MBA-STNet: Bayes-enhanced Discriminative Multi-task Learning for Flow Prediction

Hao Miao*, Jiaxing Shen*, Jiannong Cao, *Fellow, IEEE*, Jiangnan Xia, and Senzhang Wang†

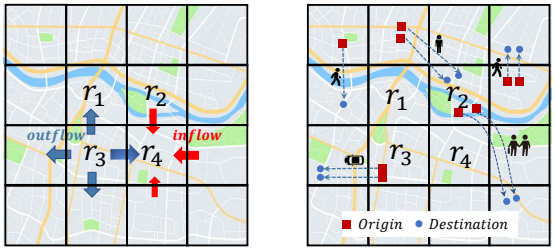
Abstract—Crowd flow prediction, which aims to predict the in/out flows of different areas of a city, plays a critically important role in many real-world applications including intelligent transportation systems and public safety. The challenges of this problem lie in both the dynamic mobility patterns 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 Bayes-enhanced Adversarial Spatial Temporal Network entitled MBA-STNet to effectively address it. MBA-STNet 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 discriminative shared features, an adversarial loss on shared feature extraction is incorporated to reduce information redundancy. A Bayesian heterogeneous Spatio-temporal Attention Network is designed to learn the complex spatio-temporal correlations and alleviate the problem of data uncertainty. We also design an attentive temporal queue to capture the complex temporal dependency automatically without the help of domain knowledge. Extensive evaluations are conducted over the bike and taxicab trip datasets in New York. The results demonstrate that the proposed MBA-STNet is superior to state-of-the-art methods.

Index Terms—Flow Prediction, Multi-task Learning, Bayesian Neural Network, Adversarial Learning.

1 Introduction

Crowd flows including in- and out- flows reflect the human mobility dynamics in different areas of a city, as shown in Fig. 1(a). Such flows can be measured by various human mobility data such as taxi trajectories, shared bike trips, and subway check-in/out records. Crowd flow prediction, which refers to predicting the sizes of crowds entering or leaving a particular region of a city, plays a vital role in various urban computing applications and has attracted increasing research interest recently [1], [2], [3]. For example, an accurate forecast of the traffic flow can facilitate more effective traffic management and derive better travel route planning [4].

Due to the significance of crowd flow prediction, a considerable research effort has been devoted to developing effective prediction models [4], [5], [6]. 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 [3], [7], [8]. However, many real-world applications not only care about how many people will enter a region but also need to know where the crowds come from, namely the Origin-Destination (OD) of the flows (Fig. 1(b)). As illustrated in Fig. 1, the crowd flows are highly correlated to flow OD. It is obvious that the inflow of a region is equal to the sum of OD flows ending in that region, and the outflow of a region is equal to the sum of flow OD starting from that region. OD prediction aims to predict the number of passengers from one region to another [9], [10]. Although they are highly correlated to each other, most existing works conduct crowd prediction and flow OD prediction tasks separately without considering their mutual influence. 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.



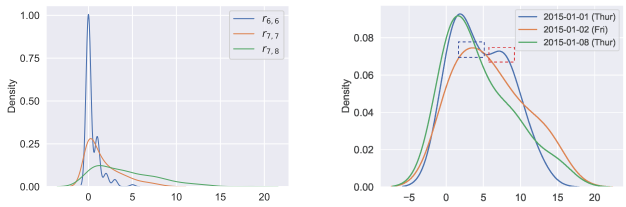
(a) Crowd Flow (b) Flow OD

Fig. 1. Illustration of crowd flow and flow OD, where each grid denotes a region.

However, this problem is non-trivial to address due to the following challenges. 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

- *Hao Miao is with the Department of Computer Science, Aalborg University, and with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China. E-mail: haom@cs.aau.dk*
- *Jiaxing Shen, Jiannong Cao are with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China. E-mail: {jiaxshen, jiannong.cao}@polyu.edu.hk*
- *Jiangnan Xia and Senzhang Wang are with School of Computer Science and Engineering, Central South University, Changsha, China. E-mail: xjn1374545699@csu.edu.cn, szwang@csu.edu.cn*

∗: equal contribution; †: corresponding author.



(a) Spatial uncertainty (b) Temporal uncertainty

Fig. 2. Examples of spatial and temporal uncertainty in crowd flow distribution, where the distributions are generated by kernel density estimation (KDE) [11].

learning. Although some multi-task models are proposed [12], they still suffer from the issue that the learned common features and task-specific features are somewhat blended, leading to information redundancy. In addition, data uncertainties, including spatial uncertainty and temporal uncertainty, are common in our flow data. An example of the uncertainty in terms of crowd flow distributions in New York is illustrated in Fig. 2. Fig. 2 (a) reveals that the spatial distributions in different regions (i.e., $r_{6,6}, r_{7,7}, r_{7,8}$ on January 1, 2015) are different, even they are geographically close. Fig. 2 (b) shows the temporal uncertainty through analyzing the data distributions in different days (i.e., Jan. 1, Jan. 2 and Jan. 8, 2015) of the region $r_{9,9}$. The data distribution on Jan 1 (Thursday) is similar to that on Jan 8 (Thursday), but there is a data fluctuation which is marked with a red rectangle. Moreover, the blue rectangle shows there is a distribution shift between the data on Jan. 1 and Jan. 2. The data distribution fluctuation and shift both show the existence of temporal uncertainty. Such data uncertainties are not considered in existing multi-task learning methods, which will lead to model performance degrade [13], [14]. Besides, the spatial correlations of the crowd flows are complex, and sometimes do not follow the spatial smoothness [1], [15]. With the development of public transportation, the First Law of Geography: "near things are more related than distant things" may not fully reflect the spatial correlations of the crowd flows in urban areas. It is necessary to incorporate the semantic spatial correlations. Finally, 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.

To tackle the aforementioned challenges, we propose a Multi-task Bayes-enhanced Adversarial Spatio-Temporal Network entitled MBA-STNet to predict the crowd flows and OD of the flows simultaneously. To better capture the spatial correlations of the flow data, we propose to model the raw flow trajectories as two types of data representations, semantic spatial-temporal graphs and flow images. The semantic spatial-temporal graph is built adaptively following the graph construction rules in AGCRN [16]. Inspired by bayesian neural networks [13], a Bayes-enhanced Heterogeneous Spatial-Temporal Attention Net (BHSTAN) is proposed to first encode the graphs and images separately, and then integrates them together for the aim of overcoming the problem of data uncertainty, and fuse the feature learning on semantic spatial-temporal graphs and flow images. We jointly learn features of the two tasks under a shared-private framework which generally 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 space could contain some task-specific features, while some sharable features could also be mixed in private spaces [17], we propose to add an adversarial loss for shared features and an orthogonality loss for private 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 and OD of the flows in a long period (several weeks), from which the most useful ones for future prediction are attentively selected.

To summarize, our main contributions are as follows.

- We propose a novel bayes-enhanced 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 orthogonality loss are integrated into the multi-task learning framework to reduce feature redundancy, so that features in different latent spaces are more distinguishable.
- A Bayes-enhanced heterogeneous spatial-temporal attention net is proposed to integrate the local and global spatial features from the flow images and semantic spatial temporal graphs, and alleviate the influence of data uncertainty. To automatically capture the complex temporal correlations, an attentive temporal queue is designed to select the most relevant historical data representations for prediction.

The proposed MBA-STNet is an extended version of MT-ASTN [15] which was proposed in our earlier paper published at CIKM2020. In a nutshell, We replace Heterogeneous Spatio-Temporal Network in MT-ASTN with a newly designed Bayesian Heterogeneous Spatio-Temporal Attention Net in order to fuse feature learning of graphs and images and relieve the influence of data uncertainty. The construction of semantic graphs in [15] is labor-intensive, we improve it by constructing an adaptive semantic spatio-temporal graph for global spatial dependencies capturing, which is capable of reflecting the global mobility patterns of crowds with the training of the model. We also re-conduct most of the experiments and add several experiments to demonstrate the superiority of MBA-STNet.

The remainder of this paper is organized as follows. Section 2 will review related works. Section 3 will give some important notations and a formal problem definition. Section 4 will show the model framework and introduce our

methodology. Evaluations are given in Section 5. Finally, this paper is concluded in Section 6.

2 Related Work

This work is highly relevant to the research topics of crowd flow prediction and OD prediction. Next, we will review related works from the two aspects.

2.1 Crowd Flow Prediction

Crowd flow prediction, which has attracted rising interest due to the increasingly available urban data and rich applications, has been studied for many years [3], [6], [18]. Traditional crowd flow prediction models are mostly statistics-based approaches such as ARIMA [19], [20] and SVR [21]. The major limitation of the above statistics-based traffic flow prediction models is that the complex temporal and spatial correlations of spatio-temporal data are hard to be captured due to their limited learning capacity.

With the advances of deep learning techniques, various deep neural network models are broadly applied and have achieved much better performance than traditional statistics-based shallow models, such as ST-ResNet [3], ConvLSTM [7] and DCRNN [8]. Wang et al. [2] provided a comprehensive review of recent progress in applying deep learning techniques for spatio-temporal data mining. A line of studies applied CNN to capture the spatial correlation by treating the traffic flow data of the entire city as images. Another line of research is combining CNN model and RNN model to capture both spatial and temporal correlations [22]. Yao et al. [1] proposed a Spatial-Temporal Dynamic Network (STDN) model for road network based flow prediction. Cheng et al. [23] Proposed the DeepTransport model which combined CNN and RNN to capture the spatial temporal traffic data within a transport network.

Recently, more and more studies focus on using graph neural networks [24] for spatio-temporal data prediction [8], [25], [26]. Li et al [8] proposed the Diffusion Convolutional Recurrent Neural Network (DCRNN) to model the traffic flow as a diffusion process on a directed road graph, which is a deep learning framework for traffic flow forecasting. Yu et al. [27] proposed STGCN model, which applied ChebNet graph convolution and 1D convolution to extract spatial dependencies and temporal correlations. Bai et al. [16] proposed an adaptive graph convolution recursive network (AGCRN), which used node adaptive parameter learning and data adaptive graph generation modules to enhance the traditional graph convolution network to automatically capture the fine-grained spatio-temporal correlation in traffic sequences.

Even though the above mentioned methods have achieved great performance, these works consider crowd flow prediction as a single task, but ignore the impact of flow OD.

2.2 OD Prediction

Origin-Destination (OD) prediction [28], [29], [30] can benefit many real applications such as ride-hailing services and traffic management. Traditional methods mostly used regression based approaches or statistic-based approaches to predict or estimate the dynamic vehicle OD matrix in a transportation network [31]. 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. [10] proposed Grid-Embedding based Multi-task Learning model namely GEML to predict the number of passengers from one region to another. Liu et al. [9] 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. [29] developed a deep learning model called multi-scale convolutional long short-term memory network (MultiConvLSTM) to predict the future travel demand and the OD flows. Zhang el at. [32] proposed a channel-wise attentive split-convolutional neural network for OD flow forecasting, which explicitly considers the unique characteristics of the urban rail transit system.

However, most existing works consider OD flow prediction and crowd flow prediction as two separate problems and ignore the high correlation between them. Although Zhang et al. [12] 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. MDL simply concatenates the features of different tasks without distinguish 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 region as $R = \{r_{1,1}, \ldots, r_{i,j}, \ldots, r_{m,n}\}$, where $r_{i,j}$ represents the $i\text{-}th$ row and $j\text{-}th$ column cell region of the grid map.

Definition 2. **Corwd 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:

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

where $T_r : g_1 \rightarrow g_2 \rightarrow ... \rightarrow g_{T_r}$ is a trajectory at time slot t 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 [3], we denote the inflow and outflow of all the cell regions in t as a crowd flow image $\mathcal{X}^t \in \mathcal{R}^{m \times n \times 2}$, 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

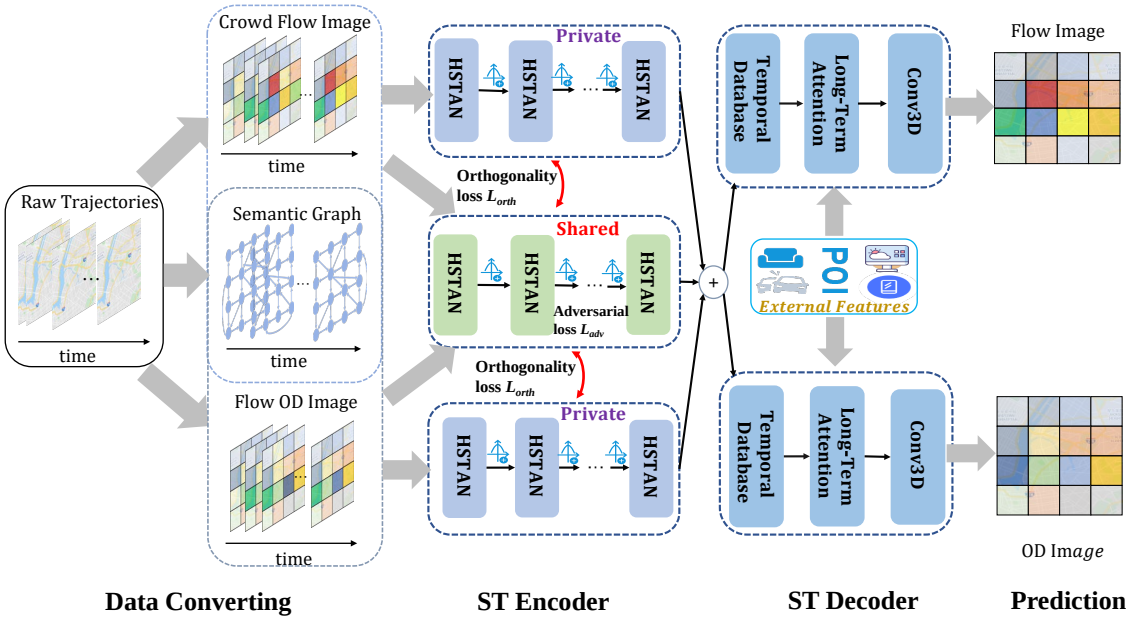


Fig. 3. Framework of proposed MBA-STNet model, which contains four parts: Data Converting, ST Encoder, ST Decoder, and Prediction.

sizes of flows starting from i-th cell region and ending at j-th cell region of R.

Definition 4. **Flow OD image** Given the flow OD matrix $D^t \in \mathcal{R}^{N\times N}$ at time slot t, we construct the flow OD image $\mathcal{M}^t \in \mathcal{R}^{m\times n\times N}$ with $N = m \times n$ channels. Each channel $\mathcal{M}^t(:,:,i)$ denotes the size of trajectories staring 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

Fig. 3 shows the framework of the proposed MBA-STNet, which consists of four major steps: Data Converting, Spatio-Temporal (ST) Encoder, Spatio-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 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 fig. 3, the designed ST encoder consists of a shared encoder, and two private encoders for two tasks respectively. The ST encoder contains several stacked Bayesian Heterogeneous Spatial-Temporal Attention Net (BHSTAN) layers. We first design Heterogeneous Spatio-Temporal Attention Net (HSTAN) to fuse feature learning of flow/OD images and semantic ST graphs, which will be introduced in Section 4.2 in detail. For overcoming the problem of data uncertainty, we incorporate the idea of bayesian neural network into HSTAN and further propose BHSTAN layers whose parameters conform to a specific distribution (e.g., Gaussian Distribution) to make the model more general and robust. Specifically, we assume the distribution of input data conforms to a specific distribution, and then train the proposed BHSTAN with parameters multiply sampled from this distribution. In order to extract pure shared features of the two tasks, we propose to use adversarial learning to train the shared ST encoder. An orthogonality loss L_{orth} is also proposed, which aims to prevent the extracted private features from mixing with common features. We will elaborate this step in Section 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, long-term 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 long-term attention mechanism 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 the Definitions 2-4, we need to first convert the raw flow trajectories $\mathcal{P}$ to crowd flow images, flow OD images and semantic ST graphs. Following [3], [12], we first model

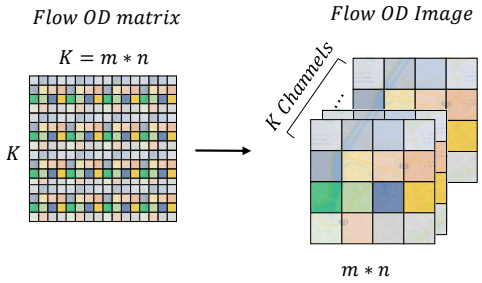


Fig. 4. Converting flow OD matrix to Flow OD image.

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 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 Fig. 4. The reason 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 the adaptive semantic ST graphs $G^t = \{V, A^t\}$, inspired by AGCRN and GraphWaveNet [16], [33]. The nodes V are the cell regions, and the adaptive adjacency A^t represents the similarity between each pair of region nodes. To achieve the adaptive adjacency, we first randomly initialize a learnable node embedding: $E_A \in \mathcal{R}^{N \times d_e}$ for all nodes, where d_e denotes the dimension of node embedding, and each row of E_A represents the embedding of a specific node. Then we can infer the semantic spatial dependencies between each pair of nodes by multiplying E_A and E_A^T as follows:

$$A = Softmax(ReLU(E_A E_A^T)) \tag{2}$$

where $Softmax$ is used to normalize the adaptive matrix. Note that the adaptive semantic adjacency matrix changes over time. This process is similar as constructing the graph based on nodes similarity.

4.2 Heterogeneous Spatial-Temporal Attention Net

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 Attention network (HSTAN), as shown in Fig. 5 to learn the data representations of images and graphs separately, and then fuse them together. In addition, to address the challenge of data uncertainty, we follow the idea of bayesian deep learning methods and incorporate it into HSTAN.

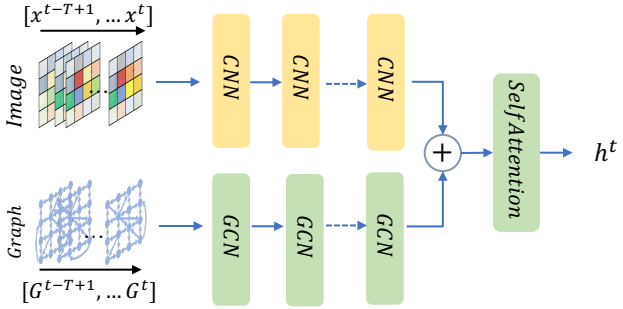


Fig. 5. Illustration of the proposed HSTAN.

Given a specific time slot t, HSTAN adopts Conv2D layers to learn the latent representation h^t_{image} for crowd flow and flow OD images, and stacked GCN layers to learn the latent representation h^t_G for the semantic ST graph. Here, we use 2-dimensional convolutions on the tensors of crowd flow images and flow OD images to capture the local spatial correlations. To more broadly capture the spatial correlations, we construct a hierarchical semantic ST graph representation learning with GCN. The two types of data representation are integrated by the following formula

$$h^t_{int} = h^t_{image} \oplus h^t_G \tag{3}$$

where $\oplus$ is the feature addition operation across channels. Finally, to capture the temporal correlations, we input the integrated representations over T time slots to the multi-head self-attention mechanism as follows.

$$[h^{t-T+1}, \cdots, h^t] = MultiHeadAttn([h^{t-T+1}_{int}, \cdots, h^t_{int}]) \tag{4}$$

where $MultiHeadAttn$ represents multi-head self-attention mechanism.

Finally, to solve the problem of data uncertainty, we construct bayes-enhanced heterogeneous spatial-temporal attention net (BHSTAN) by borrowing the idea of bayesian neural network which considers the weights of the CNN, GCN and self-attention mechanism conform to a specific distribution. The formula is as follows.

$$h^t = BHSTAN(\mathcal{X}^t, \mathcal{G}^t, W_{bayes}) \tag{5}$$

where W_{bayes} represents learnable parameters.

4.2.1 Learning Spatial Dependencies with CNN and GCN

To fully capture spatial dependencies of crowd flow and flow OD data, especially local and global spatial correlations, we employ 2-dimensional convolutional neural network (CNN) [34] and Graph Convolutional network (GCN) [24].

We first model the crowd flow and flow OD in city regions as "images", that the modeling process can be seen from the definition 2 and definition 4. When we get the

"images", the local spatial correlations can be learned by CNN. The corresponding formulas are as follows:

$$\begin{aligned} h^t_{image,flow} &= CNN(\mathcal{X}^t, W_{flow}) \\ h^t_{image,OD} &= CNN(\mathcal{M}^t, W_{OD}) \end{aligned} \tag{6}$$

where $h^t_{image,flow}$ and $h^t_{image,OD}$ represent latent representations of crowd flow image and flow OD image at time slot t respectively, $CNN(\cdot)$ represents the CNN operation, W_{flow}, W_{OD} are learnable parameters.

However, CNN is not enough for full spatial feature learning, especially learning global spatial correlations. To solve this problem, we create semantic spatio-temporal graphs and then use spectral convolutions [24] on the constructed semantic ST graphs, which can be formulated as follows:

$$h^t_G = f_\odot(N^t, A^t) = \sigma(D^{-\frac{1}{2}}\widetilde{A^t}D^{-\frac{1}{2}}N^tW^t) \tag{7}$$

where $f_\odot(\cdot)$ is GCN operation, N^t and A^t are node embeddings and adaptive adjacency of graph G^t, $\widetilde{A^t}$ is A^t with added self-connections, $D_{ii} = \sum_j \widetilde{A^t_{ij}}$ is the degree matrix, and W^t represents the learnable weight matrix. Specifically, the learned $i\text{-}th$ global spatial hidden features can be formulated as follows:

$$h^t_{G,l} = \sigma(D^{\frac{1}{2}}\widetilde{A^t}D^{\frac{1}{2}}h^t_{G,l-1}W^t_l) \tag{8}$$

where $h_{G,0}$ is equal to $\mathcal{X}^t$, W^t_L is the trainable matrix of filter parameters in the $l\text{-}th$ graph convolutional layer.

4.2.2 Learning Short-term Temporal Dependencies with Self-attention Mechanism

Besides the spatial correlations, flow forecasting also involves complex temporal dependencies [3], [35]. We input the extracted spatial features into Multi-head Self-Attention mechanism [36] as follows for short-term temporal feature learning. There are three important components: *Query(Q)*, *Key(K)* and *Value(V)*, which are all aforementioned learned spatial representations h^t_{int}. This attention mechanism is compoued by concatenating the output matrix of each attention head and projecting it by W:

$$\begin{aligned} &MultiHeadAttn(\cdot) = Concat(head1, \cdots, headi) * W \\ &headi = Attention(QW^Q_i, KW^K_i, VW^V_i) \\ &Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}}) * V \end{aligned} \tag{9}$$

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.

4.2.3 Bayes-enhanced Heterogeneous Spatial-Temporal Network

To alleviate the problem of data uncertainty and make our framework more robust, we incorporate the idea of Bayesian Deep Learning [13], [14] into ST Encoder, especially the HSTAN. In the traditional deep learning models, the weights are always fixed which are randomly initialized in the beginning of the model training, and makes the model very sensitive to the input data. In this paper, the raw flow trajectories data were collected by location-based devices, such as GPS. It is inevitable that such devices sometimes fail in some extreme cases, such as sensor failure and poor network signal, which will result in collecting uncertain data. If we train the model with these uncertain data, the model performance will be degraded. Bayesian deep learning methods (BNNs) assume that weight of each layer is not fixed but conforms to a distribution and then sample weights from the distribution with the model training, which will make the model more robust.

From the point of probability theory view, a traditional neural network (e.g., CNN, GCN) with one or multiple layers is a probability model $Pr(y|x, w)$, where w represents the collection of all weights of all layers. Under the traditional way of inference, the training of fixed value w follows the maximum likelihood estimation with the training set $\mathcal{D} = \{y_i|x_i\}$, i.e.,

$$\begin{aligned} w^* &= argmax_w logPr(D|w) \\ &= argmax_w \sum_i logPr(y_i|x_i, w) \end{aligned} \tag{10}$$

If we introduce regularization term to avoid overfitting, the optimal parameters follow the maximum a posterior probability [37]:

$$\begin{aligned} w^* &= argmaxx_w logPr(\mathcal{D}|w) + logPr(w) \\ &= argmax_w logPr(w|\mathcal{D}) \end{aligned} \tag{11}$$

But when we consider the parameters of neural network layers following the posterior distributions embedded in the training set, the probabilistic model can exploit data uncertainties and estimate distributions based on Bayesian inference [38].

Following previous work [13], [39], we modify the neural network (i.e., CNN, GCN and Self-attention mechanism) in a Bayesian way, denoted generally by $y = f_w(x)$ where parameters actually follow the posterior distributions of the training set, to cope with data uncertainties. We use zero-mean Gaussian as the prior distribution over the parameter space $Pr(w)$ which can bring about the benefit of regularization [40], [41]. Although it is still an open problem to select the prior for the Bayesian neural network, previous works [40], [41], [42] demonstrated that the standard normal distributions is a suitable choice and has achieved promising performance in many tasks. Therefore, we use standard normal distribution as our prior. Note that we employ the posterior distribution to generate samples instead of prior distribution. According to Bayes theorem, the posterior distribution can be obtained by

$$Pr(w|\mathcal{D}) = \frac{Pr(\mathcal{D}|w)Pr(w)}{Pr(\mathcal{D})} \tag{12}$$

Nonetheless, equation 12 is intractable due to the high computation complexity. To overcome this problem, we employ variational inference to approximate the posterior $Pr(w|\mathcal{D})$ through a variational distribution $q(w|\Theta)$ parameterized by θ through minimizing the Kullback-Leibler (KL) divergence between $q(w|\Theta)$ and $Pr(w|\mathcal{D})$.

$$\begin{aligned} \theta^* &= argmin_\theta KL[q(w|\theta)||Pr(w|\mathcal{D})] \\ &= argmin_\theta \int q(w|\theta) log \frac{q(w|\theta)}{Pr(w)Pr(\mathcal{D}|w)} dw \\ &= argmin_\theta KL[q(w|\theta)||Pr(w)] - \mathbb{E}_{q(w|\theta)}[logPr(\mathcal{D}|w)] \end{aligned} \tag{13}$$

We denote KL Loss L_{KL} as follows:

$$L_{KL} = KL[q(w|\theta)||Pr(w)] - \mathbb{E}_{q(w|\theta)}[logPr(\mathcal{D}|w)] \quad (14)$$

Borrowing the idea of Monte Carlo optimization method, the KL Loss L_{KL} could be further written as:

$$L_{KL} = \sum_{i=1}^{n} logq(w^i|\theta) - logPr(w^i) - logPr(\mathcal{D}|w^i) \quad (15)$$

where w^i is the sampled weight of the i-th input data. The full related theoretical analysis could be seen in [13], [43].

To summarize, Bayesian neural network samples weights from a trained distribution $\mathcal{N}(\mu, \sigma\sigma^T)$ to alleviate the problem of data uncertainty. We assume the weights of CNN, GCN and self-attention mechanism conform to a specific distribution, and then convert to Bayesian CNN (BCNN), Bayesian GCN (BGCN), and Bayesian Self-attention Mechanism (BAttention). And then construct a Bayesian Heterogeneous Spatial-Temporal Attention Net (BHSTAN) which will be used in Spatio-temporal Encoder. The entire process can be expressed as:

$$\begin{aligned}
&h^t_{image} = BCNN(\mathcal{X}^t_{image}, W_{image}) \\
&h^t_G = BGCN(\mathcal{X}^t_G, W_G) \\
&h^t_G : \mathcal{R}^{B\times 1\times N\times C} \rightarrow \mathcal{R}^{B\times 1\times m\times n\times C} \\
&h^t_{int} = h^t_{image} \oplus h^t_G \\
&[h^{t-T+1}, \cdots, h^t] = BAttention([h^{t-T+1}_{int}, \cdots, h^t_{int}])
\end{aligned} \quad (16)$$

where B represents batch size, N is equal to $m \times n$.

4.3 Spatio-Temporal (ST) Encoder

The crowd flow images, flow OD images and the semantic graphs are input into ST Encoder. ST Encoder contains a shared ST Encoder for shared feature learning, and two private ST Encoders for task-specific features learning of the two tasks. 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 adopt above-mentioned BHSTAN layers to first learn the data representations of images and graphs separately and then fuse them together as shown in Fig. 5. Each encoder consists of stacked BHSTAN layers.

Shared ST encoder aims to learn the common features that are shared by all the tasks. Inspired by the work [44], we also adopt adversarial learning to help the shared ST encoder extract pure shared features to reduce information redundancy. The purpose of the two private encoders is to learn task-specific features of the two tasks. After learning the shared features and task-specific private features, orthogonality constraint upon them is incorporated to make them more separable.

Adversarial loss L_{adv} Generative Adversarial Networks (*GANs*) [45] 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

$$\min_G \max_D (E_{x\sim P_{data}}[logD(x)] + E_{z\sim p(z)}[log(1 - D(G(z)))]) \quad (17)$$

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.

$$L_{adv} = \frac{1}{\mathcal{L}} \min_{\theta_s} \left(\max_{\theta_D} \left(\sum_{i=1}^{\mathcal{L}} \sum_{j=1}^{m} y^{i,j} logDis\left(h^{i,j}_{shared}\right) \right) \right) \quad (18)$$

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

$$L_{orth} = \sum_{i=1}^{m} ||H^T_{i,shared} H_{i,private}||^2_F \quad (19)$$

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 Spatio-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 Fig. 3, 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.

$$h^t_{all} = h^t_{shared} + h^t_{private} \quad (20)$$

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.

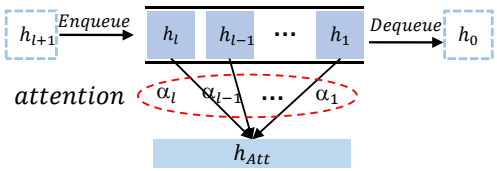


Fig. 6. Illustration of attentive temporal queue.

4.4.1 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. Fig. 6 shows the architecture of attentive 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.

4.4.2 Conditional Multi-head Self-Attention

Here we adopt the *multi-head self-attention*, which is a famous attention mechanisms used in Transformer [36]. 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 attend 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 output matrix of each attention head and projecting it by W:

$$\begin{aligned} MultiHead &= Concat(head_1, \dots, head_i) * W \\ head_i &= Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \end{aligned} \tag{21}$$

where W is the parameter matrix, W_i^Q, W_i^K, W_i^V are the linear transformation parameters of the *query*, *key*, and *value* respectively. Two fully-connected layers are stacked to learn representation of external features upon e^t, where e^t is external features at time slot t. We can view the first layer followed by an activation function as an embedding layer. The second layer is used to map low to high dimensions that make the shape of learned features E^t the same as h^t. At each *conditional attention head*, the hidden state with external information stored in temporal queue $\mathcal{H}_i = \left([h^{t-l+1}, E^{t-l+1}], \dots, [h^t, E^t]\right)$ where E^t is the external features in time slot t and $[,]$ denotes the 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:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}}) * V. \tag{22}$$

Algorithm 1 Multi-task Bayes-enhanced Adversarial Spatio-Temporal Attention Network

Input: $\{\mathcal{X}^t, \mathcal{M}^t | t = 1, \dots, T\}$: Crowd flow and flow OD images; E: External features;
Output: MBA-STNet model

$\mathcal{D}_{train} \rightarrow \emptyset$
for $t \in \mathcal{T}$ **do** *//$\mathcal{T}$ is available time set*
  put an training instance $(\{\mathcal{X}^t, \mathcal{M}^t, E^t | t \in [t, t+T)\}, \{\mathcal{X}^{t+1}, \mathcal{M}^{t+1}\})$ into $\mathcal{D}_{train}$ *// T is the length of time slot*
end for
while *not converge* **do**
  Sequentially select a batch of instances $\mathcal{D}_{batch}$ from $\mathcal{D}_{train}$
  $S_{item} \leftarrow 0$
  for $S_{item} < S_{MAX}$ **do** *// S_{MAX} is the number of bayes sampling*
    Sample weights θ from a specific Gaussian distribution
    $h^t_{private,flow} \leftarrow$ Flow private feature learning by $FlowEncoder(\cdot)$
    $h^t_{private,OD} \leftarrow$ OD private feature learning by $ODEncoder(\cdot)$
    $h^t_{shared} \leftarrow$ Flow and OD Shared feature learning by $SharedEncoder(\cdot)$
    $h^t_{all,flow}, h^t_{all,OD} \leftarrow$ integrated flow and OD feature learning by Eq.20
    Store historical learned feature into temporal Queue
    $H_{flow}, H_{OD} \leftarrow$ Flow and OD long-term temporal feature learning by conditional multi-head attention
    $\hat{\mathcal{X}^{t+1}} \leftarrow Conv3D(H_{flow})$
    $\hat{\mathcal{M}^{t+1}} \leftarrow Conv3D(H_{OD})$
    Update θ based on Eq. 24.
  end for
  Return the learned MBA-STNet model
end while

4.5 Overall Objective Function

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

$$L_{task} = \frac{1}{m}\sum_{i=1}^{m}\sqrt{\frac{1}{\mathcal{L}}\sum_{j=1}^{\mathcal{L}}||\hat{Y}^{i,j} - Y^{i,j}||^2} \tag{23}$$

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 contains four parts: prediction loss of the two tasks L_{task}, KL loss L_{KL}, adversarial loss L_{adv},

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.	

and orthogonality constraint loss L_{orth}. We combine them together and the overall loss function is as follows.

$$L_{all} = L_{task} + L_{adv} + L_{orth} + \gamma * L_{KL} \tag{24}$$

where γ is set to 0.01 in this paper. In addition, the network model is trained with backpropagation and the adversarial training is optimized via gradient reversal layers [44]. The pseudo-code of the algorithm is shown in Algorithm 1, where $FlowEncoder(\cdot)$, $ODEncoder(\cdot)$ and $SharedEncoder(\cdot)$ represent Crowd flow ST Encoder, Flow OD ST Encoder and shared ST Encoder, respectively. The time complexity of this algorithm is $O(nlogN)$.

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** [7] 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 [3]** 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 [1]** 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 [10]** 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 [12]** MDL is a multi-task learning framework for predicting both the node flows and edge flows on a spatial-temporal network.
- **MT-ASTN [15]** MT-ASTN is a recent state-of-the-art multi-task adversarial spatial-temporal network model to predict crowd flow and flow OD simultaneously.

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

- MBA-STNet (L_{bayes}) This model removes the KL loss for Bayesian deep learning. Through comparing with it, we test whether the incorporation of Bayesian assumption is helpful to defeat the influence of uncertainty data and improve the prediction performance.
- MBA-STNet (L_{adv}) This model drops 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.
- MBA-STNet (Gra) This model removes 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.
- MBA-STNet (Que) This model removes the temporal queue from the ST decoder. Attention is only applied on the input data sequence. Through comparing with

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**
MBA-STNet	**2.905**	**0.067**	**7.157**	**0.082**	2.227	0.061	**5.976**	0.059

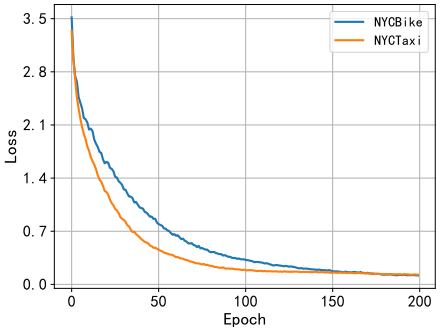
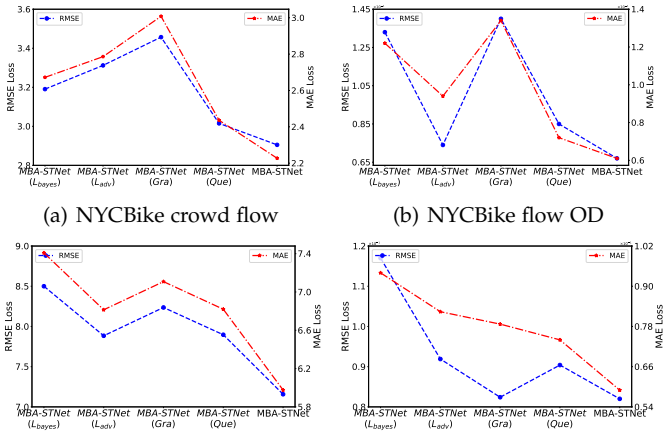


Fig. 7. Loss curve of MBA-STNet on the two datasets.



(a) NYCBike crowd flow (b) NYCBike flow OD

(c) NYCTaxi crowd flow (d) NYCTaxi flow OD

Fig. 8. RMSE and MAE comparison with variant methods.

it, we test whether the temporal queue is useful for our model to capture the complex temporal correlations of the crowd flows.

5.1.3 Implementation details

We implement our model with Pytorch framework on NVIDIA Quadro RTX 8000 GPU. The model parameters are set as follows. The data size of crowd flow images is $6 \times 16 \times 16 \times 2$ 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 $6 \times 16 \times 16 \times 256$, where 256 is the number of channels which is also the number of cell regions. The learning rate and batch size are set to 0.000001 and 32, respectively. The CNN and AGCN in HSTN model of ST Encoder contain 3 layers whose structure is $6 \times 16 \times 16 \times 16$, $6 \times 16 \times 16 \times 32$ and $6 \times 16 \times 16 \times 64$. We use one layer Conv3D in ST Decoder for crowd flow image prediction, whose structure is $1 \times 16 \times 16 \times 2$. The structure of Conv3D in ST Decoder for flow OD prediction is $1 \times 16 \times 16 \times 256$. 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:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|\hat{\mathcal{X}}^{t+1} - \mathcal{X}^{t+1}|$$
$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}||\hat{\mathcal{X}}^{t+1} - \mathcal{X}^{t+1}||^2} \quad (25)$$

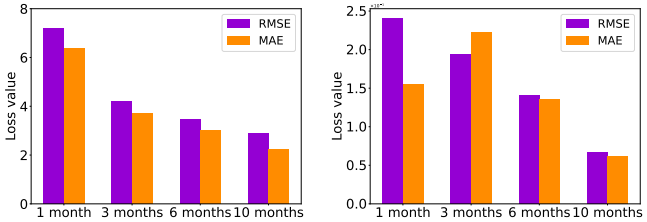
where $\hat{\mathcal{X}}^t$ is the prediction and $\mathcal{X}^{t+1}$ is the ground truth.

5.1.5 Convergence of the algorithm

Fig. 7 shows the training loss curves of the algorithm on the two datasets. One can see that MBA-STNet converges after about 180 epochs on both datasets, which shows it converges quickly. The loss curves drops smoothly, and there are still a few fluctuations on the loss during training. This is mainly because the data used for training is normalized. In addition, the zero-mean distribution we used as prior brings about the benefits of regularization [40], [41]. In the following experiment, we train MBA-STNet on both datasets 200 epochs.

5.2 Comparison with Baselines

Table 2 shows the comprehensive performance comparison among different methods over the two datasets. The best



(a) NYCBike crowd flow (b) NYCBike flow OD

Fig. 9. RMSE and MAE comparison with different size of training data.

results are highlighted with bold font, and the best results achieved by baselines are underlined.

It shows that the newly proposed MBA-STNet achieves the best results in most cases. It shows that traditional statistics based method ARIMA achieves the worse performance among all the methods in both cases. It is not surprising because ARIMA only uses the time series data of each region, but ignores the spatial dependency. On NYCBike dataset, compared with the best results achieved by baselines, MBA-STNet reduces RMSE of crowd flow prediction and flow OD prediction from 2.995 (MT-ASTN) to 2.905 which surpasses the MT-ASTN by nearly 3%, and from 0.074 (MT-ASTN) to 0.067, respectively. On NYCTAxi dataset, MBA-STNet improves the RMSE of the two tasks from 12.299(MT-ASTN) to 7.157, and from 0.087 (MT-ASTN) to 0.082 (5.7% improved), respectively. Both are significant improvements. On crowd flow prediction, the drop of RMSE on NYCTaxi is also remarkable from 12.299 to 7.157.

Table 2 illustrates that RMSE and MAE on NYCBike are almost all much smaller than NYCTaxi. It is reasonable because the bike trips can only be 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 MBA-STNet is more effective than existing state-of-the-art single- and multi-task learning approaches on the two prediction tasks.

5.3 Comparison with Variation Models

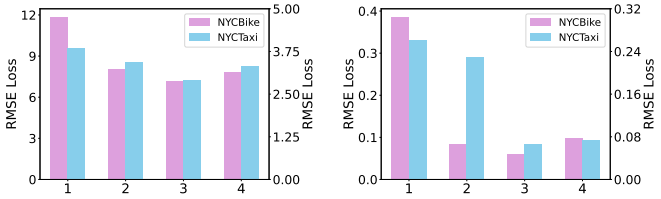
To examine whether the components in MBA-STNet are all helpful to the prediction task, we compare MBA-STNet with its variants MBA-STNet (L_{bayes}), MBA-STNet (L_{adv}), MBA-STNet (Gra), MBA-STNet (Que). The results are shown in Fig. 8. One can see that the Bayesian training, the adversarial loss, the 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 which follows the MT-ASTN. On NYCTaxi dataset, the bayesian training is more important in both crowd flow prediction and flow OD prediction tasks. 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 MBA-STNet with the single-task version of MBA-STNet named SBA-STNet. The single-task model SBA-STNet has the similar network structure as MBA-STNet shown in Fig. 3. The difference is that SBA-STNet performs crowd flow prediction and flow OD prediction separately without learning the shared features. Thus SBA-STNet 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 MBA-STNet are lower than that of the SBA-STNet, 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 performance improved 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 0.117 to 0.061 and NYCBike from 0.074 to 0.067.

TABLE 3
Single- and multi-task learning comparison

Dataset	Methods	Crowd Flow		Flow OD	
		RMSE	MAE	RMSE	MAE
NYCBike	SBA-STNet	3.312	2.786	0.074	0.073
	MBA-STNet	2.905	2.227	0.067	0.061
NYCTaxi	SBA-STNet	8.500	7.408	0.117	0.094
	MBA-STNet	7.157	5.976	0.061	0.059



(a) Crowd flow (b) Flow OD

Fig. 10. RMSE comparison with various HSTAN layer numbers

5.5 Ablation Study

To evaluate the effects of training data amount, length of time slots, parameter sensitivity analysis, sizes of cell regions and length of temporal queue, we perform the ablation studies as follows.

5.5.1 *Effects of training data amount*

To study the effect of training data amount on the model performance, we conduct experiments by sampling different amount of data in NYCBike datasets. To test how the amount of training data influences the prediction performance on the crowd flow and flow OD prediction tasks, we conduct the experiment with 1 month, 3 months, 6 months and 10 months data for training. The results are shown in Fig. 9. One can see the RMSE decreases from around 7.176 to 2.905 over NYCBike with the increase of the training data amount for flow prediction. It can also be seen that the RMSE decreases from around 0.241 to 0.067 over NYCBike for OD prediction as the training data increase. It shows that

TABLE 4
Performance comparison with different time slots over NYCTaxi

Time Slots	Input		Prediction	
	RMSE	MAE	RMSE	MAE
1 step	33.327	26.347	7.157	5.976
3 steps	16.352	12.942	13.620	10.770
6 steps	7.157	5.976	17.459	13.832

more training data can lead to better prediction performance as more useful knowledge can be learned.

5.5.2 Effects of length of time slots

We then evaluate the performance of the methods on different historical input time slots, as well as the prediction time slots (i.e., multi-step prediction). To test the effect of the time slot number to the model performance, we set the historical step size as 1, 3, 6 to predict the crowd flow and OD flow in the future 1 hour over NYCTaxi dataset. The results are shown in Table 4. One can see the RMSE decreases from 33.327 to 7.157 for flow prediction, which shows that the model performance presents an increase trend with the increase of the input time slot number. In addition, we also test the performance of the methods on multi-step prediction. We set the historical step size as 6 to predict the crowd flow and OD flow in the future 1, 3, 6 steps over NYCTaxi dataset, respectively. From Table 4, one can see with the increase of future prediction steps, MAE increases from 5.976 to 13.832, which is still comparable with baselines. The results show the effectiveness of temporal queue for long-term prediction.

5.5.3 Parameter sensitivity analysis

We next study how sensitive the model is on the deep neural structure (e.g., HSTAN Layers). We only give the RMSE comparison on both crowd flow and flow OD prediction tasks over NYCTaxi and NYCBike datasets. Fig. 10 shows the performance bars under different number of HSTAN layers. One can see that the performance on crowd flow and flow OD prediction first drops significantly and then slightly rises up with the increase of HSTAN layers. It shows that 3 layers of HSTAN is a reasonable setting in this experiment. More layers will degrade the model performance as it will lead to a more complex model with more parameters which thus may result in overfitting.

5.5.4 Sizes of cell regions

It is possible that the size of cell regions can affect the model performance. We conduct experiments under different settings of cell region partition (i.e., $4 \times 4, 8 \times 8, 12 \times 12, 16 \times 16, 32 \times 32$). Fig. 11 shows the RMSE comparison on crowd flow prediction over NYCTaxi and NYCBike datasets. One can see RMSE decreases from 4.327 to 2.905, and then increases to 3.053 over NYCBike when cell regions become more fine-grained. It shows that appropriate cell region division will lead to better model performance as it brings more spatial knowledge. But when the partition becomes more fine-grained, the model performance will degrade due to data sparsity.

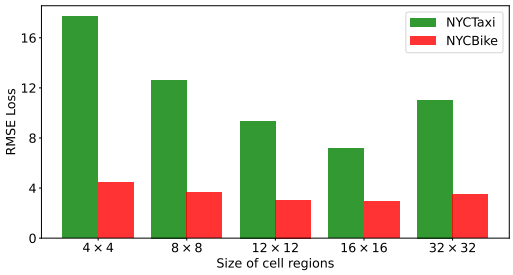


Fig. 11. RMSE comparison with different size of cell regions.

TABLE 5
Performance comparison with different length of Temporal Queue

Dataset	Length	Crowd Flow		Flow OD	
		RMSE	MAE	RMSE	MAE
NYCBike	1 day	3.456	2.988	0.231	0.151
	7 days	3.190	2.673	0.133	0.122
	15 days	2.905	2.227	0.067	0.061
NYCTaxi	1 day	12.596	11.197	0.288	0.206
	7 days	8.690	7.558	0.144	0.105
	15 days	7.157	5.976	0.061	0.059

5.5.5 Length of temporal queue

From MT-ASTN [15] and Fig. 8, it can be easily found that temporal queue which is used to extract long-term features is very effective for both crowd flow and flow OD prediction tasks. We finally study the influence of the length of temporal queue on model performance. We conduct experiments under different settings of temporal queue length (e.g., 1 day, 7 days, 15 days) which can be shown in table 5. Due to the restriction of the device performance (i.e., GPU memory), we set the maximum length of the temporal queue to 15 days. From Table 5, it shows the model performance improves with the increase of temporal queue. One can see that RMSE decreases from 3.456 (1 day) to 2.905 (15 days) on crowd flow prediction, and from 0.231 (1 day) to 0.067 (15 days) on flow OD prediction over NYCBike. The results show the temporal queue is effective for both tasks as it can encode long-term temporal dependencies well.

5.6 Case Study

5.6.1 Visualization on prediction 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 Fig. 12. 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

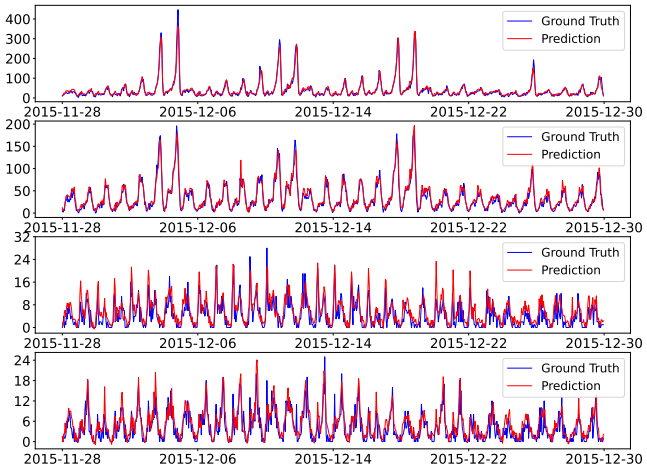


Fig. 12. Prediction vs ground truth on region $r_{8,8}$ (top to down: taxi inflow, taxi outflow, bike inflow and bike outflow).

queue. The result also shows that the designed attentive temporal queue component is effective to complex temporal trends from the long-range temporal data.

6 Conclusion

In this paper, we proposed a novel Multi-task Bayes-enhanced Adversarial Spatio-Temporal Network model to jointly predict 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. The orthogonality constraint between shared and private features were also incorporated to further decompose shared and private features. Considering the complex spatial spatial-temporal correlations and data uncertainty, the proposed spatial-temporal network utilized a designed BHSTAN 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. In the future, It would be also interesting to further design a new strategy to solve the problem of memory explosion when cell regions become more fine-grained.

Acknowledgement

Authors would like to express their appreciation to reviewers and editors for their valuable comments and effort. This work was supported by TRS RGC Theme-based Research Scheme 2020/21 (Tenth Round) (T41-603/20-R), CRF RGC Collaborative Research Fund 2018/19 (RGC No.: C5026-18G) and PolyU Internal Start-up Fund (P0035274). It was also partially supported by the National Natural Science Foundation of China (No.: 62172443) and the Fundamental Research Funds for the Central Universities (No.: NZ2020014).

References

[1] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 5668–5675.

[2] S. Wang, J. Cao, and P. Yu, "Deep learning for spatio-temporal data mining: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[3] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," *Proceedings of AAAI*, 2017.

[4] B. Du, H. Peng, S. Wang, M. Z. A. Bhuiyan, L. Wang, Q. Gong, L. Liu, and J. Li, "Deep irregular convolutional residual lstm for urban traffic passenger flows prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 972–985, 2019.

[5] Y. Zhang, S. Wang, B. Chen, J. Cao, and Z. Huang, "Trafficgan: Network-scale deep traffic prediction with generative adversarial nets," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[6] Z. Lin, J. Feng, Z. Lu, Y. Li, and D. Jin, "Deepstn+: Context-aware spatial-temporal neural network for crowd flow prediction in metropolis," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1020–1027.

[7] X. Shi, Z. Chen, H. Wang, D. Y. Yeung, W. K. Wong, and W. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," *Advances in Neural Information Processing Systems*, 2015.

[8] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations*, 2018.

[9] L. Liu, Z. Qiu, G. Li, Q. Wang, W. Ouyang, and L. Lin, "Contextualized spatial–temporal network for taxi origin-destination demand prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3875–3887, 2019.

[10] Y. Wang, H. Yin, H. Chen, T. Wo, J. Xu, and K. Zheng, "Origin-destination matrix prediction via graph convolution: A new perspective of passenger demand modeling," in *Proceedings of KDD*, 2019, pp. 1227–1235.

[11] G. R. Terrell and D. W. Scott, "Variable kernel density estimation," *The Annals of Statistics*, pp. 1236–1265, 1992.

[12] J. Zhang, Y. Zheng, J. Sun, and D. Qi, "Flow prediction in spatio-temporal networks based on multitask deep learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 3, pp. 468–478, 2019.

[13] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *International Conference on Machine Learning*. PMLR, 2015, pp. 1613–1622.

[14] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059.

[15] S. Wang, H. Miao, H. Chen, and Z. Huang, "Multi-task adversarial spatial-temporal networks for crowd flow prediction," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 1555–1564.

[16] L. BAI, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[17] D. Q. Nguyen, C. X. Chu, S. Thater, M. Pinkal *et al.*, "Sequence to sequence learning for event prediction," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 2017, pp. 37–42.

[18] X. Cheng, R. Zhang, J. Zhou, and W. Xu, "Deeptransport: Learning spatial-temporal dependency for traffic condition forecasting," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[19] S. Shekhar and B. M. Williams, "Adaptive seasonal time series models for forecasting short-term traffic flow," *Transportation Research Record*, vol. 2024, no. 1, pp. 116–125, 2007.

[20] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 871–882, 2013.

[21] J. Ahn, E. Ko, and E. Y. Kim, "Highway traffic flow prediction using support vector regression and bayesian classifier," in *2016 International conference on big data and smart computing (BigComp)*. IEEE, 2016, pp. 239–244.

[22] X. Zhou, Y. Shen, Y. Zhu, and L. Huang, "Predicting multi-step citywide passenger demands using attention-based neural networks," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 736–744.

[23] X. Cheng, R. Zhang, J. Zhou, and W. Xu, "Deeptransport: Learning spatial-temporal dependency for traffic condition forecasting,"

in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.
[24] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.
[25] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 914–921.
[26] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 1234–1241.
[27] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 3634–3640.
[28] J. Ren and Q. Xie, "Efficient od trip matrix prediction based on tensor decomposition," in *Proceedings of MDM*, 2017, pp. 180–185.
[29] K. Chu, A. Y. S. Lam, and V. O. K. Li, "Deep multi-scale convolutional lstm network for travel demand and origin-destination predictions," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2019.
[30] D. Zhang, F. Xiao, M. Shen, and S. Zhong, "Dneat: A novel dynamic node-edge attention network for origin-destination demand prediction," *Transportation Research Part C: Emerging Technologies*, vol. 122, p. 102851, 2021.
[31] M. Bierlaire and F. Crittin, "An efficient algorithm for real-time estimation and prediction of dynamic od tables," *Operations Research*, vol. 52, no. 1, pp. 116–127, 2004.
[32] J. Zhang, H. Che, F. Chen, W. Ma, and Z. He, "Short-term origin-destination demand prediction in urban rail transit systems: A channel-wise attentive split-convolutional neural network method," *Transportation Research Part C: Emerging Technologies*, vol. 124, p. 102928, 2021.
[33] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 1907–1913.
[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
[35] Y. Yang, J. Cao, M. Stojmenovic, S. Wang, Y. Cheng, C. Lum, and Z. Li, "Time-capturing dynamic graph embedding for temporal linkage evolution," *IEEE Transactions on Knowledge and Data Engineering*, 2021.
[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
[37] R. Bassett and J. Deride, "Maximum a posteriori estimators as a limit of bayes estimators," *Mathematical Programming*, vol. 174, no. 1, pp. 129–144, 2019.
[38] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference." *Journal of Machine Learning Research*, vol. 14, no. 5, 2013.
[39] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," *Advances in neural information processing systems*, vol. 28, pp. 2575–2583, 2015.
[40] M. Sun, T. Zhang, Y. Wang, G. Strbac, and C. Kang, "Using bayesian deep learning to capture uncertainty for residential net load forecasting," *IEEE Transactions on Power Systems*, vol. 35, no. 1, pp. 188–201, 2019.
[41] Y. Gal, "Uncertainty in deep learning," *Ph.D. dissertation*, 2016.
[42] P. H. Garthwaite, J. B. Kadane, and A. O'Hagan, "Statistical methods for eliciting probability distributions," *Journal of the American Statistical Association*, vol. 100, no. 470, pp. 680–701, 2005.
[43] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.
[44] P. Liu, X. Qiu, and X. Huang, "Adversarial multi-task learning for text classification," *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 1–10, 2017.
[45] I. J. Goodfellow, J. Pouget-Abadiey, M. Mirza, B. Xu, D. Warde-Farley, S. Ozairz, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proceedings of NeurIPS*, 2014.

Hao Miao received the M.E. in Computer science and technology from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2021. He is currently a PhD Fellow in Computer Science at Aalborg University and a Research Assistant at The Hong Kong Polytechnic University (PolyU). From September 2019 to November 2019, he was a Visiting Student in PolyU, Hong Kong, China. His research interests include Spatio-temporal data analytic, incremental learning and federated learning.

Jiaxing Shen received B.E. in Software Engineering from Jilin University in 2014 and Ph.D. in Computer Science from PolyU in 2019. He is currently a Research Assistant Professor in the Department of Computing at The Hong Kong Polytechnic University. His research interests include Mobile Computing, Data Mining, Social Computing, Affective Computing, and Internet of Things. He has published over 20 high-quality papers in top conferences and journals including INFOCOM, WWW, TMC, and TOIS. He also won the best paper award of INFOCOM 2020. He served as a reviewer for many top international conferences and journals.

jiannong Cao received the M.Sc. and Ph.D. degrees in computer science from Washington State University, Pullman, WA, USA, in 1986 and 1990, respectively. He is currently the Chair Professor with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. His current research interests include parallel and distributed computing, mobile computing, and big data analytics. Dr. Cao has served as a member of the Editorial Boards of several international journals, a Reviewer for international journals/conference proceedings, and also as an Organizing/ Program Committee member for many international conferences.

Jiangnan Xia received the B.E. degree in Software Engineering from NanChang University, Nanchang, China, in 2021. She is currently a Master student of Central South University in the department of Software Engineering. Her research interest includes Spatio-temporal data mining and deep learning.

Senzhang Wang received the B.Sc. degree from Southeast University, Nanjing, China, in 2009, and the Ph.D. degree in computer science from Beihang University, Beijing, China, in 2015. He is currently a Professor with School of Computer Science and Engineering, Central South University, Changsha. He has published over 100 papers on the top international journals and conferences such as Knowledge and Information Systems, ACM SIGKDD Conference on Knowledge Discovery and Data Mining, AAAI Conference on Artificial Intelligence. His current research interests include data mining and social network analysis.