Domain Adversarial Spatial-Temporal Network: A Transferable Framework for Short-term Traffic Forecasting across Cities

Yihong Tang∗
Department of Urban Planning and Design
University of Hong Kong
Hong Kong SAR, China
yihongt@connect.hku.hk

Ao Qu∗
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Boston, USA
qua@mit.edu

Andy H.F. Chow
Department of Advanced Design and Systems Engineering
City University of Hong Kong
Hong Kong SAR, China
andychow@cityu.edu.hk

William H.K. Lam
Department of Civil and Environmental Engineering
The Hong Kong Polytechnic University
Hong Kong SAR, China
william.lam@polyu.edu.hk

S.C. Wong
Department of Civil Engineering
University of Hong Kong
Hong Kong SAR, China
hhecwsc@hku.hk

Wei Ma†
Department of Civil and Environmental Engineering
Research Institute for Sustainable Urban Development
The Hong Kong Polytechnic University
Hong Kong SAR, China
wei.w.ma@polyu.edu.hk

ABSTRACT
Accurate real-time traffic forecast is critical for intelligent transportation systems (ITS) and it serves as the cornerstone of various smart mobility applications. Though this research area is dominated by deep learning, recent studies indicate that the accuracy improvement by developing new model structures is becoming marginal. Instead, we envision that the improvement can be achieved by transferring the “forecasting-related knowledge” across cities with different data distributions and network topologies. To this end, this paper aims to propose a novel transferable traffic forecasting framework: Domain Adversarial Spatial-Temporal Network (DastNet). DastNet is pre-trained on multiple source networks and fine-tuned with the target network’s traffic data. Specifically, we leverage the graph representation learning and adversarial domain adaptation techniques to learn the domain-invariant node embeddings, which are further incorporated to model the temporal traffic data. To the best of our knowledge, we are the first to employ adversarial multi-domain adaptation for network-wide traffic forecasting problems. DastNet consistently outperforms all state-of-the-art baseline methods on three benchmark datasets. The trained DastNet is applied to Hong Kong’s new traffic detectors, and accurate traffic predictions can be delivered immediately (within one day) when the detector is available. Overall, this study suggests an alternative to enhance the traffic forecasting methods and provides practical implications for cities lacking historical traffic data. Source codes of DastNet are available at https://github.com/YihongT/DASTNet.

KEYWORDS
Traffic Forecasting; Transfer Learning; Domain Adaptation; Adversarial Learning; Intelligent Transportation Systems

∗These authors contributed equally to this work.
†Corresponding author.

Figure 1: An overview of the transferable traffic forecasting problem and its applications.

1 INTRODUCTION
Short-term traffic forecasting [4, 24] has always been a challenging task due to the complex and dynamic spatial-temporal dependencies of the network-wide traffic states. When the spatial attributes and temporal patterns of traffic states are convoluted, their intrinsic interactions could make the traffic forecasting problem intractable. Many classical methods [11, 55] take temporal information into consideration and cannot effectively utilize spatial information. With the rise of deep learning and its application in intelligent
transportation systems (ITS) [2, 10, 65], a number of deep learning components, such as convolutional neural networks (CNNs) [38], graph neural networks (GNNs) [25], and recurrent neural networks (RNNs) [15], are employed to model the spatial-temporal characteristics of the traffic data [6, 12, 19, 27, 46]. These deep learning based spatial-temporal models achieve impressive performances on traffic forecasting tasks.

However, recent studies indicate that the improvement of the forecasting accuracy induced by modifying neural network structures has become marginal [24], and hence it is in great need to seek alternative approaches to further boost up the performance of the deep learning-based traffic forecasting models. One key observation for current traffic forecasting models is that: most existing models are designed for a single city or network. Therefore, a natural idea is to train and apply the traffic forecasting models across multiple cities, with the hope that the “knowledge related to traffic forecasting” can be transferred among cities, as illustrated in Figure 1. The idea of transfer learning has achieved huge success in the area of computer vision, language processing, and so on [30, 39, 43], while the related studies for traffic forecasting are premature [61].

There are few traffic forecasting methods aiming to adopt transfer learning to improve model performances across cities [40, 49, 52, 53, 60]. These methods partition a city into a grid map based on the longitude and latitude, and then rely on the transferability of CNN filters for the grids. However, the city-partitioning approaches overlook the topological relationship of the road network while modeling the actual traffic states on road networks has more practical value and significance. The complexity and variety of road networks’ topological structures could result in untransferable models for most deep learning-based forecasting models [35]. Specifically, we consider the road networks as graphs, and the challenge is to effectively map different road network structures to the same embedding space and reduce the discrepancies among the distribution of node embedding with representation learning on graphs.

As a practical example, Hong Kong is determined to transform into a smart city. The Smart City Blueprint for Hong Kong 2.0 was released in December 2020, which outlines the future smart city applications in Hong Kong [21]. Building an open-sourced traffic data analytic platform is one essential smart mobility application among those applications. Consequently, Hong Kong’s Transport Department is gradually releasing the traffic data starting from the middle of 2021 [22]. As the number of detectors is still increasing now (as shown in Figure 2), the duration of historical traffic data from the new detectors can be less than one month, making it impractical to train an existing traffic forecasting model. This situation also happens in many other cities such as Paris, Shenzhen, and Liverpool [26], as the concept of smart cities just steps into the deployment phase globally. One can see that a successful transferable traffic forecasting framework could enable the smooth transition and early deployment of smart mobility applications.

To summarize, it is both theoretically and practically essential to develop a network-wide deep transferable framework for traffic forecasting across cities. In view of this, we propose a novel framework called Domain Adversarial Spatial-Temporal Network (DASTNet), which is designed for the transferable traffic forecasting problem. This framework maps the raw node features to node embeddings through a spatial encoder. The embedding is induced to be domain-invariant by a domain classifier and is fused with traffic data in the temporal forecaster for traffic forecasting across cities. Overall, the main contributions of our work are as follows:

- We rigorously formulate a novel transferable traffic forecasting problem for general road networks across cities.
- We develop the domain adversarial spatial-temporal network (DASTNet), a transferable spatial-temporal traffic forecasting framework based on multi-domains adversarial adaptation. To the best of our knowledge, this is the first time that the adversarial domain adaption is used in traffic forecasting to effectively learn the transferable knowledge in multiple cities.
- We conduct extensive experiments on three real-world datasets, and the experimental results show that our framework consistently outperforms state-of-the-art models.
- The trained DASTNet is applied to Hong Kong’s newly collected traffic flow data, and the results are encouraging and could provide implications for the actual deployment of Hong Kong’s traffic surveillance and control systems such as Speed Map Panels (SMP) and Journey Time Indication System (JTIS) [48].

The remainder of this paper is organized as follows. Section 2 reviews the related work on spatial-temporal traffic forecasting and transfer learning with deep domain adaptation. Section 3 formulates the transferable traffic forecasting problem. Section 4 introduces details of DASTNet. In section 5, we evaluate the performance of the proposed framework on three real-world datasets as well as the new traffic data in Hong Kong. We conclude the paper in Section 6.

## 2 RELATED WORKS

### 2.1 Spatial-Temporal Traffic Forecasting

The spatial-temporal traffic forecasting problem is an important research topic in spatial-temporal data mining and has been widely studied in recent years. Recently, researchers utilized GNNs [25, 41, 51, 57, 64] to model the spatial-temporal networked data since GNNs are powerful for extracting spatial features from road networks. Most existing works use GNNs and RNNs to learn spatial and temporal features, respectively [66]. STGCN [63] uses CNN to model temporal dependencies. ASTGCN [19] utilizes attention mechanism to capture the dynamics of spatial-temporal dependencies. DCRNN [28] introduces diffusion graph convolution to describe the information diffusion process in spatial networks. DMSTGCN [20] is based on STGCN and learns the posterior graph for one day through back-propagation. [33] exploits both spatial and semantic neighbors of each node by constructing a dynamic weighted

![Figure 2: Available detectors in Hong Kong in September 2021 (left) and January 2022 (right).](image-url)
The main challenge of transfer learning is to effectively reduce the discrepancy in data distributions based on multiple domain discriminators. Particularly, we consider multiple road networks  and propose a domain adversarial domain adaptation method to multi-domain learning scenarios and proposed a rigorous formulation of the transferable traffic forecasting problem. In this section, we first present definitions relevant to our work then employ Maximum Mean Discrepancy (MMD) to improve the feature transferability and learn domain-invariant information. The conventional domain adaptation paradigm transfers knowledge from one source domain to one target domain. In contrast, multi-domain learning refers to a domain adaptation method in which multiple domains' data are incorporated in the training process.

### 2.2 Transfer Learning with Deep Domain Adaptation

The main challenge of transfer learning is to effectively reduce the discrepancy in data distributions across domains. Deep neural networks have the ability to extract transferable knowledge through representation learning methods [62]. [32] and [31] employ Maximum Mean Discrepancy (MMD) to improve the feature transferability and learn domain-invariant information. The conventional domain adaptation paradigm transfers knowledge from one source domain to one target domain. In contrast, multi-domain learning refers to a domain adaptation method in which multiple domains’ data are incorporated in the training process.

In recent years, adversarial learning has been explored for generative modeling in Generative Adversarial Networks (GANS) [17]. For example, Generative Multi-Adversarial Networks (GMANS) [13] extends GANS to multiple discriminators including formidable adversary and forgiving teacher, which significantly eases model training and enhances distribution matching. In [16], adversarial training is used to ensure that learned features in the shared space are indistinguishable to the discriminator and invariant to the shift between domains. [44] extends existing domain adversarial domain adaptation methods to multi-domain learning scenarios and proposed a multi-adversarial domain adaptation (MADA) approach to capture multi-mode structures to enable fine-grained alignment of different data distributions based on multiple domain discriminators.

### 3 PRELIMINARIES

In this section, we first present definitions relevant to our work then rigorously formulate the transferable traffic forecasting problem.

**Definition 1 (Road Network  )**. A road network is represented as an undirected graph  where  is a set of nodes with ,  is a set of edges, and  is the corresponding adjacency matrix of the road network. Particularly, we consider multiple road networks consisting of  source networks and one target network. ,  denotes the  source road network,  denotes the target road network, and we have .

**Definition 2 (Graph Signals  )**. Let , denote the traffic state observed on  as a graph signal with node signals for , where  represents the number of features of each node (e.g., flow, occupancy, speed). Specifically, we use  to denote the observation on road network  at time , and  denotes the observation of node  at time , respectively.

We now define the transferable traffic forecasting problem.

**Definition 3 (Transferable traffic forecasting)**. Given historical graph signals observed on both source and target domains as input, we can divide the transferable traffic forecasting problem into the pre-training and fine-tuning stages.

In the pre-training stage, the forecasting task maps historical node (graph) signals to future node (graph) signals on a source road network , for .

In the fine-tuning stage, to solve the forecasting task  for the same function initialized with parameters , shared from the pre-trained function is fine-tuned to predict graph signals on the target road network, for .

Note that the topology of  can be different from that of  and  represents the process of transferring the learned knowledge  from  to  . How to construct  to make it independent of network topology is the key research question in this study. To this end, the parameter sharing mechanism in the spatial GNNs is utilized to construct  [68]. For the following sections, we consider the study time period:

### 4 PROPOSED METHODOLOGY

In this section, we propose the domain adversarial spatial-temporal network (DASTNet) to solve the transferable traffic forecasting problem. As shown in Figure 3, DASTNet is trained with two stages, and we use two source domains in the figure for illustration. We first perform pre-training through all the source domains in turn without revealing labels from the target domain. Then, the model is fine-tuned on the target domain. We will explain the pre-training stage and fine-tuning stage in detail, respectively.

#### 4.1 Stage 1: Pre-training on Source Domains

In the pre-training stage, DASTNet aims to learn domain-invariant knowledge that is helpful for forecasting tasks from multiple source domains. The learned knowledge can be transferred to improve the traffic forecasting tasks on the target domain. To this end, we design three major modules for DASTNet: spatial encoder, temporal forecaster, and domain classifier.
The spatial encoder aims to consistently embed the spatial information of each node in different road networks. Mathematically, given a node $v$’s raw feature $e_v \in \mathbb{R}^{D_e}$, in which $D_e$ is the dimension of raw features for each node, the spatial encoder maps it to a $D_t$-dimensional node embedding $f_v \in \mathbb{R}^{D_t}$, i.e., $f_v = M_e(e_v; \theta_e)$, where the parameters in this mapping $M_e$ are denoted as $\theta_e$. Note that the raw feature of a node can be obtained by a variety of methods (e.g., POI information, GPS trajectories, geo-location information, and topological node representations).

Given a learned node embedding $f_v$ for network $G_t$, the temporal forecaster fulfils the forecasting task $\mathcal{T}_t$, presented in Equation 1 by mapping historical node (graph) signals to the future node (graph) signals, which can be summarized by a mapping $(\hat{X}_t^D, \cdots, \hat{X}_t^H) = M_t((X^D_{t-H+1}), \cdots, X^H_t), f_v; \theta_t), \forall v \in V_{G_t}$, and we denote the parameters of this mapping $M_t$ as $\theta_t$.

Domain classifier takes node embedding $f_v$ as input and maps it to the probability distribution vector $\tilde{d}_v$ for domain labels, and we use notation $d_v$ to denote the one-hot encoding of the actual domain label of $f_v$. Note that the domain labels include all the source domains and the target domain. This mapping is represented as $\tilde{d}_v = M_d(f_v; \theta_d)$. We also want to make the node embedding $f_v$ domain-invariant. That means, under the guidance of the domain classifier, we expect the learned node embedding $f_v$ is independent of the domain label $d_v$.

At the pre-training stage, we seek the parameters $(\theta_e, \theta_t)$ of mappings $(M_e, M_t)$ that minimize the loss of the temporal forecaster, while simultaneously seeking the parameters $\theta_d$ of mapping $M_d$ that maximize the loss of the domain classifier so that it cannot identify original domains of node embeddings learned from spatial encoders. Note that the target domain’s node embedding is involved in the pre-training process to guide the target spatial encoder to learn domain-invariant node embeddings. Then we can define the loss function of the pre-training process as:

$$L(\cdot; \theta_e, \theta_t, \theta_d) = L^{src}(\cdot; \theta_e, \theta_t) + \lambda L^{adv}(\cdot; \theta_e, \theta_d)$$

$$= L^{src}(M_t(f_v; (e_v; \theta_e); \theta_t), \cdot) + \lambda L^{adv}(M_d(f_v; e_v; \theta_d), d_v)$$

where $\lambda$ trades off the two losses. $L^{src}(\cdot, \cdot)$ represents the prediction error on source domains and $L^{adv}(\cdot, \cdot)$ is the adversarial loss for domain classification. Based on our objectives, we are seeking the parameters $(\tilde{\theta}_e, \tilde{\theta}_t, \tilde{\theta}_d)$ that reach a saddle point of $L$:

$$\left(\tilde{\theta}_e, \tilde{\theta}_t\right) = \arg \min_{\theta_e, \theta_t} L(\cdot; \theta_e, \theta_t, \tilde{\theta}_d),$$

$$\tilde{\theta}_d = \arg \max_{\theta_d} L(\cdot; \tilde{\theta}_e, \tilde{\theta}_t, \theta_d).$$

Equation 4 essentially represents the min-max loss for GANS, and the following sections will discuss the details of each component in the loss function.

4.1.1 Spatial Encoder. In traffic forecasting tasks, a successful transfer of trained GNN models requires the adaptability of graph topology change between different road networks. To solve this issue, it is important to devise a graph embedding mechanism that can capture generalizable spatial information regardless of domains. To this end, we generate the raw feature $e_v$ for a node $v$ by node2vec [18] as the input of the spatial encoder. Raw node features learned from the node2vec can reconstruct the “similarity” extracted from random walks since nodes are considered similar to each other if they tend to co-occur in these random walks. In addition to modeling the similarity between nodes, we also want to learn localized node features to identify the uniqueness of the local topology around nodes. [58] proves that graph isomorphic network (GIN) layer is as powerful as the Weisfeiler-Lehman (WL) test [54] for distinguishing different graph structures. Thus, we adopt GIN layers with mean aggregators proposed in [58] as our
We note that previous studies mainly use GPS trajectories to learn where $\mathbf{u}$ is a trainable parameter, $k = 1, \ldots, K$, and $K$ is the total number of layers in $\mathbf{G}_n$. The node $\mathbf{v}$’s embedding can be obtained by $\mathbf{f}_v = \mathbf{f}_v^{(K)}$. We note that previous studies mainly use GPS trajectories to learn the location embedding [7, 56], while this study utilizes graph topology and aggregate traffic data.

4.1.2 Temporal Forecaster. The learned node embedding $\mathbf{f}_v$ will be involved in the mapping $\mathbf{M}_f$ to predict future node signals. Now we will introduce our temporal forecaster, which aims to model the temporal dependencies of traffic data. Thus, we adapted the Gated Recurrent Units (GRU), which is a powerful RNN variant [9, 15]. In particular, we extend GRU by incorporating the learned node embedding $\mathbf{f}_v$ into its updating process. To realize this, the learned node embedding $\mathbf{f}_v$ is concatenated with the hidden state of GRU (we denote the hidden state for node $\mathbf{v}$ at time $\tau$ as $h_v^{(\tau)}$). Details of the mapping $\mathbf{M}_f$ is shown below:

$$u_v^{(\tau)} = \sigma(\Theta_u [X_v^{(\tau)}; h_v^{(\tau-1)}] + b_u),$$

$$r_v^{(\tau)} = \sigma(\Theta_r [X_v^{(\tau)}; h_v^{(\tau-1)}] + b_r),$$

$$c_v^{(\tau)} = \tanh[\Theta_c [X_v^{(\tau)}; r_v^{(\tau)} \odot h_v^{(\tau-1)}] + b_c],$$

$$h_v^{(\tau)} = \text{MLP}_{\text{gra}}(\mathbf{f}_v, u_v^{(\tau)} \odot h_v^{(\tau-1)} + (1 - u_v^{(\tau)} \odot c_v^{(\tau)})).$$

where $u_v^{(\tau)}$, $r_v^{(\tau)}$, $c_v^{(\tau)}$ are update gate, reset gate and current memory content respectively. $\Theta_u$, $\Theta_r$, and $\Theta_c$ are parameter matrices, and $b_u$, $b_r$, and $b_c$ are bias terms.

The pre-training stage aims to minimize the error between the actual value and the predicted value. A single-layer perceptron is designed as the output layer to map the temporal forecaster’s output $\hat{h}_v^{(\tau)}$ to the final prediction $\hat{X}_v^{(\tau)}$. The source loss is represented by:

$$\mathcal{L}^\text{src} = \frac{1}{H} \sum_{t=1}^{t+H} \frac{1}{|N_{\mathbf{v}_t}|} \sum_{\mathbf{v} \in V_{\mathbf{v}_t}} \left\| \mathbf{\hat{X}}_v^{(\tau)} - X_v^{(\tau)} \right\|_1.$$  

4.1.3 Domain Classifier. The difference between domains is the main obstacle in transfer learning. In the traffic forecasting problem, the primary domain difference that leads to the model’s inability to conduct transfer learning between different domains is the spatial discrepancy. Thus, spatial encoders are involved in learning domain-invariant node embeddings for both source networks and the target network in the pre-training process.

To achieve this goal, we involve a Gradient Reversal Layer (GRL) [16] and a domain classifier trained to distinguish the original domain of node embedding. The GRL has no parameters and acts as an identity transform during forward propagation. During the backpropagation, GRL takes the subsequent level’s gradient, and passes its negative value to the preceding layer. In the domain classifier, given an input node embedding $\mathbf{f}_v$, $\theta_d$ is optimized to predict the correct domain label, and $\theta_d$ is trained to maximize the domain classification loss. Based on the mapping $\mathbf{d}_v = \mathbf{M}_d(\mathbf{f}_v; \theta_d) = \text{Softmax}(\text{MLP}_d(\mathbf{f}_v))$, $\mathcal{L}^\text{adv}$ is defined as:

$$\mathcal{L}^\text{adv} = \frac{1}{|V_{\text{all}}|} \sum_{\mathbf{v} \in V_{\text{all}}} \frac{1}{|V_{1}|} \sum_{\mathbf{v} \in V_{1}} \langle \mathbf{d}_v, \log \text{Softmax}(\text{MLP}_d(\mathbf{f}_v)) \rangle,$$

4.2 Stage 2: Fine-tuning on the Target Domain

The objective of the fine-tuning stage is to utilize the knowledge gained from the pre-training stage to further improve forecasting performance on the target domain. Specifically, we adopt the parameter sharing mechanism [39]: the parameters of the target spatial encoder and the temporal forecaster in the fine-tuning stage are initialized with the parameters trained in the pre-training stage.

Moreover, we involve a private spatial encoder combined with the pre-trained target spatial encoder to explore both domain-invariant and domain-specific node embeddings. Mathematically, given a raw node feature $e_v$, the private spatial encoder maps it to a domain-specific node embedding $\mathbf{f}_v$, this process is represented as $\mathbf{f}_v = \mathbf{M}_e(e_v; \theta_e)$, where $\mathbf{M}_e$ has the same structure as $\mathbf{M}_s$, and the parameter $\theta_e$ is randomly initialized. The pre-trained target spatial encoder maps the raw node feature $e_v$ to a domain-invariant node embedding $\mathbf{f}_v$, i.e., $\mathbf{f}_v = \mathbf{M}_s(e_v; \theta_{s1}, \theta_{s2})$, where $\theta_{s1}$ and $\theta_{s2}$ means that $\theta_{s1}$ is initialized with the trained parameter $\theta_{s}$ in the pre-training stage. Note that $\mathbf{M}_e$ and $\mathbf{M}_s$ are of the same structure, and the process to generate $\mathbf{f}_v$ and $\mathbf{f}_e$ is the same as in Equation 5.

Before being incorporated into the pre-trained temporal forecaster, $\mathbf{f}_e$ and $\mathbf{f}_v$ are combined by MLP layers to learn the combined node embedding $\mathbf{f}_v^{\text{tar}}$ of the target domain:

$$\mathbf{f}_v^{\text{tar}} = \text{MLP}_{\text{emb}}(\text{MLP}_{\text{pre}}(\mathbf{f}_e) + \text{MLP}_{\text{pre}}(\mathbf{f}_v)),$$

then given node signal $X_v^{(\tau)} (v \in V_{\text{tgt}})$ at time $\tau$ and $\mathbf{f}_v^{\text{tar}}$ as input, $\tilde{X}_v^{(\tau)}$ is computed based on Equation 6, 7, 8, and 9. We denote the target loss at the fine-tuning stage as:

$$\mathcal{L}^{\text{tar}} = \frac{1}{H} \sum_{t=1}^{t+H} \frac{1}{|N_{\mathbf{v}_t}|} \sum_{\mathbf{v} \in V_{\mathbf{v}_t}} \left\| \mathbf{\hat{X}}_v^{(\tau)} - X_v^{(\tau)} \right\|_1.$$  

5 EXPERIMENTS

We first validate the performance of DastNet using benchmark datasets, and then DastNet is experimentally deployed with the newly collected data in Hong Kong.
5.1 Offline Validation with Benchmark Datasets

We evaluate the performance of DastNet on three real-world datasets, PEM504, PEM507, PEM508, which are collected from the Caltrans Performance Measurement System (PEMS) [37] every 30 seconds. There are three kinds of traffic measurements in the raw data: speed, flow, and occupancy. In this study, we forecast the traffic flow for evaluation purposes and it is aggregated to 5-minute intervals, which means there are 12 time intervals for each hour and 288 time intervals for each day. The unit of traffic flow is veh/hour (vph). The within-day traffic flow distributions are shown in Figure 4. One can see that flow distributions vary significantly over the day for different datasets, and hence domain adaption is necessary.

The road network for each dataset is constructed according to actual road networks, and we defined the adjacency matrix based on connectivity. Mathematically, \( A_{i,j} = \begin{cases} 1, & \text{if } v_i \text{ connects to } v_j \\ 0, & \text{otherwise} \end{cases} \), where \( v_i \) denotes node \( i \) in the road network. Moreover, we normalize the graph signals by the following formula: \( X = \frac{X - \text{mean}(X)}{\text{std}(X)} \), where function mean and function std calculate the mean value and the standard deviation of historical traffic data respectively.

5.1.1 Baseline Methods.

- **HA [29]**: Historical Average method uses average value of historical traffic flow data as the prediction of future traffic flow.
- **SVR [45]**: Support Vector Regression adopts support vector machines to solve regression tasks.
- **GRU [8]**: Gated Recurrent Unit (GRU) is a well-known variant of RNN which is powerful at capturing temporal dependencies.
- **GCN [25]**: Graph Convolutional Network can handle arbitrary graph-structured data and has been proved to be powerful at capturing spatial dependences.
- **TGCN [66]**: Temporal Graph Convolutional Network performs stably well for short-term traffic forecasting tasks.
- **STGCN [63]**: Spatial-Temporal Graph Convolutional Network uses ChebNet and 2D convolutions for traffic prediction.
- **DCRNN [28]**: Diffusion Convolutional Recurrent Neural Network combines GNN and RNN with diffusion convolutions.
- **AGCRN [1]**: Adaptive Graph Convolutional Recurrent Network learns node-specific patterns through node adaptive parameter learning and data-adaptive graph generation.
- **STGODE [14]**: Spatial-Temporal Graph Ordinary Differential Equation Network capture spatial-temporal dynamics through a tensor-based ODE.

To demonstrate the effectiveness of each key module in DastNet, we compare with some variants of DastNet as follows:

- **TEMPORAL FORECASTER**: DastNet with only the TEMPORAL FORECASTER component. This method only uses graph signals as input for pre-training and fine-tuning.
- **TARGET ONLY**: DastNet without training at the pre-training stage. The comparison with this baseline method demonstrate the merits of training on other data sources.
- **DASTNET w/o DA**: DastNet without the adversarial domain adaptation (domain classifier). The comparison with this baseline method demonstrate the merits of domain-invariant features.
- **DASTNET w/o PRI**: DastNet without the private encoder at the fine-tuning stage.

Above variants’ other settings are the same as DastNet.

5.1.2 Experimental Settings. To simulate the lack of data, for each dataset, we randomly select ten consecutive days’ traffic flow data from the original training set as our training set, and the validation/testing sets are the same as [27]. We use one-hour historical traffic flow data for training and forecasting traffic flow in the next 15, 30, and 60 minutes (horizon=3, 6, and 12, respectively). For one dataset \( D \), DastNet-related methods and Trans GRU are pre-trained on the other two datasets, and fine-tuned on \( D \). Other methods are trained on \( D \). All experiments are repeated 5 times. Other hyper-parameters are determined based on the validation set. We implement our framework based on Pytorch [42] on a virtual workstation with two 11G memory Nvidia GeForce RTX 2080Ti GPUs. To suppress noise from the domain classifier at the early stages of the pre-training procedure, instead of fixing the adversarial domain adaptation loss factor \( \mathcal{F} \). We gradually change it from 0 to 1: \( \mathcal{F} = \frac{2}{\text{current step}} \cdot \eta - 1 \), where \( \mathcal{F} = \frac{2}{\text{total steps}} \cdot \eta \) was set to 10 in all experiments. We select the SGD optimizer for stability and set the maximum epochs for fine-tuning stage to 2000 and set K of Gin encoders as 1 and 64 as the dimension of node embedding. For all model we set 64 as the batch size. For node2vec settings, we set \( p = q = 1 \), and each source node conduct 200 walks with 8 as the walk length and 64 as the embedding dimension.

Table 1 shows the performance comparison of different methods for traffic flow forecasting. Let \( x_i \in X \) denote the ground truth and \( \hat{x}_i \) represent the predicted values, and \( \Omega \) denotes the set of training samples’ indices. The performance of all methods are evaluated based on (1) Mean Absolute Error (\( \text{MAE}(x, \hat{x}) = \frac{1}{|\Omega|} \sum_{i \in \Omega} |x_i - \hat{x}_i| \)), which is a fundamental metric to reflect the actual situation of the prediction accuracy. (2) Root Mean Squared Error (\( \text{RMSE}(x, \hat{x}) = \sqrt{\frac{1}{|\Omega|} \sum_{i \in \Omega} (x_i - \hat{x}_i)^2} \)), which is more sensitive to abnormal values. (3) Mean Absolute Percentage Error (\( \text{MAPE}(x, \hat{x}) = \frac{1}{|\Omega|} \sum_{i \in \Omega} \left| \frac{x_i - \hat{x}_i}{x_i} \right| \)). It can be seen that DastNet achieves the state-of-the-art forecasting performance on the three datasets for all evaluation metrics and all prediction horizons. Traditional statistical methods like HA and SVR are less powerful compared to deep learning methods such as GRU. The performance of GCN is low, as it overlooks the temporal patterns of the data. DastNet outperforms existing spatial-temporal models like Tgcn, TGCN, DCRNN, AGCRN and the state-of-the-art method STGODE.
DastNet achieves approximately 9.4%, 8.6% and 10.9% improvements compared to the best baseline method in MAE, RMSE, MAPE, respectively. Table 2 summarize the improvements of our methods, where "−" denotes no improvements.

Table 2: Comparison between 1) GRU and the TARGET ONLY (Upper); 2) DastNet and the best baseline (Lower).

| Methods      | 15min MAE | 30min MAE | 60min MAE | 15min RMSE | 30min RMSE | 60min RMSE | 15min MAPE(%) | 30min MAPE(%) | 60min MAPE(%) |
|--------------|-----------|-----------|-----------|------------|------------|------------|----------------|----------------|----------------|
| 15min        |           |           |           |            |            |            |                 |                 |                 |
| GRU          |           |           |           |            |            |            |                 |                 |                 |
| DastNet      |           |           |           |            |            |            |                 |                 |                 |
| GRU          |           |           |           |            |            |            |                 |                 |                 |
| DastNet      |           |           |           |            |            |            |                 |                 |                 |

DastNet outperforms the best baseline method in terms of MAE, RMSE, MAPE, with improvements of approximately 9.4%, 8.6% and 10.9%, respectively.

Effects of Domain Adaptation. To demonstrate the effectiveness of the proposed adversarial domain adaptation module, we visualize the raw feature of the node $e_n$ (generated from node2vec) and the learned node embedding $f_n$ (generated from spatial encoders) in Figure 5 using t-SNE [50]. As illustrated, node2vec learns graph connectivity for each specific graph, and hence the raw features are separate in Figure 5. In contrast, the adversarial training process
Figure 5: Visualization of $e_v$ and $f_v$ by t-SNE. Successfully guides the spatial encoder to learn more uniformly distributed node embeddings on different graphs.

Figure 6: Sensitivity analysis, future 30-minute traffic flow forecasting results under different training set sizes. Sensitivity Analysis. To further demonstrate the robustness of DastNet, we conduct additional experiments with different sizes of training sets. We change the number of days of traffic flow data in the training set. To be more specific, we use four training sets with 1 day, 10 days, 30 days and all data, respectively. Then we compare DastNet with Stgcn and Tgcn. The performance of Dcrnn degrades drastically when the training set is small. To ensure the complete display in the Figure, we do not include it in the comparison and we do not include Stgode because of its instability. We measure the performance of DastNet and the other two models on PEMS04, PEMS07, and PEMS08, by changing the ratio (measured in days) of the traffic flow data contained in the training set. Experimental results of the sensitivity analysis are provided in Figure 6. In most cases, we can see that Stgcn and Tgcn underperform HA when the training set is small. On the contrary, DastNet consistently outperforms other models in predicting different future time intervals of all datasets. Another observation is that the improvements over baseline methods are more significant for few-shot settings (small training sets). Specifically, the approximate gains on MAE decrease are 42.1% / 23.3% / 14.7% / 14.9% on average for 1/10/30/all days for training compared with Tgcn and 46.7%/35.7%/30.7%/34% compared with Stgcn.

Figure 7: Visualization of the predicted flow. Case Study. We randomly select six detectors and visualize the predicted traffic flow sequences of DastNet and Stgcn follow the setting in [14], and the visualizations are shown in Figure 7. Ground true traffic flow sequence is also plotted for comparison. One can see that the prediction generated by DastNet are much closer to the ground truth than that by Stgcn. Stgcn could accurately predict the peak traffic, which might be because DastNet learns the traffic trends from multiple datasets and ignores the small oscillations that only exist in a specific dataset.

5.2 Experimental Deployment in Hong Kong By the end of 2022, we aim to deploy a traffic information provision system in Hong Kong using traffic detector data on strategic routes from the Transport Department [23]. The new system could supplement the existing Speed Map Panels (SMP) and Journey Time Indication System (JTIS) by employing more reliable models and real-time traffic data. For both systems, flow data is essential and collected from traffic detectors at selected locations for the automatic incident detection purpose, as the JTIS and SMP make use of the flow data to simulate the traffic propagation, especially after car crashes [48]. Additionally, DastNet could be further extended for speed forecasting. As we discussed in Section 1, the historical traffic data for the new detectors in Hong Kong are very limited. Figure 8 demonstrates: a) the spatial distribution of the newly deployed detectors in January 2022 and b) the corresponding traffic flow in Hong Kong. After the systematic process of the raw data as presented in c), traffic flow on the new detectors can be predicted and fed into the downstream applications once the detector is available. We use the traffic flow data from three PEMS datasets for pre-training, and use Hong Kong’s traffic flow data on January 10,
2022 to fine-tune our model. All Hong Kong’s traffic flow data on January 11, 2022 are used as the testing set. We use 614 traffic detectors (a combinations of video detectors and automatic licence plate recognition detectors) to collect Hong Kong’s traffic flow data for the deployment of our system, and the raw traffic flow data is aggregated to 5-minute intervals. We construct Hong Kong’s road network $G_{HK}$ based on distances between traffic detectors and define the adjacency matrix through connectivity. Meanwhile, HA and spatial-temporal baseline methods Tgcn, STGcn and STGode are adopted for comparisons. All experiments are repeated for 5 times, and the average results are shown in Table 3. One can read from the table that, with the trained DastNet from other datasets, accurate traffic predictions can be delivered to the travelers immediately (after one day) when the detector data is available.

Table 3: Performance comparison on the newly collected data in Hong Kong.

|       | 15min    | 30min    | 60min    |
|-------|----------|----------|----------|
| HK    | MAE 15.79 RMSE 23.95 MAPE 16.96% | MAE 17.84 RMSE 27.00 MAPE 18.70% | MAE 21.66 RMSE 33.41 MAPE 22.42% |
| Ha    | Tgcn     | STGcn    | STGode   |
|       | 22.39 30.50 27.54% | 22.39 30.48 26.76% | 25.95 35.61 27.98% |
|       | 39.86 55.79 46.80% | 39.34 55.34 45.62% | 42.52 58.95 52.94% |
|       | 63.46 86.08 54.77% | 66.19 87.36 60.23% | 66.76 92.83 58.65% |
| DastNet| 11.71 17.69 12.89% | 13.87 21.25 14.91% | 17.09 26.47 18.24% |

6 CONCLUSION

In this study, we formulated the transferable traffic forecasting problem and proposed an adversarial multi-domain adaptation framework named Domain Adversarial Spatial-Temporal Network (DastNet). This is the first attempt to apply adversarial domain adaptation to network-wide traffic forecasting tasks on the general graph-based networks to the best of our knowledge. Specifically, DastNet is pre-trained on multiple source datasets and then fine-tuned on the target dataset to improve the forecasting performance. The spatial encoder learns the uniform node embedding for all graphs, the domain classifier forces the node embedding domain-invariant, and the temporal forecaster generates the prediction. DastNet obtained significant and consistent improvements over baseline methods on benchmark datasets and will be deployed in Hong Kong to enable the smooth transition and early deployment of smart mobility applications.

We will further explore the following aspects for future work:

1. Possible ways to evaluate, reduce and eliminate discrepancies of time-series-based graph signal sequences across different domains.
2. The effectiveness of the private encoder does not conform to domain adaptation theory [3], and it is interesting to derive theoretical guarantees for the necessity of the private encoder on target domains. In the experimental deployment, we observe that the performance of existing traffic forecasting methods degrades drastically when the traffic flow rate is low. However, this situation is barely covered in the PEMS datasets, which could potentially make the current evaluation of traffic forecasting methods biased.
3. The developed framework can potentially be utilized to learn the node embedding for multi-tasks, such as forecasting air pollution, estimating population density, etc. It would be interesting to develop a model for a universal location embedding [5], which is beneficial for different types of location-related learning tasks [7, 56].

ACKNOWLEDGMENTS

This study was supported by the Research Impact Fund for “Reliability-based Intelligent Transportation Systems in Urban Road Network with Uncertainty” and the Early Career Scheme from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU R5029-18 and PolyU/25209221), as well as a grant from the Research Institute for Sustainable Urban Development (RISUD) at the Hong Kong Polytechnic University (Project No. P0038288). The authors thank the Transport Department of the Government of the Hong Kong Special Administrative Region for providing the relevant traffic data.
[53] Senzhang Wang, Hao Miao, Jiyue Li, and Jianmong Cao. 2021. Spatio-Temporal Knowledge Transfer for Urban Crowd Flow Prediction via Deep Attentive Adaptation Networks. *IEEE Transactions on Intelligent Transportation Systems* (2021).

[54] Boris Weisfeiler and Andrei Leman. 1968. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series 2, 9* (1968), 12–16.

[55] Billy M Williams and Lester A Hoel. 2003. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of transportation engineering* 129, 6 (2003), 664–672.

[56] Bing Yu, Haoteng Yin, and Zhanying Zhu. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875* (2017).

[57] Xiyue Zhang, Chao Huang, Yong Xu, and Liangzhao Xiao. 2020. Spatial-temporal convolutional graph attention networks for citywide traffic flow forecasting. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1853–1862.

[58] Yongxin Yang and Timothy M Hospedales. 2014. A unified perspective on multi-domain and multi-task learning. *arXiv preprint arXiv:1412.7489* (2014).

[59] Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. 2020. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 1234–1241.

[60] Jie Zhou, Ganqui Cui, Shengdeng Hu, Zhongnan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open* 1 (2020), 57–81.