IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received August 21, 2019, accepted September 21, 2019, date of publication October 2, 2019, date of current version October 16, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2944990*

# Pre-Trained Bidirectional Temporal Representation for Crowd Flows Prediction in Regular Region

**WENYING DUAN[1], LIU JIANG[2], NING WANG[1], AND HONG RAO[1]**
[1]Information Engineering School, Nanchang University, Nanchang 330031, China
[2]Civil and Environmental Engineering, Stanford University, Stanford, CA 94305, USA

Corresponding author: Hong Rao (raohong@ncu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61262047 and Grant 81960325, in part by the Science and Technology Foundation of the Jiangxi Education Department under Grant GJJ14141, and in part by the Key Research and Development Program of Jiangxi Province under Grant 20171BBE50063.

**ABSTRACT** Deep learning-based methods for predicting spatial-temporal data such as crowd flows need to consider both spatial dependency and temporal dependency. Previous research mainly focused on modeling spatial dependency, whereas studies on temporal dependency are few. Existing finite deep learning-based methods for temporal dependency modeling can be divided into RNN-based methods and domain knowledge-based methods. However, RNN-based methods are hard to learn very long-term temporal dependency, and domain knowledge-based methods cannot model temporal dependency automatically, depending on data pre-processing based on prior knowledge. In view of the problem, crowd flows prediction in regular gridded regions are studied and a model called Pre-trained Bidirectional Temporal Representation (PBTR) based on Transformer encoder is proposed capable of modeling very long-term temporal dependency automatically. PBTR is simple, scalable, and can be combined with any other spatial component. Furthermore, we introduce Crowd Flows Prediction based on PBTR (CPPBTR) to form a Transformer based encoder-decoder framework. There are two decode stages in the proposed model. At decoder-stage 1, 'draft' sequence is generated. At decoder-stage 2, each timestep of the 'draft' sequence is masked and fed into PBTR to predict the refined flow for each masked position. Experiment results demonstrate that our method outperforms RNN-based methods and domain knowledge-based methods.

**INDEX TERMS** Traffic prediction, neural networks, transformer, bidirectional temporal representation, very long-term temporal dependency.

## I. INTRODUCTION

In this paper, crowd flows prediction in regular gridded regions is studied [1], [2], [7]. Traffic flow predicting is a very important component of urban computing, however, it is a very challenging task, affected by a complex of factors: spatial dependencies (or spatial representation), temporal dependencies (or temporal representation) and external influences (weather conditions, events and so on). Deep learning has been successful in many fields [19]. Recently, many researchers have tried to use neural networks for many types of traffic flow prediction including taxi demand prediction [6], crowd flows prediction [1], [2], [7], speed prediction [8] and so on [20]. Crowd flow is a typical spatial-temporal data. Researchers have considered temporal and spatial properties of crowd flows when applying deep neural networks (DNNs) [8]. In order to model the spatial dependency of crowd flows, a popular way is to treat the flow data at each timestep as an image and, utilize convolutional neural network (CNNs) to model spatial correlation [1], [2], [21]. However, not all traffic data can be represented by images with time series. With the rise of graph neural networks, an increasing number of researchers uses graphs to describe traffic data and applies graph neural networks (GNNs) to learn spatial representation. In terms of modeling temporal dependency, it remains very challenging because there are multiple types of temporal dependencies:

- **Closeness:** the flows are affected by recent timesteps. Taking a example, a congestion occurring at 7pm will affect traffic flow at 7pm.

The associate editor coordinating the review of this manuscript and approving it for publication was Hongxiang Li.

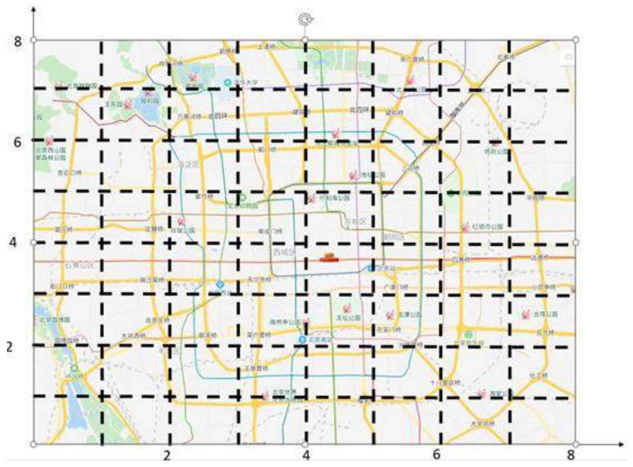


**FIGURE 1.** Regular gridded map(8 × 8) of Beijing. In this illustration, Beijing is divided into 64 regular grids.

- **Periods:** the crowd flows have many types of periods: daily, weekly, *etc*. For example, flows at 6pm on every working day demonstrate a relatively high consistency, but a big difference expected between flows at 4pm and flows at 6 pm even on a same day (daily period) and the conditions on this Monday is similar to last Monday (weekly period).

To tackle this challenge, scholars usually utilize recurrent neural networks (RNNs) and its variants such as LSTM, GRU, *etc*. [6], [8] to model temporal dependencies. Although RNNs are capable to model time series such as crowd flows data, it is difficult to learn the multiple types of temporal dependencies of crowd flows because the main deficit of RNNs is that RNNs cannot model very long-term temporal dependency. For example, assuming a time interval to be 1 hour in this study, there are 24 intervals per day. The crowd flows are predicted every half an hour/one hour in advance. To guarantee the data amount for training, it is necessary to use at least data of one month so that the neural networks could learn the periodicity and trend of crowd flows data. In this case, the length of the input sequence is at least 672. Such long-range sequence induces optimization difficulty for RNNs, making them infeasible in applications. Therefore, some scholars proposed domain knowledge-based method by manually selecting the data of key moments as input according to the periodicity, trend of traffic. [1], [2], [6], [7]. However, these approaches rely heavily on prior knowledge with main focus on modeling spatial dependency, and research effort into automatic modeling of very long-temporal dependency is yet to be thoroughly investigated. Recently, transformer is widely used in sequence modeling, and both BERT [12] and GPT-2 [13] have demonstrated that transformer-based architecture can model very long sequences (512 in BERT and 1024 in GPT-2). Therefore, we adopt Transformer [3] as the main structure of the model to be established in this study.

Further, pretraining has achieved great success in computer vision, natural language processing, video understanding and so on. It is also introduced to traffic prediction. Different unsupervised pretraining methods of learning traffic representation has been explored. The most two successful pretraining methods are autoregressive (AR) modeling and autoencoding. Lv et al. used sparse stacked autoencoder (SAE) to learn traffic representation [8]. On the top of the pre-trained SAE, a logistic layer is applied to predict traffic. The experimental results demonstrate the effectiveness of the pre-training model. Furthermore, the pre-training approach of learning video representation also provide ideas for traffic prediction since that video understanding is a similar task to the prediction of crowd flows (crowd flows are treated as a sequence of images, just like video). Srivastava *et al.* introduced an unsupervised method consisting of a LSTM autoencoder model and a LSTM future predictor model (AR modeling) to learn both forward and backward representations of video. This method achieved an improvement on supervised task. Sun *et al.* used Srivastava's approach to predict traffic and achieved remarkable results [14].

BERT is a notable example of denoising autoencoding (DAE), it is allowed to utilize bidirectional information for reconstruction. It achieved remarkable performance in language understanding task. Inspired by BERT, many researchers introduced DAE into sequence task. For instance, Sun et.al. Proposed BERT4Rec [22] for sequential recommendation. BERT4Rec applied DAE for modeling behavior sequence. In general, bidirectional representation learning has been successful in multiple types of sequence. However, the bidirectional representation learning has not been applied in traffic prediction. Previous works were AR modeling and performed training and inference in a left-to-right (L2R) manner. They predicted the current flow conditioned on previous flow, whether it is domain knowledge-based or RNN-based. We believe that bidirectional representation learning is helpful in learning a good temporal representation. Motivated by this, a bidirectional temporal representation model known as Pre-trained Bidirectional Temporal Representation model (PBTR) based on Transformer is designed. It is a kind of denoise autoencoder modeling, which is different from autoregressive modeling based on Bi-RNNs. It is easy to use, capable and can be treated as a component to combine with any model utilized to learn spatial representation (or capture spatial structure), learning spatial-temporal representation simultaneously. Then, we introduce CPPBTR, which stands for Crowds flow Prediction based on Bidirectional Temporal Representation. There are two decode stages in our model. The decoder-stage 1 generates 'draft' sequence. At decoder-stage 2, we mask flow of each step of the 'draft' sequence and feed it into PBTR to predict the refined flow for each masked position. The contributions of our work are three-fold:

- We propose a pre-train deep bidirectional temporal representation model called PBTR. To the best of our knowledge, this is the first bidirectional spatial-temporal representation model.

- We design a novel crowd flows prediction algorithm (CPPBTR) based on PBTR.
- We conduct experiment on three datasets (Beijing taxicabs' trajectory, NYC bike trajectory and NYC taxi trajectory). The result of experiment demonstrates the performance of our model is superior to state-of-the-art methods.

## II. PRELIMINARY

In this section, the crowd flows prediction problem is briefly introduced. Then, we revisit the transformer.

### A. CROWD FLOWS PREDICTION

*Definition 1 (Region) [1]:* In this study, a city is partitioned into an grid map M $\in \mathbb{R}^{I\times J}$ based on the longitude and latitude where a grid is denoted as a region. Please see [1] for more details.

$$M = \begin{bmatrix} m_{11} & \cdots & m_{1J} \\ \vdots & \ddots & \vdots \\ m_{I1} & \cdots & m_{IJ} \end{bmatrix} \tag{1}$$

*Definition 2 (Inflow/Outflow) [1]:* Let $\mathbb{P}$ be a collection of trajectories at the $t^{th}$ timestep. For a grid $(i, j)$ that lies at the $i^{th}$ row and the $j^{th}$ cloumn, the inflow and outflow of the rows at the timestep $t$ are defined respectively as:

$$\mathrm{x}_t^{\mathrm{in,i,j}} = \sum_{T_r\in\mathbb{P}} |\{t > 1 | g_{t-1} \notin (i,j) \wedge g_t \in (i,j)\}| \tag{2}$$

$$\mathrm{x}_t^{\mathrm{out,i,j}} = \sum_{T_r\in\mathbb{P}} |\{t > 1 | g_t \in (i,j) \wedge g_{t+1} \notin (i,j)\}| \tag{3}$$

*Problem (Region):* The goal is to predict the future crowd flows given historical observations of flow. Let $x_t \in \mathbb{R}^{2\times m\times n}$ represent the flows observed at timestep $t$. The crowd flows prediction problem aims to learn a function that maps historical observations of flows $X = \{x_t | t = 1, 2, \ldots, n-1\}$ to $x_n$ at next timestep.

### B. TRANSFORMER

Transformer [3] is a powerful sequence modeling tool, it is a stack of several identical layers. The identical layer is called Transformer Block. A Transformer Block consists of a multi-head attention sub-layer and a position-wise fully connected feed-forward network sub-layer, a residual and layernorm connection is added between all sub-layers as shown as Fig. 2.

#### 1) MULTI-HEAD ATTENTION

Mathematically, given query matrix $Q$, key matrix $K$, and value matrix $V$, the matrix of outputs is calculated as follow:

$$\begin{aligned} \text{Attention}(Q, K, V) &= \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \\ \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \\ \text{where head}_i &= \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \end{aligned} \tag{4}$$

#### 2) POSITION-WISE FULLY CONNECTED FEED-FORWARD NETWORK

After obtaining outputs by the multi-head attention mechanism, a position-wise fully connected feed-forward network sub-layer is added. This forward network is defined as follows:

$$\text{FeedForward}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{5}$$

where $W_1$, $W_2$, $b_1$ and $b_2$ are learnable parameters.

## III. RELATED WORK

In this section, we introduce some deep learning methods for traffic prediction, as well as attention mechanisms that are widely used in various deep learning frameworks and tasks, including traffic prediction.

### A. TEMPORAL DEPENDENCY MODELING

The current neural networks models for modeling the temporal dependency of spatial-temporal data such as crowd flows can be classified into two categories: RNN-based methods and Domain Knowledge-based methods.

RNN-based methods usually apply long-short-term memory (LSTM) network to model temporal dependency. Yu et al. [20] applied LSTM to capture the temporal dependency for forecasting the traffic under extreme conditions. However, they did not consider the spatial dependency. DMVST-Net (Multi-View Spatial-Temporal Network) [6] is a LSTM-based model for taxi demand prediction, considering both spatial dependency and temporal dependency in a joint deep neural network model. It proposed Local-CNN model that captures local characteristics of regions in relation to their neighbors, then applied LSTM to model temporal dependency. Li et al. [10] introduced Diffusion Convolutional Recurrent Neural Network (DCRNN), using DCGRU to implement an encoder-decoder architecture for forecasting traffic, which can model both temporal dependency and spatial dependency. It proposed Diffusion Convolutional Gated Recurrent Unit (DCGRU) to replace the matrix multiplications in GRU with diffusion convolution. However, the forementioned methods cannot model very long-term temporal dependency.

Domain Knowledge-based methods believe that the temporal dependency mainly contains multiple views according to the temporal closeness (recent segment) [1], [2], [7], [15], period [1], [2], [7], [14], [15] and trend [1], [2], [7], [14]. Guo et al. [15] considered the recent segment and two types of period (daily and weekly). Zhang et al. [14] considered not only the period, but also considered the trend (monthly and quarterly). The corresponding recent, daily, weekly, monthly, and quarterly timesteps are selected as the key timesteps, to construct five views. [14] achieved state-of-the art performance on predicting citywide crowd flows in irregular regions. Domain knowledge-based methods depend on the prior knowledge. We believe that the core of domain knowledge-based method is feature selection. It is based on prior knowledge to select corresponding timesteps, that is, manually selecting the temporal feature from spatial-temporal data such as traffic. As mentioned in section 1, we believe that traffic data delivers essential information. Therefore, the prior knowledge of traffic is easy to obtain and effective for traffic prediction of human behavior since the regularity of human behavior. However, not all prior knowledge of spatial-temporal data has such advantages.

In summary, although RNN-based method can automatically learn the temporal representation, it cannot learn the very long-term temporal dependency. Domain knowledge-based methods are unable to learn temporal dependency automatically. Our approach is capable of learning very long-term dependency automatically and not rely on domain knowledge.

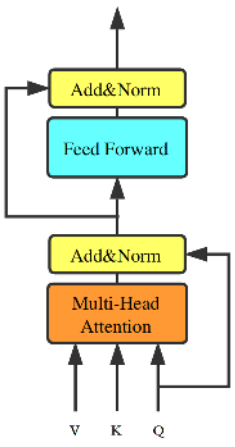


**FIGURE 2.** **The structure of a Transformer Block.**

## B. ATTENTION MECHANISM

Recently, attention mechanisms have been widely used in various tasks such as natural language generation, speech recognition, image caption and so on. Bahdanau et al. [16] proposed soft attention and applied it to machine translation, greatly improve the performance of neural machine translation. V. Ashish et al. [3] proposed multi-head attention and self-attention to model sequence, achieving significant success in machine translation. Compared with RNNs, self-attention mechanism is easier to capture very long-term co-dependency and increase the parallelism of computing. Furthermore, attention mechanisms are widely applied to various fields, including traffic prediction. DeepTransport [9] uses attention mechanism to quantify the spatial-temporal relations. It focused on graph with time series, using LSTM to capture temporal dependency. ATSGCN (Attention based Spatial-Temporal Graph Convolution Network) [15] is another attention based model which applied graph convolutional network (GCN) to learn the spatial dependency of graph-based traffic data. It utilized a novel spatial-temporal attention mechanism to capture the dynamic spatial and temporal dependency on the graph-based traffic network.

## IV. OUR APPROACH

In this section, we introduce our proposed approach. In general, it contains two components: Pre-train Bidirectional Temporal Representation model (PBTR) as well as Crowds flow Predicton based on PBTR (CPPBTR).

### A. PRE-TRAIN BIDIRECTIONAL TEMPORAL REPRESENTATION MODEL

#### 1) LOSS FUNCTION

Let $x$ and $\hat{x}$ be the ground-truth flow matrix and predicted flow matrix respectively. The objective function we employ here is mean squared error:

$$\mathcal{L}(\mathbf{x}_t, \widehat{\mathbf{x}}_t) = \|\mathbf{x}_t - \widehat{\mathbf{x}}_t\|_2^2 \tag{6}$$

#### 2) ARCHITECTURE OF PBTR

PBTR consists of multi-layer Transformer encoder. Please see [16] for more details. PBTR and Spatial Components such as residual block [1], [2], Local-CNN [6] or any other neural networks can be combined to create a composited model which is able to learn temporal representation and spatial representation as shown in Fig. 2.

FNN (Fully Connected Network) is the external component for external representation learning, a two-layer neural network consisting of an embedding layer and a fully connected layer. The external information we considered is weather, holiday and metadata (i.e. DayOfWeek, WeekDay/Weekend).

#### 3) BIDIRECTIONAL TEMPORAL REPRESENTATION LEARNING TASK

Inspired by BERT, we design a task like Masked Language Model (LM) to learn bidirectional temporal representation. The training data generator chooses 7% of frames (a frame is the crowd flow at one timestep) at random, e.g. in Fig. 2, $x_t$ and $x_{t+4}$ are chosen. The data generator performs the follow procedure:

- 90% of the time, mask the chosen frame with zeros.
- 10% of the time, keep the chosen frame.

On the top of PBTR is a linear layer (the prediction layer in Fig. 2) used to predict crowd flows. Let $h_t$ be the representation produced by PBTR at timestep $t$. The prediction value is denoted as follow:

$$\hat{x}_t = \tanh\left(w_1 h_t + b_1\right) \tag{7}$$

where $w_1$ and $b_1$ are learnable parameters of the prediction layer. Tanh is the activation function, defined as the following equation.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{8}$$

We only predict the masked frames, and the learning objective of PBTR is to minimize mean squared error between the predicted flow matrix and the true flow matrix:

$$loss = \sum_{t \in \mathcal{J}} \mathcal{L}\left(\hat{x}_t, x_t\right) \tag{9}$$

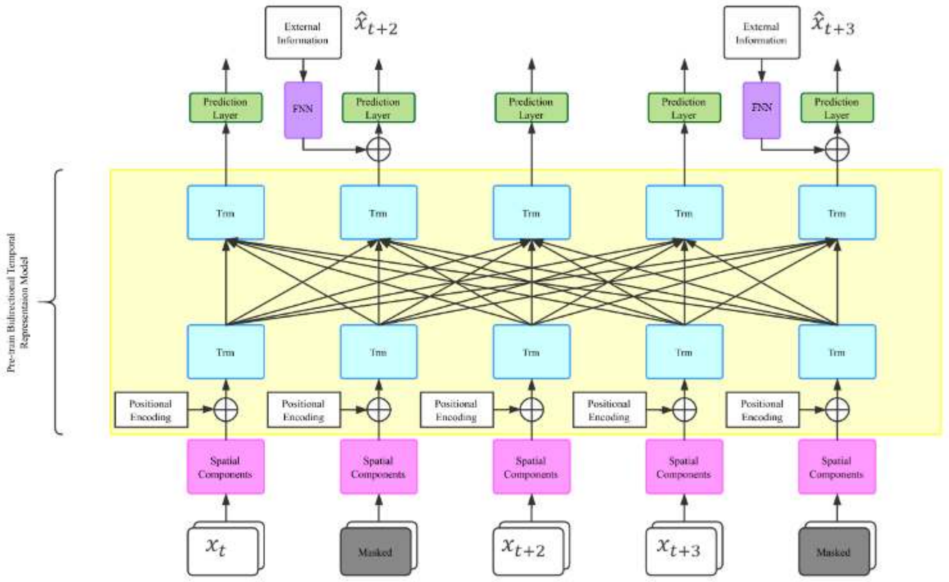


**FIGURE 3.** The structure of PBTR. Trm stands for Transformer block. The temporal representations are jointly conditioned on both left and right flow in all layers.

**Algorithm 1** Pre-Training Processing of PBTR

```
Input: $\mathcal{D}$: training dataset
Input: $\Pi_\theta$: PBTR model to be trained with parameters $\theta$
Input: $\theta_0$: Initial parameters
    $t \leftarrow 0$ (Initialize timestep)
1: repeat
2:     $t \leftarrow t + 1$
3:     Sample a crow flow sequence $X \sim \mathcal{D}$
4:     Randomly mask $x_t$ in $X$ according to the generator performing.
5:     Calculate the reconstruction $\widehat{X}$
6:     Calculate $loss(\widehat{X}, X)$ according to Eq. 6.
7:     $\theta_t \leftarrow \theta_{t-1} - \nabla_\theta loss(\widehat{X}, X)$
8: until converge
Output: $\Pi_\theta$ with trained parameters $\theta_t$
```

where $x_t$ is the ground-truth at timestep $t$, $\mathcal{J}$ is all available timestep.

### 4) WHY BIDIRECTIONAL REPRESENTATION LEARNING

We believe that such Left-to-right learning (auto-regressive modeling) restricts the power of transformer. In terms of L2R learning, the leftward flow need to be prevented to preserve the auto-regressive property. Usually, researchers implement this inside of scaled dot-product attention by masking out all values in the input softmax which correspond to illegal connections [3].

As claimed in introduction of this paper, we need length of 720 (one month) or more to learning the periodicity of temporal representation. However, to preserve the auto-regressive property, scaled dot-product attention cannot fully learn the periodicity of temporal representation. For auto-regressive modeling: $x_t = a_1x_1 + a_2x_2 + \cdots, +a_{t-1}x_{t-1} + \varepsilon$. In particular, $x_t = f\,(\text{MultiHead}\,(x_{1:t-1}, x_{1:t-1}, x_{t-1}))$, where $x_{1:t}$ is $\{x_1, x_2, \ldots, x_{t-1}\}$, $f(x)$ is a non-linear function. This means the information flow after timestep $t$ cannot attend to the information flow before timestep $t$ (See Fig. 4(a)). Thus, even if the sequence is long enough, the attention layer cannot fully learn the periodicity of temporal representation (Nearly half of the elements in the matrix are masked). And PBTR, each timestep can attend to all timesteps (See Fig.4(b)). So PBTR can fully learn the periodicity of temporal representation.

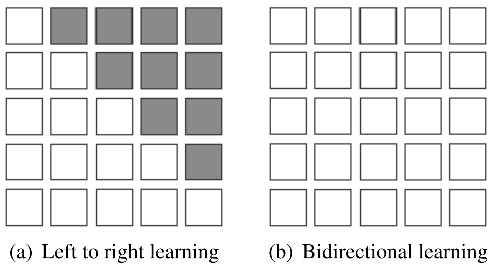


**FIGURE 4.** The attention matrix of left to right learning and bidirectional learning. Gray block means the value is masked.

## B. CROWDS FLOW PREDICTOR BASED ON PBTR

In this sub-section, we will demonstrate how to predict flows by utilizing PBTR. The predictor is based on Transformer, consisting of PBTR as the encoder and a decoder with two stages.

### 1) ENCODER

The encoder consists of a spatial module S and a PBTR. It computes the spatial-temporal representation $H$, denoted

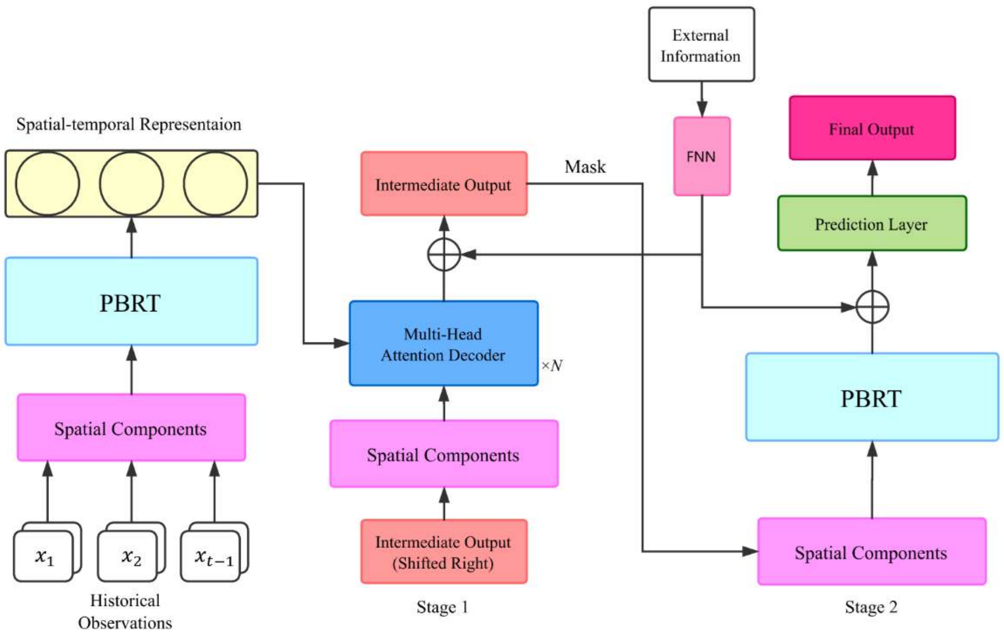


**FIGURE 5. The architecture of CPPBTR.The PBTR is utilized as the encoder and the second stage decoder. The first stage decoder is a multi-head attention decoder.**

by following equation.

$$H = PBRT\left(S\left(x_1, x_2, \ldots x_{t-1}\right)\right) \tag{10}$$

### 2) DECODER-STAGE 1

The attention mechanism makes the model focus on critical information, but it also limits the interaction of information. The multi-head attention mechanism can increase the space of the network and allow more information to be transmitted without increasing the time complextiy. So, in the decoder-stage 1, we introduce a $N$ multi-head attention layer Transformer decoder to predict crowds flow. At the $t_{th}$ timestep, the decoder-stage 1 predicts flow conditioned on previous flow, the spatial-temporal representation and external information $E_t$ at timestep $t$ as shown in Fig. 2.

$$\widehat{x_t} = Decoder_1\left(\hat{x}_{<t}, H, E_t\right) \tag{11}$$

The objective of decoder-stage 1 is to minimize the mean squared error, in which $X$ is the ground truth.

$$loss_1 = \mathcal{L}(\widehat{X}, X) \tag{12}$$

However, the experimental results demonstrated that if the encoder of transformer is initialized with the parameters of PBTR, and the direct application of a L2R decoder is not feasible enough. If PBTR is utilized in decoder, the decoder would be negatively impacted by the incomplete sequence since that PBTR is a kind of DAE modeling, but the transformer decoder is a kind of autoregressive modeling. To tackle this problem, we introduce another decoder called decoder-stage 2 to mitigate it in the predictor which will be detailed in the next sub-section.

### 3) DECODER-STAGE 2

The main reason to introduce the decoder-stage 2 is to enhance the decoder utilizing the spatial-temporal representation of PBTR, so we reuse the encoder and its parameters are fixed during training. The decoder-stage 2 receives a flow sequence generated by decoder-stage 1 $\widehat{X}$ as input and generated the final output $\bar{X}$. The process of decoder-stage 2 is as shown in algorithm 2.

The decoding of stage 2 process provides a complete input sequence such that the process of decoder stage 2 is similar to the bidirectional temporal representation learning task. Therefore, decoder-stage 2 can benefit from the representation of the PBTR.

The learning objective of this process is shown as follow:

$$loss_2 = \mathcal{L}(\bar{X}, X) \tag{13}$$

The objective of our model during training is the sum of $loss_1$ and $loss_2$:

$$loss = loss_1 + loss_2 \tag{14}$$

**Algorithm 2** Processing of Decoder-Stage 2

```
Input: intermediate output generated by decoder-stage $\widehat{X} = \{\hat{x}_t, \hat{x}_{t+1}, \ldots, \hat{x}_{t+T}\}$
Output: final output $\bar{X}$
1: for timestep i ($t \leq i \leq t+T$) do
2:     mask $\hat{x}_i \in \widehat{X}$
3:     feed into decoder-stage 2
4:     generate $\bar{x}_i$
5:     replace $\hat{x}_i$ in $\bar{X}$ with $\bar{x}_i$
6: end for
7: return final output $\bar{X}$
```

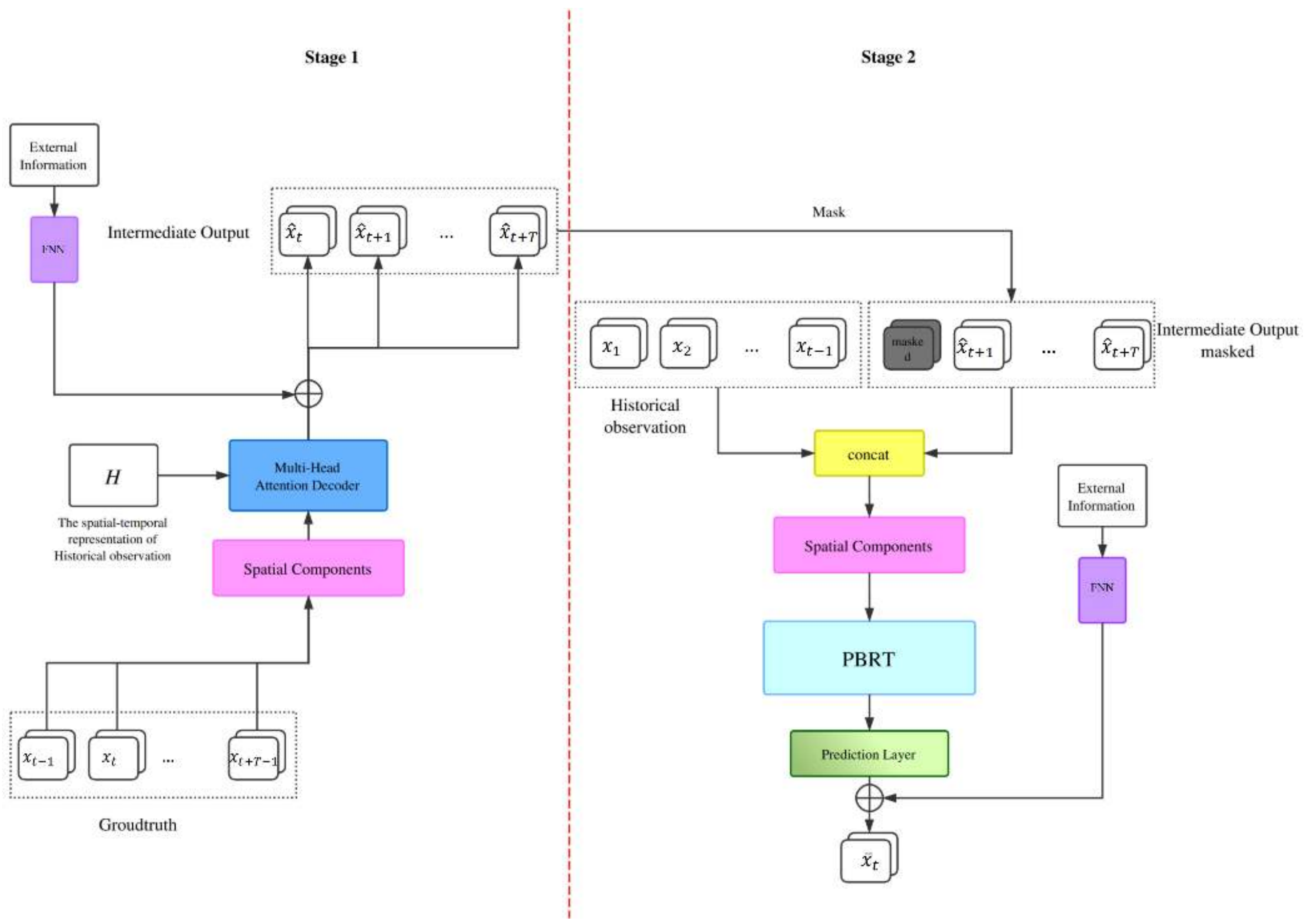
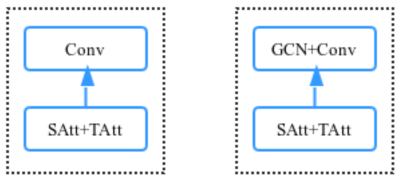


FIGURE 6. An example of decoding process. The process of predicting $x_t$ is illustrated. On the left side of the red dotted line is deocder-stage 1, on right side of the red dotted line is decoder-stage 2. In stage 1, model generated intermediate output. In stage 2, $x_t$ is masked. After that, the masked sequence and historical observation sequence are concatenated into a new sequence. Then the new sequence is fed into PBTR to generate the final output $\bar{x}_t$.

TABLE 1. Details of datasets.

| Dataset | TaxiBJ | BikeNYC | TaxiNYC |
|---|---|---|---|
| Datatype | TaxiGPS | Bike rent | TaxiGPS |
| Location | Beijing | New York | New York |
| Time Span | 7/1/2013-10/3/2013 3/1/2014-6/30/2014 3/1/2015-6/30/2015 11/1/2015-4/10/2016 | 4/1/2014-9/30/2014 | 1/1/2011-12/31/2016 |
| Time interval | 30 minutes | 1 hour | 1 hour |
| Gird map size | (32,32) | (16,8) | (10,20) |
| **External infirmations** | | | |
| Holidays | 41 | 20 | 20 |
| Weather | 16 types(e.g., Sunny, Rainy) | N/A | N/A |



(a) ST block in ASTCNN (b) ST block in ASTGCN

FIGURE 7. The only difference between ASTCNN (Fig. a) and ASTGCN (Fig. b) is that ST block in ASTCNN doesn't have GCN layer.

## V. EXPERIMENT

### A. SETTINGS

#### 1) DATASETS

**TaxiBJ:** Dataset I comes from taxicab GPS data and meteorology data in Beijing from four time spans. The data from the last four weeks is chosen as testing data, and all data before that as training data.

**BikeNYC:** Dataset II taken from the New York City (NYC) Bike system in 2014. The last 10 days are chosen as testing data, and the others as training data.

**TaxiNYC:** Dataset III is taxi GPS data for NYC from 1st Jan. 2011 to 30th Jun. 2016. The last four weeks of dataset is test set, and all data before that is the training set.

#### 2) BASELINES

**Historical average (HA):** The historical average predicts the inflow and outflow using the average value of historical flows in the corresponding timestep (i.e., 11:30am-12:00am on

**TABLE 2.** Comparison of baselines, the best results are bolded.

| Method | TaxiBJ | | BikeNYC | | TaxiNYC | |
|---|---|---|---|---|---|---|
| | RMSE | MAPE | RMSE | MAPE | RMSE | MAPE |
| HA | 57.69 | 21.21% | 21.58 | 27.44% | 40.82 | 22.16% |
| DeepST | 18.18 | 25.72% | 7.43 | 24.67% | 27.39 | 21.56% |
| ST-ResNet | 16.89 | 26.69% | 6.33 | 24.02% | 25.78 | 21.35% |
| LSTM | 20.62 | 27.84% | 7.97 | 26.51% | 29.64 | 23.19% |
| GRU | 20.24 | 27.77% | 6.94 | 26.73% | 29.84 | 23.24% |
| FC-LSTM | 18.80 | 25.13% | 6.66 | 23.05% | 27.43 | 22.17% |
| DMVST-Net | 17.13 | 22.37% | 6.58 | 21.89% | 23.49 | 18.83% |
| ASTCNN | **16.24** | **21.03%** | 6.06 | 21.60% | 23.15 | **18.34%** |
| CPPBTR+residual unit | 16.44 | 21.15% | **4.81** | 21.21% | 22.67 | 18.46% |
| CPPBTR+Local-CNN | **16.39** | 21.20% | 4.97 | **21.14%** | **22.41** | 18.39% |
| CPPBTRpure | 17.09 | 22.06% | 5.27 | 22.06% | 23.78 | 19.87% |

Monday, its corresponding periods are all historical timesteps from 11:30am-12:00am on)

**LSTM [4]:** Long-short-term-memory network, which can learn temporal dependency. The best performance of RNN is reported.

**GRU [17]:** Gated-recurrent-unit network. The setting is as same as LSTM.

**FC-LSTM [5]:** Encoder-decoder framework utilizing LSTM. Encoder firstly encodes the previous flows, then the decoder predicts crowd flows.

**ST-ResNet [2]:** A typical domain knowledge-based deep learning method. It extracts three types (trend, period and closeness) of keyframes as input based on domain knowledge to learn temporal representation. CNN with residual block is used to learn spatial representation.

**DMVST-Net [6]:** A deep learning model consists of three views: temporal view (via LSTM), spatial view (via local CNN), semantic view (modeling correlation among regions sharing similar temporal patterns). It achieves state-of-the art results on predicting taxi demand. We directly used the code provided by [6].

**ASTCNN [15]:** Attention based Spatial-Temporal Graph Convolutional Networks (ASTGCN) is a method of handling graph series. ASTGCN is a stack of ST-Block, we removed GCN layers in ST-Block so that it can handle gridded data. The variant of ASTGCN is called ASTCNN.

### 3) VARIANTS OF CPPBTR

CPPBTR aims to learn an effective temporal representation and can be combined with any spatial components. Hence, to make fair comparison, three variants of CPPBTR are introduced in this sub-section.

**CPPBTR + residual unit:** This variant is mainly compared with ST-ResNet. We used residual unit which is as same as ST-ResNet as the spatial component.

**CPPBTR + Local-CNN:** This variant is mainly compared with DMVST-net, the spatial component is as same as DMVST-net.

**CPPBTRpure:** This variant is manly compared with LSTM, GRU and FC-LSTM. We only use CPPBTR without any spatial component since that LSTM and GRU do not have any spatial component.

### 4) HYPERPARAMTERS

We set the layer of PBTR and decoder-stage 2 to 3, set the model dim to 1024 (512 on BikeNYC). Due to insufficient data and limited GPU, no more layers are implemented. An Adam optimizer [11] with a learning rate of 2e-4 is applied to train our model. The length of training data used when training PBTR add CPPBTR is 720 (30days in terms of BikeNYC, 15days in terms of TaxiBJ).

During training, batch size is 128. Due to GPU memory limitation that the local CNN's GPU memory cost is huge, we use gradient accumulation, set accumulate step to 4 and feeds 32 samples at each step. All these experiments were run on 2 NVIDIA P100 GPUs, Keras [18] is used to build our model.

### 5) EVALUATION METRIC

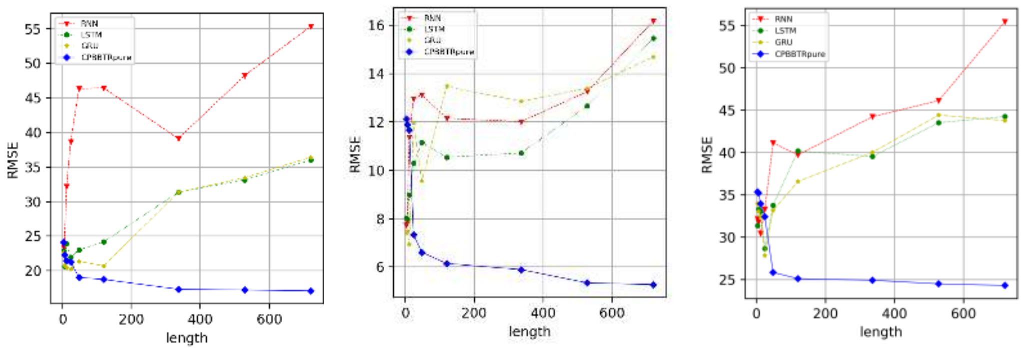
We employ Rooted Mean Square Error (RMSE) and Mean absolute Percentage Error (MAPE) to evaluate our model:

$$RMSE = \sqrt{\frac{1}{N}\sum_i (x_i - \hat{x}_i)^2}, \quad MAPE = \frac{1}{N}\sum_i \left|\frac{x_i - \hat{x}_i}{x_i}\right|$$

## *B. RESULTS*

Table 2 summarizes the results. We can see that our method significantly outperforms all the other baseline methods. In particular, we have the following findings:

1) CPPBTR + residual unit achieves better performance than ST-ResNet. Our model outperforms it by 2.6%/10.3% (RMSE/MAPE) on TaxiBJ, by 24.0%/12.0% (RMSE/MAPE) on BikeNYC and by 12.1%/13.5% (RMSE/MAPE) on TaxiNYC.
2) CPPBTR + Local-CNN works better than all baselines, outperforming DMVST-Net by 4.3%/5.2% (RMSE/MAPE) in terms of TaixBJ, by 24.5%/3.4% (RMSE/MAPE) in terms of BikeNYC and by 4.59%/2.23% (RMSE/MAPE) in terms of TaxiNYC.
3) CPPBTRpure outperforms all the RNN-based models which don't have spatial component. This implies that Transformer-based model is more effective than RNN-based model for capturing temporal dependency of traffic data.



(a) TaxiBJ (b) BikeNYC (c) TaxiNYC

**FIGURE 8. The performance of CPPBTR and baselines with different sequence lengths, which include: the RMSE of the CPPBTR and baselines on TaxiBJ (Fig. a); the RMSE of the CPPBTR and baselines on BikeNYC (Fig. b); the RMSE of the CPPBTR and baselines on TaxiNYC (Fig. c).**

4) ASTCNN is the best performing domain-based knowledge model. It achieves the best performance on both RMSE and MAPE in terms of TaxiBJ, but our methods achieve performances comparable to ASTCNN. The results illustrate that attention mechanism is effective.

We can see that PBTR + residual unit and PBTR + Local-CNN achieve the lowest RMSE and MAPE on BikeNYC and TaxiNYC, only a bit higher RMSE and MAPE than ASTCNN. The reason is that the time granularity of BikeNYC and TaxiBJ are different. The time interval of TaxiBJ is 1 hour, whereas BikeNYC has a time interval of half an hour. Assuming we use data of 28days (4weeks), the length of flow sequnece is 672 in terms of BikeNYC. However, if the sequence length is 672, we can only use data of 14 days (2weeks) in terms of TaxiBJ to train our model. It is hard for our model to learn periodicity and trend of temporal dependency since that the data of 14 days are not enough. However, knowledge-based methods (DeepST, ST-ResNet&ASTCNN) don't suffer from time granularity.

## VI. ANALYSIS

In this section, we conduct experiments to further analyze our model.

### A. THE ABILITY OF MODELING VERY LONG-TERM TEMPORAL DEPENDENCY

In this experiment, CPPBTR is compared to RNNs (vanilla RNN, LSTM and GRU). The comparison is aimed to demonstrate that CPPBTR can model very long-term temporal dependency better than RNNs. The length of input sequence is fixed as one of $\{3, 6, 12, 24, 48, 120, 336, 720\}$. We don't try a sequence longer than 720 since that the GPU resources are limited. From Fig. 7, we can see that as the sequence length goes longer, the RMSE of our model in terms of both datasets decreases. When the sequence length is short (3, 6, 12, 24), the three variants of RNN perform better than CPPBTR, indicating that RNNs can model closeness of temporal dependency very well. However, when the length of input sequence is greater than 120, RNN, LSTM and GRU have very bad performance. This demonstrates that RNNs cannot learn very long-term temporal dependencies of crowd flows, whereas our model can.

**TABLE 3. Performance comparison of Transformer-based methods, the best results are bolded.**

| Method | RMSE | | |
|---|---|---|---|
| | TaxiBJ | BikeNYC | TaxiNYC |
| Vanilla Transformer | 17.17 | 7.64 | 26.15 |
| Transformer-Pretrained | 17.23 | 7.19 | 25.87 |
| BiLSTM-Pretrained | 19.86 | 7.21 | 27.33 |
| CPPBTRpure | **17.09** | **5.27** | **23.78** |

### B. EFFECTIVE OF BIDIRECTIONAL TEMPORAL REPRESENTATION

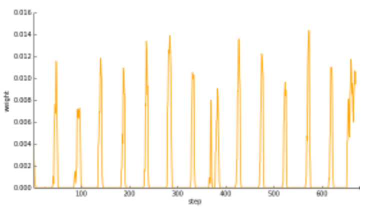
Recall that in our model, we introduce Pre-trained Bidirectional Temporal Representation model (PBTR). To illustrate the effectiveness of the PBTR, we compare the performance of CPPBTR and following methods:

1) Vanilla Transformer: a standard transformer, the settings are as same as CPPBTR, consisting of a 3-layer encoder and a 3-layer decoder.
2) Transformer-Pretrained : its model structure is as same as Vanilla Transformer. The encoder is pre-trained via forward auto-regressive modeling.
3) BiLSTM-Pretrained: a LSTM based encoder-decoder model. The encoder is a pre-trained Bi-LSTM model. Different from PBTR, Bi-LSTM have two LSTM units. One performs forward auto-regressive modeling, and the other performs backward auto-regressive modeling.

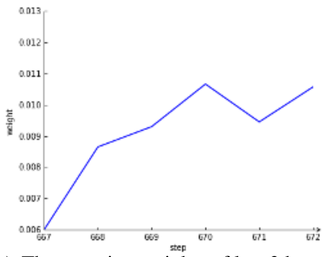
We observe that CPPBTR significantly outperforms other models, demonstrating the effectiveness of the bidirectional temporal representation.

### C. VISUALIZATION OF ATTENTION

A sequence of length 672 (data of 4 weeks) is selected from BikeNYC as input to predict the flow of the 673th timestep.



(a) The global attention distribution.



(b) The attention weights of last 3 hours.

**FIGURE 9. Visualization of the attention distribution.**

Then, we visualized the attention distribution of the decoding side of CPPBTR. Fig. 5 indicates that the time dependency learned by our method is in some respects consistent with domain-based methods, some aspects are inconsistent. We have the following findings:

- **Consistent aspect:** Fig. 8(a) demonstrates that the attention distribution is sparse. The weights of most frames are zero. This is consistent with the domain knowledge-based method, where only a small number of frames (keyframes) contributes to traffic prediction. From Fig. 8(b), we also notice that frames close to the $673^{th}$ timestep are more weighted. It demonstrates that the recent flows are of the great importance to prediction. This is consistent with domain knowledge-based methods. This also shows that our model can model the closeness of temporal dependency.
- **Inconsistent aspect:** Fig. 8(a) shows that the change of attention distribution has regularity and periodicity. It also proves that the proposed method can learn regularity and periodicity of temporal dependency automatically. However, Fig. 8(a) demonstrates that the period of attention weight is approximately 2 days, whereas domain knowledge-based methods assume that the period of flows is one day or one week. The temporal dependency our method learned is not consistent with the domain knowledge.

### D. PREDICTING CROWDS FLOW USING PBTR

CPPBTR achieved good performance. However, the number of parameters of CPPBTR is much more than PBTR. In this sub-section we explore how to use PBTR for traffic prediction directly. The observation of training PBTR shows that the loss of PBTR's on the training set is very small, but the results of test are poor. We believe that this phenomenon is caused by inconsistent testing and training process. During training, PBTR can observe the future information. However,

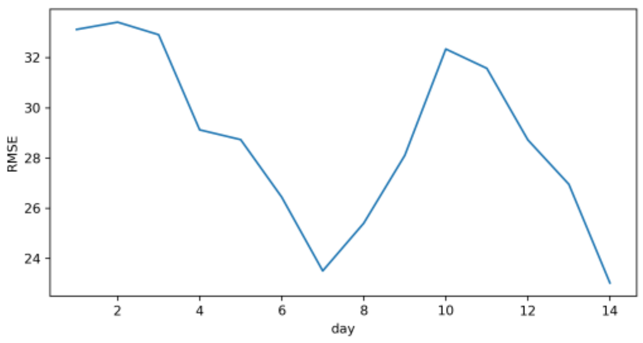


**FIGURE 10. Effect of lengths of l based on RMSE using the dataset TaxiNYC.**

**TABLE 4. Performance comparison of PBTR and PBRT + HA, HA-based data augmentation achieves remarkable results.**

| Method | RMSE | | |
|---|---|---|---|
| | TaxiBJ | BikeNYC | TaxiNYC |
| PBTR | 18.87 | 7.91 | 27.59 |
| PBTR+HA | **16.91** | **5.12** | **23.45** |

only previous crowd flows can be used as input data during inference.

We proposed a data augment method (PBTR + HA) based on historical average (HA) that makes training and test process consistent. Given a previous input sequence (left sequence) $X_L = \{x_{t-l}, x_{t-l+1}, \cdots, x_{t-1}\}$, predict the flow $x_t$ at timestep $t$. According to HA, input sequence on the right side of $x_t$ is calculated as follow:

$$X_R = \frac{1}{N}\sum_{i=1}^{N}\left\{x_{t+1-N*p}, x_{t+2-N*p}, \cdots, x_{t+l-N*p}\right\} \tag{15}$$

where $l = |X_L|$, p is weekly period. Denote masked $x_t$ as $\mathcal{M}_t$ Then we concatenate $X_L$, $M_t$ and $X_R$ to construct the input:

$$\mathrm{I} = \mathrm{concat}\left[X_L, \mathcal{M}_t, X_R\right] \tag{16}$$

Then PBTR predict the crowd flow at timestep $t$, computed as follow:

$$\hat{x}_t = PBTR\,(\mathrm{I}) \tag{17}$$

Table 4 demonstrates that PBTR + HA significantly outperforms PBTR. We believe that PBTR + HA is effective because of two reasons:

- HA makes the process of training consistent with the process of inference.
- HA is suitable because of properties of crowd flows. Crowd flows have regularity and periodicity, and thus, the data generated by HA is a good simulation of future data.

Further, we explored how traffic and history traffic affect predicted value. For a predicted value $x_t$ at timestep $t$, we have: $x_t = a_{t-l}x_{t-l} + a_{t-l+1}x_{t-l+1} + \cdots + a_{t-1}x_{t-1} + a_t x_t^{mask} + a_{t+1}x_{t+1} + \cdots + a_{t+l-1} + \varepsilon$, where $\varepsilon$ is bias, $A_t = \{a_{t-l}, a_{t-l+1}, \ldots, a_{t+l}\}$ is the attention distribution of $x_t$ calculated by the attention mechanism.

$l$ is a hyperparameter. A larger $l$ means more data. In general, more data yield the better performance. However, our experiments find that the length of $l$ has a conditionally positive relation with performance. The results demonstrate that PBTR + HA achieves the best performance when $l = 7$ and $l = 14$. We think this is consistent with the results in subsection VI.C and the views of knowledge-based methods that attention distribution has periodicity and is sparse, Keyframes have the greatest impact on predicated value. For instance, when $l = 10$, the data $\mathcal{D}_{10}$ ( $\mathcal{D}_{10}$ represents data with a sequence length of 10 days) is more than $\mathcal{D}_7$. But the performance is reduced. This phenomenon indicates that $\mathcal{D}_{[8,10]}$ is not critical, and it makes no contribution to traffic prediction. It can be reargued as noise.

## VII. CONCLUSION

In this paper, we proposed a general framework called PBTR (Pre-trained Bidirectional Temporal Representation) to learn the temporal dependency of spatial-temporal data. We also proposed a sequence to sequence framework to predict crowd flows in regular region based on Transformer, including PBTR as the encoder. Experiment results demonstrate that PBTR is capable of modeling very long-term temporal dependency and CPPBTR outperforms all baselines.

In the future, we plan to combine our method with Graph neural networks (GNN), applying PBTR on spatial-temporal graphs such as crowd flows in irregular regions, vehicle speed prediction on road route.

## ACKNOWLEDGMENT

The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

## REFERENCES

[1] J. Zhang, Y. Zheng, D. Qi, R. Li, X. Yi, and T. Li, "Predicting citywide crowd flows using deep spatio-temporal residual networks," *Artif. Intell.*, vol. 259, pp. 147–166, Jun. 2018.

[2] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. 31st AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, 2017, pp. 1655–1661.

[3] V. Ashish, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 5998–6008.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using LSTMs," in *Proc. 32nd Int. Conf. Mach. Learn.*, Lille, France, 2015, pp. 843–852.

[6] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, 2018, pp. 2588–2595.

[7] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "DNN-based prediction model for spatio-temporal data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, Burlingame, CA, USA, 2016, pp. 92:1–92:4.

[8] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015. doi: 10.1109/TITS.2014.2345663.

[9] X. Cheng, R. Zhang, J. Zhou, and W. Xu, "DeepTransport: Learning spatial-temporal dependency for traffic condition forecasting," in *Proc. Int. Joint Conf. Neural Netw.*, Rio de Janeiro, Brazil, Jul. 2018, pp. 1–8.

[10] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. 6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, 2018, pp. 1–16.

[11] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: https://arxiv.org/abs/1810.04805

[13] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, to be published.

[14] J. Sun, J. Zhang, Q. Li, X. Yi, and Y. Zheng, "Predicting citywide crowd flows in irregular regions using multi-view graph convolutional networks," 2019, *arXiv:1903.07789*. [Online]. Available: https://arxiv.org/abs/1903.07789

[15] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. 33rd AAAI Conf. Artif. Intell.*, Honolulu, HI, USA, 2019, pp. 922–929.

[16] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. 3rd Int. Conf. Learn. Represent.*, San Diego, CA, USA, 2015, pp. 1–15.

[17] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2015, *arXiv:1409.1259*. [Online]. Available: https://arxiv.org/abs/1409.1259

[18] Franois Chollet. (2015). *Keras*. [Online]. Available: https://keras.io

[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[20] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proc. SIAM Int. Conf. Data Mining*, Houston, TX, USA, 2017, pp. 777–785.

[21] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, pp. 818–827, Apr. 2017.

[22] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer," 2019, *arXiv:1904.06690*. [Online]. Available: https://arxiv.org/abs/1904.06690

• • •