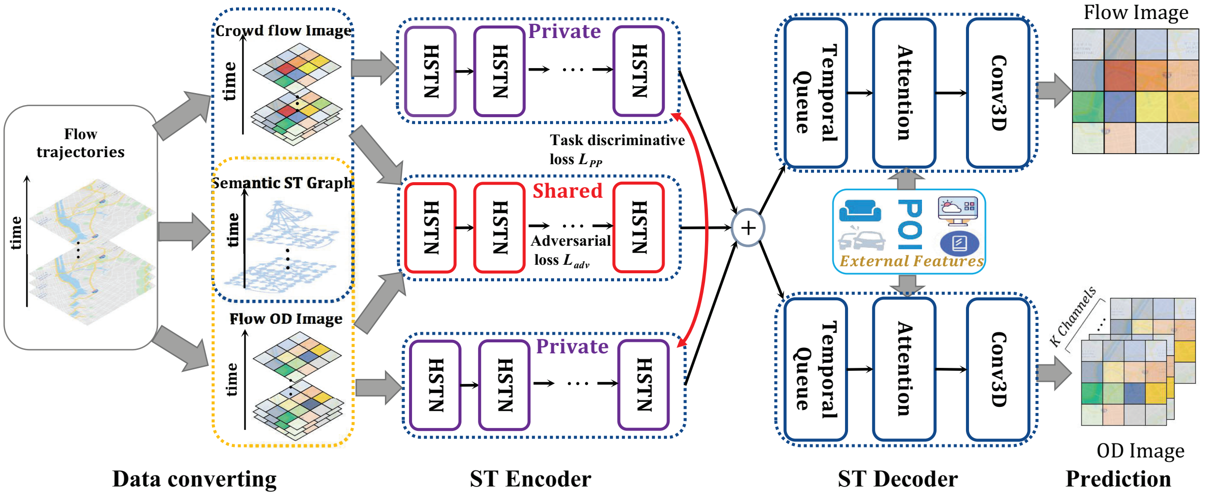


Figure 2: Framework of the proposed MT-ASTN model

traffic flow prediction. [10] proposed the Diffusion Convolutional Recurrent Neural Network (DCRNN) to model the traffic flow as a diffusion process on a directed road graph. However, these works consider crowd flow prediction as a single task, but ignore the impact of flow OD.

OD prediction. Origin-Destination (OD) prediction can benefit many real applications such as ride-hailing services and traffic management. Traditional methods mostly used regression based approaches or other statistics-based approaches to predicting or estimating the dynamic vehicle OD matrix in a transportation network [1]. Recently, researchers focus more on region-level flow OD prediction rather than the vehicle OD prediction on a road network, and more advanced techniques are used such as deep learning and tensor factorization models. Wang et al. [21] proposed Grid-Embedding based Multi-task Learning model namely GEML to predict the number of passenger demands from one region to another. Lie et al. [11] studied the taxi origin-destination demand prediction, and proposed a contextualized spatial-temporal network which can model the local spatial context, temporal evolution context, and global correlation context. Ren et al. [14] modeled the temporal OD trip matrix as a four-order tensor consisting of four attributes: origin, destination, vehicle type and time, and then proposed to use tensor decomposition technique to forecast future traffic demand. [4] developed a deep learning model called multi-scale convolutional long short-term memory network (Multi-ConvLSTM) to predict the future travel demand and the OD flows.

However, most existing works consider OD flow prediction and crowd flow prediction as two separate problems and ignore the high correlation between them. Zhang et al. [28] proposed a multi-task deep learning model MDL which can predict the flows at the nodes and transitions between nodes in a spatial-temporal network simultaneously. However, MDL simply concatenates the features of different tasks without distinguishing which features are shared and which one should be task-specific. How to effectively extract the shareable features and perform the predictions of OD and crowd flows collectively still remains an open problem.

3 NOTATIONS AND PROBLEM DEFINITION

In this section, we will first give some notations to help us state the studied problem. Then a formal problem definition will be given.

Definition 1. ***Cell region*** *The city under study is divided into a $m \times n$ grid map based on the longitude and latitude. Each grid is an equal-sized cell region. We denote all the cell regions as $R = \{r_{1,1}, ...r_{i,j}, ...r_{m,n}\}$, where $r_{i,j}$ is the i-th row and j-th column cell region of the grid map.*

Definition 2. ***Crowd flow image*** *Let $\mathcal{P}$ be a collection of crowd flow trajectories. Given a cell region $r_{i,j}$, the corresponding inflow and outflow of the crowds in time slot t can be defined as*

$$x^t_{in,i,j} = \sum_{T_r \in \mathcal{P}} |\{l > 1 | g_{l-1} \notin r_{i,j} \wedge g_l \in r_{i,j}\}|$$

$$x^t_{out,i,j} = \sum_{T_r \in \mathcal{P}} |\{l > 1 | g_l \in r_{i,j} \wedge g_{l+1} \notin r_{i,j}\}|$$

where $T_r : g_1 \rightarrow g_2 \rightarrow ... \rightarrow g_{T_r}$ is a trajectory in $\mathcal{P}$, and g_l is the geospatial coordinate; $g_l \in r_{i,j}$ means g_l is within region $r_{i,j}$; $|\cdot|$ denotes the cardinality of a set. Following [26], we denote the inflow and outflow of all the cell regions in t as a crowd flow tensor $\mathcal{X}^t \in \mathcal{R}^{m \times n \times 2}$, where $\mathcal{X}^t_{i,j,0} = x^t_{in,i,j}, \mathcal{X}^t_{i,j,1} = x^t_{out,i,j}$.

Definition 3. ***Flow OD matrix*** *We define the flow OD matrix at time slot t as $D^t \in \mathcal{R}^{N \times N}$, where $N = m \times n$ is the number of regions and each element $d^t_{i,j}$ denotes the sizes of flows starting from i-th cell region and ending at j-th cell region of R.*

Definition 4. ***Semantic spatial-temporal graph*** *We define the semantic spatial-temporal graph at time slot t as $G^t = \{V, E^t\}$, whose nodes V are the cell regions. There is an edge $e^t_{i,j}$ if there are flow trajectories whose origin is v_i and destination is v_j. The weight $w^t_{i,j}$ on edge $e^t_{i,j}$ is the size of flow from v_i to v_j. Note that the adjacency matrix of G^t is the flow OD matrix D^t.*

Definition 5. ***Flow OD image*** *Given the flow OD matrix $D^t \in \mathcal{R}^{N \times N}$ in t, we construct the flow OD images $\mathcal{M}^t \in \mathcal{R}^{m \times n \times N}$ with*

$N = m \times n$ channels. Each channel $\mathcal{M}^t(:,:,i)$ denotes the size of trajectories starting from i-th region and ending to all the other regions.

Based on the above definitions, we formally define the studied problem as follows.

Problem Definition 1. *Given the crowd flow images and the flow OD images $\{\mathcal{X}^t, \mathcal{M}^t | t = 1, \ldots, T\}$ in the cell regions R over T time slots, and the external context data E (e.g.weather, holiday, etc.), our goal is to predict $\{\mathcal{X}^{T+1}, \mathcal{M}^{T+1}\}$ simultaneously.*

4 METHODOLOGY

Figure 2 shows the framework of the proposed MT-ASTN, which consists of four major steps: data converting, spatial-temporal (ST) encoder, spatial-temporal (ST) decoder and prediction. In the data converting step, we convert the raw flow trajectories to crowd flow images, flow OD images and semantic spatial-temporal (ST) graphs. This step will be introduced in detail in Section 4.1. In the ST encoder step, the shared-private framework is adopted for jointly encoding features of the two tasks. As a popular multi-task learning framework, shared-private framework aims to separate shared features from task specific features. As shown in the figure, the designed ST encoder consists of a shared encoder and two private encoders for two tasks, respectively. The ST encoder contains several stacked Heterogeneous Spatio-Temporal Net (HSTN) layers. HSTN is designed to fuse feature learning of flow/OD images and semantic ST graphs, which will be introduced in detail in Section 4.2. In order to extract pure shared features from the entire features, we propose to use adversarial learning to train the shared ST decoder. A discriminative loss L_{PP} on task classification based on the task-specific private features is also proposed. Minimizing L_{PP} can prevent the extracted private features from mixing with some common features. We will elaborate this step in Sections 4.3.

Next, the task-specific features and shared features are fused and input into ST decoder. ST decoder includes the components of temporal queue, attention and Conv3D layers. To capture the complex temporal dependencies of the crowd flows including smoothness and periodicity, we design a novel component called temporal queue to store the latent data representations in a past long time period. Then a conditional multi-head self-attention is designed to automatically capture the most relevant historical data representations from the temporal queue to the current prediction. External features including weather and holidays are also incorporated into the attention model as external conditions. This step will be introduced in Section 4.4. Finally, several Conv3D layers are stacked to generate the final prediction on the future crowd flow image and OD image simultaneously. The overall objective function for the multi-task prediction will be described in Section 4.5.

4.1 Data Converting

Based on Definitions 2-5, we need first convert the raw flow trajectories $\mathcal{P}$ to crowd flow images, flow OD images and semantic ST graphs. Following [26, 28], we first model the crowd flow images with the size $m \times n \times 2$ as time-varying spatial maps which can be represented as time-ordered sequence of tensors, so that convolution operations can be applied for feature learning. Similarly, we first construct the flow OD matrices with the size $K \times K$ based on the origin and destinations of the raw trajectories, where $K = m \times n$. Then we convert the flow OD matrices into flow OD images, which are represented as three-dimensional tensors with the size $m \times n \times K$ as shown in Figure 3. The reasons that we convert a flow OD matrix to a flow OD image are two-fold. First, the data format of a flow OD image ($m \times n \times K$) is consistent with that of a crowd flow image ($m \times n \times 2$), and thus multi-task learning can be performed on the two data with similar neural network architecture to facilitate shareable feature learning. Second, convolution operations can be conducted on the flow OD images to learn local spatial features. As the crowd flow data may not follow the spatial smoothness property, the data representations of crowd flow images and flow OD images are not effective to explicitly reflect the global spatial correlations. For example, assume r_i is a residential area and r_j is a central business district. Although r_i and r_j may be geographically far away from each other, the crowd flows between them can be high, because people living in r_i need to go to r_j for work. To capture the global crowd flows, we also construct semantic ST graphs $G^t = \{V, E^t\}$ based on the flow OD matrix as shown in Figure 3. The nodes V of G^t are the cell regions, and the edges E^t are the crowd flows between each pair of region nodes.

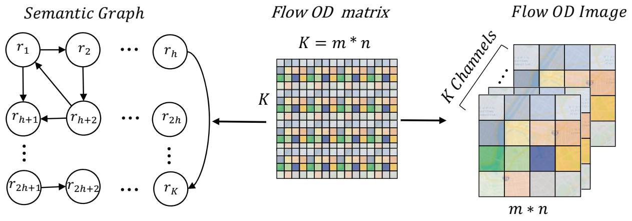


Figure 3: Converting flow OD matrix to OD image and semantic ST graph

4.2 Private Spatial-Temporal Encoder

The crowd flow images, flow OD images and the semantic ST graphs are input into the private ST encoder for task-specific private features learning. As images and graphs are represented as different data structures, they cannot be processed by a unified neural network structure. To address this issue, we propose a heterogeneous spatial-temporal network (HSTN) to first learn the data representations of images and graphs separately, and then fuse them together.

Heterogeneous Spatial-Temporal Net. HSTN adopts Conv3D layers to learn the latent representations h^t_{image} for crowd flow and flow OD images, and a GCN combined with LSTM model to learn the latent representations h^t_G for the semantic ST graphs. Here we use 3-dimensional convolutions on the tensors of crowd flow images and flow OD images to capture the local spatial and temporal correlations. To more broadly capture the spatial correlations, we construct a hierarchical semantic ST graph based on the crowd flows among the regions, and perform graph representation learning with the combination of GCN and LSTM. Finally, the two types of data representations are integrated by the following formula

$$h^t = h^t_{image} \oplus h^t_G \tag{1}$$

where $\oplus$ is the feature *concatenation* operation across channels.

Hierarchical Spatial Feature Learning. GC-LSTM [3] was initially proposed to learn the feature representations for a sequence of dynamic graphs. It contains a GCN module to capture the spatial features of each graph and a LSTM module to capture the temporal features of the graph sequence. As the semantic ST graph G^t changes over time, GC-LSTM is suitable to be adopted for feature learning on a sequence of dynamic graphs. However, the semantic ST graph shown in Figure 3 only models the crowd flows among fixed size cell regions, but cannot reflect the flows among larger areas. To more broadly capture the crowd flow patterns among regions of different scales, we propose to construct a hierarchical semantic ST graph as shown in Figure 4, and introduce the hierarchical graph convolutional LSTM (HGC-LSTM) to learn the graph representations.

Figure 4 illustrates the construction of a 3-layer semantic ST graph. As shown in the left part of the figure, given an area with 16 cell regions from r_1 to r_{16}, a layer-1 graph G_1^t is naturally constructed whose nodes are corresponding to the cell regions. The directed edges among the nodes indicate the crowd flow among the regions. Based on G_1^t, a layer-2 graph G_2^t is constructed in the middle part of the figure. G_2^t contains 4 nodes with each one corresponding to a larger region containing 4 small cell regions in G_1^t. For example, nodes $\{r_1, r_2, r_3, r_4\}$ are merged as a new node r_{17}, because the four nodes are geographically close to each other. We also add edges from node $\{r_1, r_2, r_3, r_4\}$ to node r_{17}. Similarly, a layer-3 graph G_3^t is also constructed based on G_2^t as shown in the right part of the figure, and all the nodes in G_2^t have an edge to node r_{21} of G_3^t. Finally, we merge the three-layer graphs and construct a hierarchical graph $G^t = G_1^t \cup G_2^t \cup G_3^t$ which contains 21 nodes.

Next, we use HGC-LSTM model on the hierarchical semantic ST graphs over time for feature learning as follows.

$$\begin{aligned}
i^t &= \sigma\left(W_{xi} f_{\odot}(\mathbf{X}^t, D^t) + W_{hi} f_{\odot}(h_G^{t-1}, D^t) + W_{ci} \circ C^{t-1} + b_i\right),\\
f^t &= \sigma\left(W_{xf} f_{\odot}(\mathbf{X}^t, D^t) + W_{hf} f_{\odot}(h_G^{t-1}, D^t) + W_{cf} \circ C^{t-1} + b_f\right),\\
C^t &= f^t \circ C^{t-1} + i^t \circ tanh\left(W_{xc} f_{\odot}(\mathbf{X}^t, D^t) + W_{hc} f_{\odot}(h_G^{t-1}, D^t) + b_c\right),\\
o^t &= \sigma\left(W_{xo} f_{\odot}(\mathbf{X}^t, D^t) + W_{ho} f_{\odot}(h_G^{t-1}, D^t) + W_{co} \circ C^t + b_o\right),\\
h_G^t &= o^t \circ tanh\left(C^t\right),
\end{aligned} \tag{2}$$

where '∘' denotes the Hadamard product, σ is the logistic sigmoid function, i^t, f^t, C^t o^t, and h_G^t are input gate, forget gate, memory cell, output gate and hidden state, and $W_{\alpha\beta}(\alpha \in \{x, h, c\}$, $\beta \in \{i, f, o, c\})$ are the parameters of convolutional layers in HGC-LSTM, $f_{\odot}$ represents the GCN [9] operator as follows.

$$f_{\odot}(\mathbf{X}^t, D^t) = \delta(D^t \mathbf{X}^t W) \tag{3}$$

where D^t is the adjacency matrix of the hierarchical semantic ST graph G^t, $\mathbf{X}^t$ is the graph representation matrix, $\delta(\cdot)$ is a non-linear activation function such as ReLU, and W is a weight matrix. As the hierarchical semantic ST graph contains multi-layer graphs, we need to fuse the node representations of the graphs in different layers. As the format of the learned graph features $h_G^t \in \mathcal{R}^{K \times K}$ is different from the image features $h_{image}^t \in \mathcal{R}^{m \times n}$, we cannot fuse them directly with formula (1). Therefore, we need to transform the graph features to the format of graph maps with the size $\mathcal{R}^{m \times n \times C}$ first, where C is the number of channels. Then for each layer graph map features $h_{G_k^t}^t$, we perform the deconvolution operation to make the feature size the same as $h_{G_1^t}^t$, and add it with $h_{G_1^t}^t$. The fused graph map features of the multi-layer graphs can be represented as

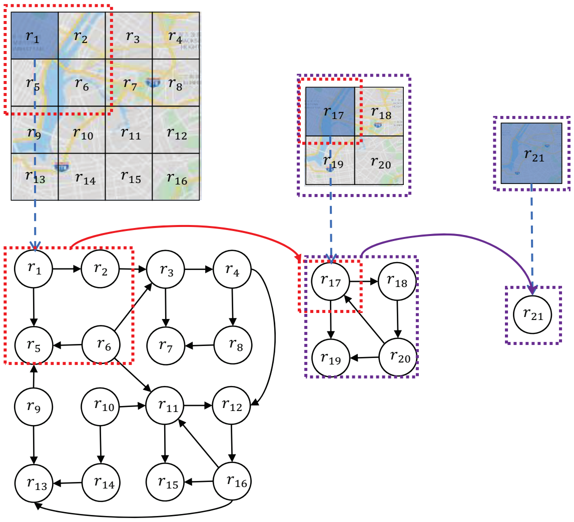


Figure 4: Illustration of hierarchical semantic ST graph

$$h_G^t = \sum_k Deconv(h_{G_k^t}^t) \tag{4}$$

where $Deconv(\cdot)$ denotes the deconvolution operation.

Task discriminative loss L_{PP} with private features. In order to prevent the extracted private features from mixing with some shared features, we introduce the task discriminative loss L_{PP} based on private features. L_{PP} aims to make the private ST encoders extract task-specific features, so that the private features of different tasks are exclusive to each other and thus can be easily distinguished. The goal is to minimize the classification error of the private features coming from which task. This can be achieved by minimizing the cross-entropy loss L_{PP} as follows.

$$L_{PP} = -\frac{1}{\mathcal{L}} \sum_{i=1}^{\mathcal{L}} \sum_{j=1}^{m} y^{i,j} logDis\left(h_{private}^{i,j}\right) \tag{5}$$

where $y^{i,j}$ denotes the task label of the i-th data sample, $h_{private}^{i,j}$ is the private feature matrix of the data and $Dis(\cdot)$ is the task discriminator. We use the following *softmax* function as the task discriminator $Dis(\cdot)$

$$Dis\left(h_{private}^{i,j}\right) = Softmax\left(W h_{private}^{i,j} + b\right) \tag{6}$$

where W is a learnable parameter matrix and b is the bias vector.

4.3 Shared Spatial-Temporal Encoder

Shared ST encoder aims to learn the common features that are shared by all the tasks. Inspired by the work [12], we also adopt adversarial learning to help the shared ST encoder extract pure shared features to reduce information redundancy. Orthogonality constraint between shared features and task-specific private features are also incorporated to make them more separable.

Adversarial loss L_{adv}. Generative Adversarial Networks (*GANs*) [7] are currently popular deep learning based generative models and are widely explored in diverse domains. The goal of GANs is to learn a generative data distribution $P_G(x)$ that is similar to the real data distribution $P_{data}(x)$ via an adversarial learning process, which can be achieved by optimizing such a min-max game

$$\underset{G}{Min}\,\underset{D}{Max}(E_{x\sim P_{data}}[logD(x)] + E_{z\sim p(z)}[log(1-D(G(z)))]) \quad (7)$$

where $G(\cdot), D(\cdot)$ are generator and discriminator, respectively.

Inspired by adversarial networks, we introduce the adversarial learning procedure to the shared ST encoder to extract pure shareable features. The general idea is that as the shared ST encoder extracts features that are invariant to both tasks, a task classifier cannot reliably distinguish the tasks based on such features. Based on this idea, the shared features can be learned through optimizing such a min-max function.

$$L_{adv} = \frac{1}{\mathcal{L}} \min_{\theta_s}\left(\max_{\theta_D}\left(\sum_{i=1}^{\mathcal{L}}\sum_{j=1}^{m} y^{i,j} logDis\left(h^{i,j}_{shared}\right)\right)\right) \quad (8)$$

where $y^{i,j}$ is the ground-truth task label indicating the type of the task, and $h^{i,j}_{shared}$ is the hidden state learned by shared Encoder. $Dis\left(h^{i,j}_{shared}\right) = Softmax\left(Wh^{i,j}_{shared} + b\right)$ represents task discriminator similar to formula (6). The differences are that here we use shared features $h^{i,j}_{shared}$, and we want to maximize the classification error. The basic idea is a min-max optimization. Given the shared features, the shared ST encoder generates a representation to mislead the task discriminator $Dis(\cdot)$. At the same time, $Dis(\cdot)$ tries its best to classify which task the features come from.

Orthogonality Constraint. An issue of the adversarial learning method for shared feature learning is that it cannot guarantee that all the shared features can be fully extracted by the shared ST encoder. That means some shared features may also appear in private feature space. To address this issue, besides adding the shared feature based task discriminative loss L_{PP}, we further add the orthogonality constraint as follows to encourage the shared and private encoders to extract different aspects of the inputs so that the two types of features are orthogonal to each other.

$$L_{orth} = \sum_{i=1}^{m} ||H^T_{i,shared} H_{i,private}||^2_F \quad (9)$$

where $||\cdot||^2_F$ is the squared Frobenius norm. $H_{i,shared}$ is the shared feature matrix and $H_{i,private}$ is the task-specific feature matrix for the i-th task.

4.4 Spatial-Temporal Decoder

The learned shared features and task-specific private features are then input into the ST decoder to decode the data representations for prediction. As show in the right part of Figure 2, the ST decoder first integrates the shared features and task-specific features for each task, and then inputs the features into an attentive temporal queue module, followed by a Conv3D layer. First, we integrate the shared features and private features as follows.

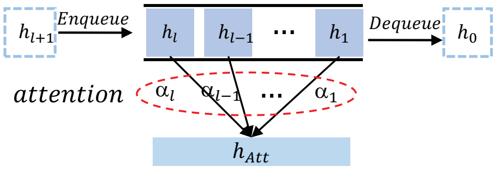


Figure 5: Illustration of attentive temporal queue

$$h^t_{all} = h^t_{shared} + h^t_{private} \quad (10)$$

where '+' represents sum operation across channels. Then, the feature h^t_{all} is input into *Temporal Queue* coupled with an attention mechanism to learn the temporal dependency.

Temporal Queue. Given a data sequence, existing neural networks including RNN and LSTM can only capture the short-term temporal dependency, but are less effective to learn long-term dependency, which is common in many spatial-temporal data prediction problem. Especially, in our task the temporal correlations of the crowd flows are multi-scale including smoothness, periodicity and trend. In order to automatically capture the complex temporal dependencies, we design a novel temporal queue which can store the latent features of a long period of time (e.g., several months). Then an attention mechanism is used to decide which previous time slots should be more attentive and the corresponding latent features are more helpful to predict the future. Figure 5 shows the architecture of temporal queue with length l, which is long enough to enable the model capture the long-term dependency. Note that the temporal queue always stores the most recent latent features in the past l time slots. It can dynamically dequeue the old features and enqueue the most recent ones when the queue is full.

Conditional Multi-head Self-Attention. Here we adopt the *multi-head self-attention*, which is a famous attention mechanisms used in Transformer [17]. Compared with other attention methods, self-attention can learn long-range dependencies from our designed temporal queue which contains a long sequence of historical latent features. Multi-head attention allows the model to jointly attent to information from different representation subspaces at different positions, and thus is more effective and robust. By considering the external context features including weather conditions and holidays, we design the conditional multi-head self-attention, which has three important components: *Query*, *Key*, and *Value*. This attention mechanism is computed by concatenating the ouput matrix of each attention head and projecting it by W:

$$\begin{aligned} MultiHead &= Concat(head_1, \ldots, head_i) * W \\ head_i &= Attention\left(QW^Q_i, KW^K_i, VW^V_i\right) \end{aligned} \quad (11)$$

where W is the parameter matrix, W^Q_i, W^K_i, W^V_i are the linear transformation parameters of the *query*, *key*, and *value* respectively. At each *conditional attention head*, the hidden state with external information stored in temporal queue $\mathcal{H}_i = \left([h^{t-l+1}, E^{t-l+1}], \ldots, [h^t, E^t]\right)$ where E^t is the external features in time slot t and [,] denotes the

Table 1: Dataset description

Dataset	NYCBike	NYCTaxi
Longitude	-74.02∼-73.95	-74.02∼-73.95
Latitude	40.67∼40.77	40.67∼40.77
Time span	1/1/2015∼31/12/2015	1/1/2015∼31/12/2015
Time interval	1 hour	1 hour
Grid map size	(16, 16)	(16, 16)
	Trajectory data	
# of trips	9 million	160 million
# of time intervals	8,754	8,754
	External features	
Weather conditions	(precipitation, snow, temperature, etc.)	
Days	weekday, weekend, holiday etc.	

concatenation operation of the two types of features, is projected onto the *query*, *key*, and *value* spaces.

Finally, we calculate the output matrix of the weighted sum of value tensors which is formulated as:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}}) * V. \tag{12}$$

4.5 Overall Objective Function

In the final prediction step, we aim to minimize the prediction error of the two tasks as follows.

$$L_{task} = \frac{1}{m\mathcal{L}} \sum_{i=1}^{m} \sum_{j=1}^{\mathcal{L}} ||\hat{Y}^{i,j} - Y^{i,j}||^2 \tag{13}$$

where m is the number of tasks, $\mathcal{L}$ is the training sample size, $\hat{Y}^{i,j}$ is the prediction and $Y^{i,j}$ is the ground truth.

The final loss function contains four parts: prediction loss of the two tasks L_{task}, private-private features discriminative loss L_{PP}, adversarial loss L_{adv} and orthogonality constraint loss L_{orth}. We combine them together and the overall loss function is as follows.

$$L_{all} = L_{task} + L_{PP} + L_{adv} + L_{orth} \tag{14}$$

The network model is trained with backpropagation and the adversarial training is optimized via gradient reversal layers [6].

5 EXPERIMENT

5.1 Dataset and Experiment Setup

5.1.1 Datasets. We select two large datasets that are widely used in crowd flow prediction for evaluation: *BikeNYC*, and *TaxiNYC*. The details of the datasets are introduced as follows.

NYCBike. This dataset contains more than 9 million bike trips in New York from January 2015 to December 2015. In total, NYCBike has established over 600 bike stations and 10,000 bikes in New York. Each bike trip contains the trip duration, start/end station IDs, start/end timestamps, station Latitude/Longitude and bike ID. For this dataset, we use the first 11 months data for training and validation, and the last month data for testing.

NYCTaxi. This dataset contains over 160 million taxicab trip records in New York from January 2015 to December 2015. On average, there are about 13 million trip records each month. Each taxi trip record includes fields capturing pick-up and drop-off dates/times, pick-up and drop-off locations, trip distances, itemized fares, rate types, payment types, and driver-reported passenger counts. For this dataset, we also use the first 11 months data for training and validation, and the last month data for testing.

We also use some external features including weather conditions and holidays. The weather conditions include precipitation, snow, temperature, wind speed, etc. Whether the day is weekday, weekend or holiday is also considered as the people mobility patterns on holidays and regular days are quite different. The data description on the two datasets and external features are shown in Table 1.

5.1.2 Baselines. We compare the proposed MT-ASTN with the following 6 baseline methods, including both single-task learning and state-of-the-art multi-task learning methods.

- **ARIMA** Auto-Regressive Integrated Moving Average(ARIMA) is a classic statistics-based method for time series prediction.
- **ConvLSTM** [16] ConvLSTM is a variant of LSTM which contains a convolution operation inside the LSTM cell. ConvLSTM considers both geographical spatial and temporal dependency of the spatial-temporal data, and is widely used in many spatial-temporal prediction tasks.
- **STResNet [26]** It is a state-of-the-art neural network based single-task learning model for urban crowd flow prediction. It stacks convolutional layers and residual unites to capture the spatial and short/long-term temporal dependencies. External features are also incorporated into ST-ResNet.
- **STDN [24]** Spatio-Temporal Dynamic Network(STDN) is a state-of-the-art unified framework to learn the dynamic similarity between locations and long-term periodic temporal shifting for urban traffic flow prediction.
- **GEML [21]** Grid-Embedding based Multi-Task Learning (GEML) is a multi-task learning framework that predicts the flow OD matrix and crowd flows simultaneously. It uses grid embedding and multi-task LSTM to capture the spatial-temporal representations of the crowd flow data.
- **MDL [28]** MDL is a recent state-of-the-art multi-task learning framework for predicting both the node flows and edge flows on a spatial-temporal network.

To further evaluate whether the key components used in our model are useful to the studied problem, we also compare the full version MT-ASTN with the following variants.

- **MT-ASTN(L_{adv})** This model removes the adversarial loss L_{adv}. Through comparing with it, we test whether the proposed adversarial learning can help extract better shared features and thus improve the prediction performance.
- **MT-ASTN(Gra)** This model drops the features of the semantic ST graphs. Through comparing with this model, we test whether integrating the semantic ST graphs can enhance the features of crowd flow images and flow OD images, and thus improve the model performance.
- **MT-ASTN(Que)** This model removes the temporal queue from the ST decoder. Attention is only applied on the input data sequence. Through comparing with it, we test whether the temporal queue is useful for our model to capture the complex temporal correlations of the crowd flows.

Table 2: RMSE and MAE comparison among different methods

Model	RMSE				MAE			
	NYCBike		NYCTaxi		NYCBike		NYCTaxi	
	Crowd flow	Flow OD	Crowd flow	Flow OD	Clowd flow	Flow OD	Crowd flow	Flow OD
ARIMA	21.821	0.964	68.709	1.844	16.521	0.148	42.623	2.545
ConvLSTM	6.997	0.120	23.169	0.551	3.482	0.050	11.344	0.320
STResNet	4.889	0.138	23.840	0.234	2.364	0.027	12.538	0.118
STDN	6.491	0.127	21.169	0.159	1.794	0.021	8.637	0.074
GEML	6.344	0.147	22.073	0.670	2.828	0.014	10.449	0.136
MDL	8.715	0.154	21.492	0.153	4.250	0.041	11.750	0.095
MT-ASTN	**2.995**	**0.074**	**12.299**	**0.087**	**1.413**	**0.011**	**6.417**	**0.030**

5.1.3 Implementation Details. We implement out model with Pytorch framework on NVIDIA Tesla M40 GPU. The model parameters are set as follows. The data size of crowd flow images is 6×16×16×2 for both datasets, where 6 is the previous time slot length used for prediction, 16×16 is the size of the cell regions, and 2 is the number of channels representing inflow and outflow. The input data size of flow OD image is 6 × 16 × 16 × 256, where 256 is the number of channels which is also the number of cell regions. The learning rate and batch size are set to 0.001 and 48, respectively. The Conv3D and HGC-LSTM in HSTN model of ST Encoder contain 3 layers whose structure is 6 × 16 × 16 × 32, 6 × 16 × 16 × 64 and 6 × 16 × 16 × 128. We use one layer Conv3D in ST Decoder for crowd flow image prediction, whose structure is 1 × 16 × 16 × 2. The structure of Conv3D in ST Decoder for flow OD prediction is 1 × 16 × 16 × 256. The baseline methods are implemented based on the original papers or we use the publicly available code. The parameters of baseline methods are set based on the original papers.

5.1.4 Evaluation Metrics. We adopt Mean Absolute Error(MAE) and Root Mean Square Error(RMSE) as the evaluation metrics defined as follows

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|\hat{\mathcal{X}}^{t+1} - \mathcal{X}^{t+1}|, RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}||\hat{\mathcal{X}}^{t+1} - \mathcal{X}^{t+1}||^2} \tag{15}$$

where $\hat{\mathcal{X}}^{t+1}$ is the prediction and $\mathcal{X}^{t+1}$ is the ground truth.

5.1.5 Convergence of The Algorithm. Figure 6 shows the training loss curves of the algorithm on the two datasets. One can see that MT-ASTN converges after about 150 epochs on both datasets, which shows it converges quickly. The loss curves do not smoothly drop, and there are some fluctuations on the loss during training. This is mainly because adversarial training is usually hard to train than regular neural networks. In the following experiment, we train MT-ASTN on both datasets 200 epochs.

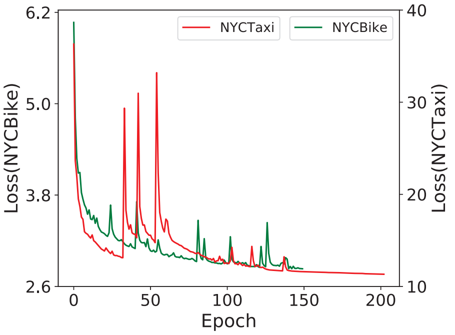


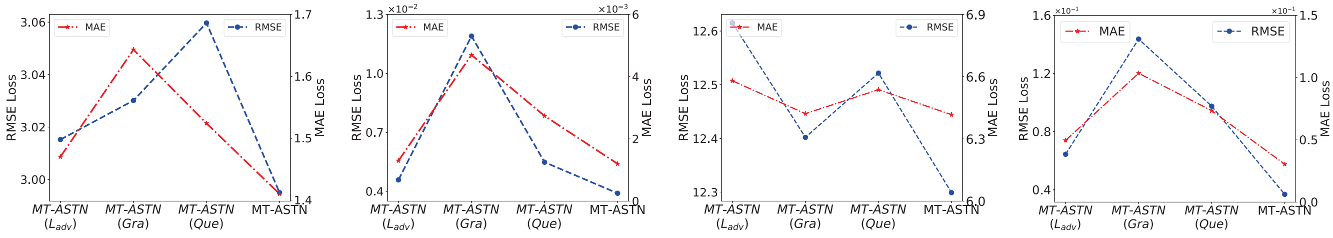
Figure 6: Loss curves of MT-ASTN on the two datasets

5.2 Comparison with Baselines

Table 2 shows the performance comparison among different methods over the two datasets. The best results are highlighted with bold font, and the best results achieved by baselines are underlined. It shows that the proposed MT-ASTN achieves the best performance among all the method on both tasks. On NYCBike dataset, compared with the best results achieved by baselines, MT-ASTN reduces MAE of crowd flow prediction and flow OD prediction from 1.794 (STDN) to 1.413, and from 0.014 (GEML) to 0.011, respectively. On NYCTaxi dataset, MT-ASTN improves the MAE of the two tasks from 8.637 (STDN) to 6.417, and from 0.074 (STDN) to 0.030, respectively. For RMSE comparison, the performance improvement of MT-ASTN is more significant. On crowd flow prediction, MT-ASTN reduces RMSE from 4.889 achieved by the best baseline STResNet to 2.995, and from 21.169 achieved by STDN to 12.299 for the two datasets, respectively. Both are significant improvements. On flow OD prediction, the drops of RMSE on two datasets are also remarkable from 0.120 (ConvLSTM) to 0.074 and from 0.153 (MDL) to 0.087, respectively. It shows that RMSE and MAE on NYCBike are much smaller than NYCTaxi. This is mainly because the bike trips are much sparser than taxi trips. In addition, the OD of bike trips can only be the bike stations deployed in fixed locations, and thus is much easier to predict than taxi trips. This result verifies that MT-ASTN is more effective than existing state-of-the-art single- and multi-task learning approaches on the two prediction tasks.

5.3 Comparison with Variant Models

To examine whether the components in MT-ASTN are all helpful to the prediction task, we compare MT-ASTN with its variants MT-ASTN(L_{adv}), MT-ASTN(Gra) and MT-ASTN(Que). The result is shown in Figure 7. One can see that the adversarial loss, the semantic ST graph features and temporal queue are all useful to the model as removing any one of them will increase the prediction error. On NYCBike dataset, the graph features seem more important on both tasks because the prediction error increases remarkably when these features are ignored. On NYCTaxi dataset, the adversarial loss is more important in crowd flow prediction, while the semantic ST



(a) NYCBike crowd flow (b) NYCBike flow OD (c) NYCTaxi crowd flow (d) NYCTaxi flow OD

Figure 7: RMSE and MAE comparison with variant methods

Table 3: Single- and multi-task learning comparison

Dataset	Methods	Crowd Flow		Flow OD	
		RMSE	MAE	RMSE	MAE
NYCBike	ST-STN	3.103	1.562	0.112	0.015
	MT-ASTN	2.995	1.413	0.074	0.011
NYCTaxi	ST-STN	15.699	7.604	0.131	0.077
	MT-ASTN	12.299	6.417	0.087	0.030

graph features are more useful in flow OD prediction. Combining these components together achieves the lowest RMSE and MAE, demonstrating that all of them are useful to the studied problem.

5.4 Single-task *vs* Multi-task Learning

To test whether multi-task learning can improve the performance of each task, we compare MT-ASTN with the single-task version of MT-ASTN named ST-STN. The single-task model ST-STN has the similar network structure as MT-ASTN shown in Figure 2. The difference is that ST-STN performs crowd flow prediction and flow OD prediction separately without learning the shared features. Thus ST-STN does not need the adversarial learning to extract shared features. The comparison result is shown in Table 3. It shows that the RMSE and MAE values achieved by multi-task learning MT-ASTN are lower than that of the ST-STN, which demonstrates the proposed multi-task learning model can capture and transfer task-invariant features across tasks, and thus improve the performance of both tasks. One can also see that the performance improvement on flow OD prediction is more significant than on the crowd flow prediction on both datasets. The learned shared features help more on the flow OD prediction task, reducing RMSE of NYCTaxi from 0.131 to 0.087 and NYCBike from 0.112 to 0.074.

5.5 Prediction Result *vs* Ground Truth

To further intuitively illustrate how accurately our model can predict the crowd flows, we visualize the predicted crowd flows and the ground truth in one figure as depicted in Figure 8. Due to space limitation, we show a case study on one month crowd flows of region $r_{8,8}$. From top to down, the four figures show the taxi inflow, taxi outflow, bike inflow and bike outflow, respectively. One can see that the red curves of prediction can accurately trace the blue curves of the ground truth including sudden changes, which demonstrates the effectiveness of the proposed model. The figure also shows that the two crowd flow datasets present obvious periodical change characteristics, which is consistent with the people mobility patterns in cities. Our model can perfectly capture the periodicity of the data, which is largely due to the usage of the proposed attentive temporal queue. The result also shows that the designed attentive temporal queue component is effective to complex temporal trends from the long-range temporal data.

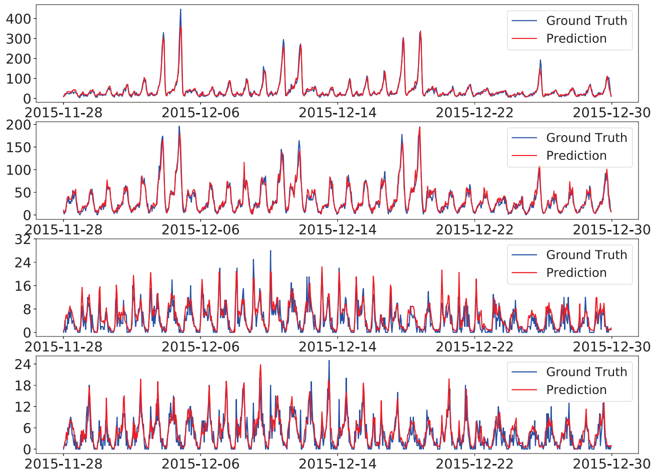


Figure 8: Prediction *vs* ground truth on region $r_{8,8}$ (top to down: taxi inflow, taxi outflow, bike inflow and bike outflow)

5.6 Case Study on the Attention Mechanism

To test whether the proposed conditional multi-head self-attention is effective to capture the complex temporal correlations, especially the periodicity of the crowd flows, we draw the attention value curve on the temporal queue whose length is 15 days. For comparison, we also show the attention value curve of hierarchical attention [23], which is an effective hierarchical attention mechanism used in text classification. Hierarchical attention contains word-level attention and sentence-level attention. If we consider the sentences of a document as the temporal queue, sentence attention can be applied on learning the relevant scores of the elements in temporal queue to the prediction. We show a case study on the attention value curves of the two methods in Figure 9. This figure shows the

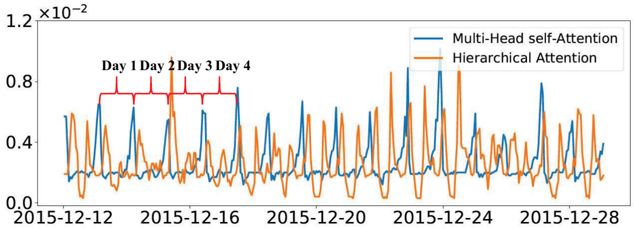


Figure 9: Attention values on temporal queue

learned attention values for the prediction on the time slot 16:00 pm, Dec 29, 2015. One can see that the curve of multi-head self-attention presents clear periodical trend, which well matches the crowd flows curves as shown in Figure 8. The curve of hierarchical attention, although also fluctuates over days, does not well capture the periodical pattern as more than one peaks can appear in one day. It shows that the proposed conditional multi-head self-attention can more effectively capture the periodicity of the data.

6 CONCLUSION

In this paper, we proposed a novel deep multi-task spatial-temporal network model coupled with adversarial learning to simultaneously predict the crowd flows and flow OD. By adopting a shared-private feature learning framework, common features shared by both tasks are effectively extracted through an adversarial shared feature learning model. To further decompose shared and private features, a discriminative loss on task classification based on task-specific features and the orthogonality constraint between shared and private features are also incorporated. Considering the complex spatial-temporal correlations on the crowd flows, the proposed spatial-temporal network utilized a designed HSTN and an attentive temporal queue for effective spatial-temporal correlations capturing. Extensive evaluations on two real large datasets showed that the proposed model mutually enhanced the performance of both tasks, and also outperformed state-of-the-art single/multi-task learning models for crowd flow prediction. In the future, it would be interesting to further study whether the proposed adversarial multi-task learning framework can be applied to other prediction tasks such as traffic congestion prediction and traffic accident detection.

ACKNOWLEDGMENTS

This work is supported by National Key R&D Program of China 2018YFB1003900, NSF of Jiangsu Province BK20171420 and CCF-Tencent Open Research Fund.

REFERENCES

[1] M. Bierlaire and F. Crittin. 2004. An Efficient Algorithm for Real-Time Estimation and Prediction of Dynamic OD Tables. *Operations Research* 52, 1 (2004).

[2] Mecit Cetin and Gurcan Comert. 2006. Short-Term Traffic Flow Prediction with Regime Switching Models. *Transportation Research Record: Journal of the Transportation Research Board* 1965 (2006).

[3] J. Chen, X. Xu, Y. Wu, and H. zheng. 2018. GC-LSTM: Graph Convolution Embedded LSTM for Dynamic Link Prediction. *arxiv* (2018).

[4] K. Chu, A. Y. S. Lam, and V. O. K. Li. 2019. Deep Multi-Scale Convolutional LSTM Network for Travel Demand and Origin-Destination Predictions. *IEEE Transactions on Intelligent Transportation Systems* (2019), 1–14.

[5] Bowen Du, Hao Peng, Senzhang Wang, Md Zakirul Alam Bhuiyan, Lihong Wang, Qiran Gong, Lin Liu, and Jing Li. 2019. Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction. *IEEE Transactions on Intelligent Transportation Systems* 21, 3 (2019), 972–985.

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-Adversarial Training of Neural Networks. *Machine learning Research* 17 (2016), 1–35.

[7] Ian J. Goodfellow, Jean Pouget-Abadiey, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozairz, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Proceedings of NeurIPS*.

[8] Aude Hofleitner, Ryan Herring, and Pieter Abbeel. 2012. Learning the Dynamics of Arterial Traffic From Probe Data Using a Dynamic Bayesian Network. *IEEE Transactions on Intelligent Transportation Systems* 13, 4 (2012), 1679–1693.

[9] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of ICLR*.

[10] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2019. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *In Proceedings of ICLR*.

[11] L. Liu, Z. Qiu, G. Li, Q. Wang, W. Ouyang, and L. Lin. 2019. Contextualized Spatial–Temporal Network for Taxi Origin-Destination Demand Prediction. *IEEE Transactions on Intelligent Transportation Systems* 20, 10 (2019), 3875–3887.

[12] P. Liu, X. Qiu, and X. Huang. 2017. Adversarial Multi-task Learning for Text Classification. In *Proceedings of ACL*.

[13] Dai Quoc Nguyen, Dat Quoc Nguyen, Cuong Xuan Chu, Stefan Thater, and Manfred Pinkal. 2017. Sequence to Sequence Learning for Event Prediciton. In *Proceedings of ACL*.

[14] J. Ren and Q. Xie. 2017. Efficient OD Trip Matrix Prediction Based on Tensor Decomposition. In *Proceedings of MDM*. 180–185.

[15] W. R.Tobler. 1970. A computer movie simulating urban growth in the detroit region. *Economic geography* 46, 1 (1970), 234–240.

[16] Xingjian Shi, 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.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Proceedings of NeurIPS*. 1–11.

[18] Senzhang Wang, Jiannong Cao, Hao Chen, Hao Peng, and Zhiqiu Huang. 2020. SeqST-GAN: Seq2Seq Generative Adversarial Nets for Multi-step Urban Crowd Flow Prediction. *ACM Transactions on Spatial Algorithms and Systems (TSAS)* 6, 4 (2020), 1–24.

[19] Senzhang Wang, Jiannong Cao, and Philip S Yu. 2019. Deep learning for spatio-temporal data mining: A survey. *arXiv preprint arXiv:1906.04928* (2019).

[20] Senzhang Wang, Xiaoming Zhang, Jianping Cao, Lifang He, Leon Stenneth, Philip S. Yu, Zhoujun Li, and Zhiqiu Huang. 2017. Computing Urban Traffic Congestions by Incorporating Sparse GPS Probe Data and Social Media Data. *ACM Trans. Inf. Syst.* 35, 4 (2017), 40:1–40:30.

[21] Yuandong Wang, Hongzhi Yin, Hongxu Chen, Tianyu Wo, Jie Xu, and Kai Zheng. 2019. Origin-Destination Matrix Prediction via Graph Convolution: A New Perspective of Passenger Demand Modeling. In *Proceedings of KDD*. 1227–1235.

[22] Billy Williams. 2001. Multivariate Vehicular Traffic Flow Prediction: Evaluation of ARIMAX Modeling. *Transportation Research Record: Journal of the Transportation Research Board* 1776 (2001).

[23] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *Proceedings of NAACL*.

[24] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2019. Revisiting Spatial-Temporal Similarity: A Deep Learning Framework for Traffic Prediction. In *Proceedings of AAAI*.

[25] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. 2018. Deep Multi-View Spatial-Temporal Network for Taxi Demand Prediction. In *Proceedings of AAAI*.

[26] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. In *Proceedings of AAAI*.

[27] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiqing Zhang, and Xiuwen Yi. 2016. DNN-based prediction model for spatio-temporal data. In *Proceedings of ACM SIGSPATIAL GIS*.

[28] J. Zhang, Y. Zheng, J. Sun, and D. Qi. 2020. Flow Prediction in Spatio-Temporal Networks Based on Multitask Deep Learning. *IEEE Transactions on Knowledge and Data Engineering* 32, 3 (2020), 468–478.

[29] Yuxuan Zhang, Senzhang Wang, Bing Chen, Jiannong Cao, and Zhiqiu Huang. 2019. Trafficgan: Network-scale deep traffic prediction with generative adversarial nets. *IEEE Transactions on Intelligent Transportation Systems* (2019).

[30] Xian Zhou, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. 2018. Predicting Multi-step Citywide Passenger Demands Using Attention-based Neural Networks. In *Proceedings of WSDM*.