Spatiotemporal Ego-Graph Domain Adaptation for Traffic Prediction With Data Missing

Chen Xu, Qiang Wang, *Member, IEEE*, Wenqi Zhang, *Member, IEEE*, and Chen Sun, *Senior Member, IEEE*

***Abstract*— As an important research field in time series processing, traffic prediction has a profound impact on people's daily lives and social development. Conventional traffic prediction relies on complete observation data. However, data missing is common in cities due to equipment failure, network interruption, etc., which poses a huge obstacle to traffic prediction. In this paper, we design a novel Spatiotemporal Ego-graph Domain Adaptation framework (SEDA) to predict traffic state in data missing scenarios. Based on the multi-dimensional topological information of local network (ego-graph), isomorphic ego-graphs are aligned across the missing data in target domain and the external data in source domain to obtain alternative data. Furthermore, a Dual-branch Cross reCoupling method (DCC) is proposed to reconstruct missing features according to the alternative data. Experimental results on real public datasets with 10%-40% missing show that SEDA averagely outperforms both the state-of-the-art knowledge transfer-based prediction baselines and the incomplete data prediction baselines by more than 0.45% and 0.86%. Ablation experiments and visualization analysis further demonstrate the effectiveness of SEDA components.**

***Index Terms*— Traffic prediction, spatiotemporal data modeling, traffic data missing, domain adaptation.**

I. Introduction

EMERGING Internet of Vehicles and automatic driving technologies put higher requirements on traffic data processing in Intelligent Transportation Systems (ITS). Vehicle sensors and loop detectors can collect a large amount of traffic observation data, but more reliable and efficient models are needed to process these large-scale spatiotemporal data. Complete and accurate traffic prediction data can boost social efficiency [1], [2], [3] and help urban management [4], [5], [6]. Hence, traffic prediction has attracted more attention in recent years.

The main characteristics of conventional traffic prediction models are data-driven and dimension-fixed. They are trained and applied in a single road network with complete data. For instance, Zhao et al. [7] combined Gated Recurrent Unit (GRU) [8] and Graph Convolution Network (GCN) [9] to jointly capture spatial and temporal dependence of traffic data in a road network. Li et al. [10] proposed a Diffusion Convolution Recurrent Neural Network (DCRNN), which introduces bidirectional graph random walks and integrates the encoder-decoder architecture and the scheduled sampling technique. Bai et al. [11] proposed an Adaptive Graph Convolutional Recurrent Network (AGCRN) to learn nodes embedding vectors and adaptively construct graph structure, which avoids the errors caused by hand-crafted adjacency matrixes. These models require a prior determination of the number of nodes or structures (adjacency matrix) of road network, and use complete historical data to optimize model parameters. However, complete observation data is not always available in reality. Data missing is an inevitable obstacle in traffic prediction.

Traffic observation data missing is a common problem in most cities due to loop detector failure and transmission loss [12], [13], [14]. In Alberta [15] and Texas [16], the mean annual missing rate is nearly 50%, and the highest missing rate even reaches 93%. In this scenario, conventional traffic prediction models cannot work well, and even break down when the missing rate gets high. To handle this problem, some work focused on missing data imputation. They use tensor decomposition [14], deep neural networks [17], and so on [18], [19], and [20] to fill in the missing data, then put the imputed data into conventional traffic prediction models. But this "Imputation + Prediction" mode may cause the accumulation of errors. Furthermore, some existing work attempted to make predictions directly based on incomplete data. They use neural networks to extract features from remaining data [21] or make predictions based on the tensor decomposition of the incomplete data [22], [23]. All of those direct/indirect prediction methods must follow ***two basic assumptions*** [24], [25]: i) The training data and testing data must be in the same feature space with independent and identical distribution, ii) the training data used to calibrate prediction models should be sufficient and representative, containing almost all the features in the feature space. However, when the missing rate is high, the remaining data cannot cover the entire feature space. The above existing work is limited to obtaining the data features from a single source and poses a high risk of model collapse.

Manuscript received 3 December 2023; revised 7 May 2024 and 7 July 2024; accepted 18 August 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62071066 and in part by the Fundamental Research Funds for Central Universities under Grant 2242022k60006. The Associate Editor for this article was Y. Hou. *(Corresponding author: Qiang Wang.)*

Chen Xu and Qiang Wang are with the National Engineering Research Center of Mobile Network Technologies, School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: xuchen@bupt.edu.cn; wangq@bupt.edu.cn).

Wenqi Zhang and Chen Sun are with the Sony (China) Research Laboratory, Beijing 100027, China (e-mail: wenqi.zhang@sony.com; chen.sun@sony.com).

Digital Object Identifier 10.1109/TITS.2024.3447549

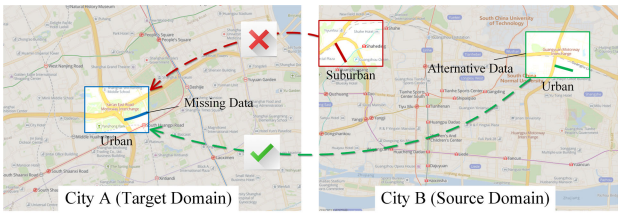


Fig. 1. Alternative data matching. Data missing occurs in City A, the similar data subsets are expected to be matched in external data source City B.

Motivated by achieving better model performance under a high missing rate, we introduce the knowledge from external datasets to assist prediction of the target dataset. From the theoretical perspective, Domain Adaptation [26] provides support for the neural networks achieving out-of-distribution generalization and cross-domain knowledge transfer. Barabasi's research on complex networks [27] reveals the correlation between the dynamic characteristics of observation and the topology of the network, which inspires us to use topological information to attenuate the effect of observations missing. However, in a high missing rate scenario, how to achieve domain adaptation based on topological information is a problem to be solved.

Specifically, two main challenges need to be addressed. ***The first challenge*** is how to find the alternative data for the missing data from an external source. An external source contains a wide variety of data (e.g. from urban, suburban). Only the data subset similar to the missing data is appropriate to supplement the missing features as shown in Figure 1. ***The second challenge*** is how to reconstruct the missing features based on the alternative data. The alternative data contains complex domain-specific factors of external sources (e.g., distribution, scale). Directly using it in reconstructing missing features will result in distortion.

To solve the above challenges, we design a novel Spatiotemporal Ego-graph Domain Adaptation framework (SEDA) to predict traffic state in data missing scenarios. Specifically, we construct multi-dimensional topological vectors to align isomorphic ego-graphs across the target domain and external source domain to obtain alternative data. Furthermore, we propose a Dual-branch Cross reCoupling method (DCC) to reconstruct missing features, where the external domain-specific factors are decoupled out and uninvolved in reconstructing. The main contributions are as follows:

- We design a novel Spatiotemporal Ego-graph Domain Adaptation framework (SEDA) that directly makes traffic prediction in data missing scenarios by introducing knowledge of external sources.
- We propose an ego-graphs cross-domain alignment method based on multi-dimensional topological information to match alternative data of missing features, which is not disturbed by data missing.
- We propose a Dual-branch Cross reCoupling method (DCC) to reconstruct missing features. It symmetrically decouples the features of alternative data into inter-domain and intra-domain branches, then cross recouples two branches with isolating external domain specificity.
- Experimental results on real public datasets show that SEDA averagely outperforms both the state-of-the-art incomplete data prediction baselines and the knowledge transfer-based prediction baselines with "Imputation + Prediction" mode more than 0.86% and 0.45%.

The rest of this paper is organized as follows: Section II is Related Work. Section III is Definition and Problem Formulation. Section IV is the Methodology. Section V is the Experiments of our proposed model on real datasets and analysis of results. Section VI is the Conclusion.

II. Related Work

In this section, we survey the work related to traffic prediction, covering both deterministic and probabilistic models, as the propaedeutics of our work. We summarized the comparison of the traffic prediction related work as shown in Table I.

A. Traffic Prediction With Incomplete Data

The problem of traffic data missing in the real-world is almost inevitable due to various reasons such as equipment failure, human error, and network communication problems. Two strategies have been proposed to solve the traffic prediction problem with missing data. ***The first strategy*** is to conduct data imputation before prediction. In [18], Wang et al. designed a time-series constraint and an adaptive Laplacian regularization spatial constraint to capture the relationship between road links based on low-rank matrix factorization. Chen et al. [28] introduced Bayesian framework into tensor factorization, and proposed an augmented tensor factorization model whose parameters are automatically learned by variational Bayes. In [33], Chen et al. defined a truncated nuclear norm (TNN) on traffic tensors of location $\times$ day $\times$ time of day and introduced a universal rate parameter to control the degree of truncation on all tensor modes on the basis of low-rank tensor completion (LRTC) framework. To make model better capture the global consistency of traffic data, Chen et al. [34] also proposed a low-rank autoregressive tensor completion (LATC) framework by introducing temporal variation as a new regularization term into the completion of a third-order (sensor $\times$ time of day $\times$ day) tensor. Unlike the method of tensor factorization, some work focuses on using neural networks to impute missing data. In [17], Yoon et al. introduced the theory of Generative Adversarial Nets (GAN) [35], where a generator is utilized to generate the missing data from the incomplete data and a discriminator is used to measure the quality of the generated data. On the basis of GAN, Xu et al. [19] proposed GA-GAN, which adds GraphSAGE [36] as a feature extraction module to aggregate the temporal-spatial information from the neighbors of each road. In addition to GAN, attention mechanism is also used to impute missing data. In [20], Wu et al. designed Multi-Attention Tensor Completion Network (MATCN). It utilizes a spatial signal propagation module and a temporal self-attention module to execute representation aggregation and dynamic dependencies extraction at the spatiotemporal level. After imputing the missing data with the above methods, the conventional traffic prediction models [7], [10], [11] work on it and get the

TABLE I
COMPARISON OF TRAFFIC PREDICTION MODELS

Model Name	Model Type	Handles Data Missing	Has Spatial Property	Knowledge Transfer	Spatial Dependency	Temporal Dependency
DCRNN [10]	Deep Learning		✓		GCN	GRU
AGCRN [11]	Deep Learning		✓		Adaptive-GCN	GRU
GRU-D [21]	Deep Learning	✓			—	Masked-GRU
LSTM-GL-ReMF [22]	Tensor Decomposition	✓	✓		GCN Regularization	LSTM Regularization
TRTF [23]	Tensor Decomposition	✓	✓		Graph Laplacian Regularization	Temporal Graph Regularization
BTTF [28]	Tensor Decomposition	✓	✓		Gaussian Model	VAR
MTPF [29]	Deep Learning	✓	✓		Adaptive-GCN	GRU
GSTAE [30]	Deep Learning	✓	✓		Adaptive-GCN	GRU
FlashST [31]	Deep Learning		✓	✓	GCN based Message Passing	Gating Mechanism
3STL [32]	Deep Learning			✓	—	LSTM
FBDA [25]	Deep Learning			✓	—	LSTM
SEDA(ours)	**Deep Learning**	✓	✓	✓	Attention-GCN	Masked-GRU

predicted value. However, this indirect prediction strategy will bring cumulative errors.

In recent years, some work has begun to focus on ***the second strategy***: direct prediction based on incomplete data. Chen et al. [37] propose a Bayesian temporal factorization (BTF) framework, which can characterize both global and local consistencies by integrating low-rank matrix/tensor factorization and vector autoregressive (VAR) process into a single probabilistic graphical model. Yang et al. [22] designed a spatial and temporal regularized matrix factorization model (LSTM-GL-ReMF), which chooses Long Short-term Memory (LSTM) model as the temporal regularizer to capture temporal dependency and Graph Laplacian as the spatial regularizer to utilize spatial correlations among the network. Baggag et al. [23] proposed a tensor representation for the series of road network snapshots, and developed a regularized factorization method (TRTF), which incorporates spatial properties of the road network and utilizes graph-based temporal dependency. Che et al. [21] incorporated two representations of missing patterns, masking and time interval into a deep model architecture and proposed GRU-D, which can capture the long-term temporal dependencies in time series. Qu et al. [29] proposed a multi-task pretraining and fine-tuning (MTPF) approach to impute and predict traffic time series. It pretrains an autoencoder with different missing data patterns and missing rates, then the autoencoder performs fine-tuning for specific missing data scenarios. Similarly based on the autoencoder architecture, Wang et al. [30] proposed a graph-based spatiotemporal autoencoder (GSTAE). It regards the imputation and prediction as two parallel tasks and trains them sequentially. Then it utilizes graph convolutional layers with a self-adaptive adjacency matrix for spatial dependencies modeling and applies gated recurrent units for temporal learning. The above works achieve the goal of direct prediction on incomplete data and alleviate the cumulative error. However, they can only acquire knowledge from one single dataset. With the increase of missing rate, the remaining knowledge in the dataset will be greatly reduced, which may prevent the model from acquiring enough features to make accurate predictions.

B. Domain Adaptation

Domain Adaptation theory (DA) is a kind of transfer learning aiming to narrow the domain gap of different data sources by learning domain-invariant representations and unify the features of data with different distributions into the same feature space.

The main field of domain adaptation application is ***classification***. In [38], Ganin et al. introduced adversarial neural networks into domain adaptation and proposed domain-adversarial learning. It embeds domain adaptation into the process of learning representation, which makes the classification decisions are made based on features that are both discriminative and invariant to the change of domains. Long et al. [39] further present conditional adversarial domain adaptation, which can capture the cross-covariance between feature representations and classifier predictions, and control the uncertainty of classifier predictions to guarantee the transferability. Considering the problem of class alignment, Kang et al. [40] proposed a Contrastive Adaptation Network (CAN) to optimize a new metric which explicitly models the intra-class domain discrepancy and the inter-class domain discrepancy. It addresses the issue of misalignment in minimizing the domain discrepancy. According to different scenarios, some work has made corresponding improvements to domain adaptation. To better conduct sentiment classification, Xue et al. [41] devised a deep adversarial mutual learning approach involving two groups of feature extractors, domain discriminators, sentiment classifiers and label probers. To achieve better facial expression recognition (FER), Xie et al. [42] focused on the local face feature transfer and unified graph representation propagation with adversarial learning for cross-domain holistic-local feature co-adaptation. For domain adaptation of graph data, Shen et al. [43] proposed an adversarial cross-network deep network embedding (ACDNE) model, which integrates adversarial domain adaptation with deep network embedding to achieve graph node classification. In [44], Zhu et al. advocated the capturing of "essential graph information" as the goal of transferable GNN training and designed EGI (Ego-Graph Information maximization). They conduct an analysis of EGI transferability regarding the difference between the local graph Laplacians of source and target graphs.

In addition to classification tasks, there is also pioneering work devoted to tackling the domain adaptation ***regression*** (DAR) problem at the theoretical or algorithmic level. Mansour et al. and Cortes and Mohri [45], [46] conducted theoretical analyses of the DAR problem. They proposed new

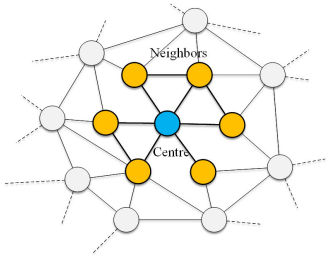


Fig. 2. Ego-graph. The blue node is the central node, the orange nodes are 1-hop neighbor nodes and the gray nodes are 2-hop neighbor nodes.

measurements of distribution distance and new settings of loss function. Chen et al. [47] found that regression performances are not robust to feature scaling and propose to match orthogonal bases to close domain shift without changing feature scale. Also, some work introduced domain adaptation into the traffic field. Li et al. [25], [32] designed a transfer learning framework based on LSTM, which transfers another road sequence features to the target road and addresses the data insufficiency at the single road level. Additionally, Li et al. [31] proposed the FlashST model for traffic prediction, which facilitates knowledge transfer between datasets via pre-training and fine-tuning. The model incorporates a lightweight spatio-temporal prompt network to capture invariant knowledge and uses a distribution mapping mechanism to align data distributions between the pre-training and downstream datasets. However, all of those works are applied to complete dataset. Knowledge transfer in missing data scenarios remains an open problem.

III. Definition and Problem Formulation

A. Definition 1: Graph

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ is utilized to represent the structure of a road network. Each traffic sensor deployed on the roads is treated as a node and $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ is a set of nodes, where $n = |\mathcal{V}|$ is the number of sensors. $\mathcal{E}$ is a set of edge and an edge $e_{ij} = (v_i, v_j) \in \mathcal{E}$ indicates the node $v_i \in \mathcal{V}$ links to the node $v_j \in \mathcal{V}$. The adjacency matrix $A \in \mathbb{R}^{n \times n}$ represents the connection of the nodes, if v_i is connected to v_j, $A_{ij} = 1$, otherwise $A_{ij} = 0$. $X_{i,t:\tau}$ represents the observation data from timestamp t to τ of node v_i.

B. Definition 2: k-Hop Ego-Graph

A k-hop ego-graph $\mathcal{S}_i$ consists of a central node v_i and a set of neighbor nodes $\mathcal{N}_i$, as shown in Figure 2. It has a k-layer centroid expansion such that the greatest distance between the central node v_i and any other nodes is k, i.e. $\forall v_j \in S_i, \left|d(v_i, v_j)\right| \leq k$, where $d(v_i, v_j)$ is the graph distance between node v_i and v_j. $\mathcal{V}_i$ and $\mathcal{E}_i$ represent the set of nodes and edges in ego-graph $\mathcal{S}_i$.

C. Definition 3: Source Domain and Target Domain

The incomplete dataset with prediction task is target domain, and the external dataset that provides alternative knowledge is source domain. The adjacency matrix A of two domains are complete, but the observation data in target domain is partially missing. We denote source domain and target domain by the letter "$\mathcal{S}$" and "$\mathcal{T}$" in the upper right of the symbols, e.g., $\mathcal{V}^{\mathcal{S}}$ indicates the set of nodes in source domain, $A^{\mathcal{T}}$ indicates the adjacency matrix in target domain, $\mathcal{S}_j^{\mathcal{T}}$ indicates an ego-graph in target domain et al.

D. Problem Formulation

Given an incomplete dataset as the target domain and an external dataset as the source domain, our purpose is to predict the future traffic state $X_{t+1}^{\mathcal{T}}$ of target domain based on the historical data $X_{t-w:t}^{\mathcal{T}}$ from target domain and the external knowledge $\mathcal{K}^{\mathcal{S}}$ from source domain. The problem is formulated as:

$$X_{t+1}^{\mathcal{T}} = \mathcal{F}_\theta(X_{t-w:t}^{\mathcal{T}}, \mathcal{K}^{\mathcal{S}} | \mathcal{G}^{\mathcal{T}}, \mathcal{G}^{\mathcal{S}}), \tag{1}$$

where $\mathcal{F}_\theta()$ is the spatiotemporal model that needs to be designed, θ is trainable parameters in the model, w is the length of time window. $\mathcal{G}$ is the graph information. We summarize the important notations involved in the paper as shown in Table VIII at the end of this paper.

IV. Methodology

In this section, we first introduce the overview of the SEDA framework which can match alternative data of missing features and make predictions based on ego-graph domain adaptation. Then, the design details of the key components in SEDA are illustrated, including ego-graphs cross-domain alignment, spatiotemporal feature extraction with missing mask, missing feature reconstruction by dual-branch cross recoupling, and loss function. Following the explanation of the framework, the selection method of the source domain is discussed.

A. Overview of the SEDA Framework

The architecture of SEDA is shown in Figure 3. The goal of SEDA is to leverage the alternative data from external sources to assist the target domain with missing data to achieve accurate traffic state prediction. The first step is to find the alternative data based on Ego-graphs Cross-Domain Alignment. The nodes with data missing in the target domain are treated as the centre to determine the ego-graphs. The ego-graphs in the source domain that have a similar topology to those in the target domain are matched to provide alternative data. Then the data of matched ego-graphs are compressed into Neighbor Feature and Central Feature based on the Spatiotemporal Feature Extraction with Missing Mask, where the Central Feature in the target domain is inaccurate or missing due to the missing data. Then, in Missing Feature Reconstruction by Dual-branch Cross reCoupling, the existing Neighbor Feature and Central Feature are decoupled and form two branches: Inter-domain branch and Intra-domain branch. The Inter-domain branch is used to narrow the domain gap, and the Inter-domain branch is used to learn the relationship between the Neighbor Feature and Central Feature. Finally, the inaccurate or missing Central Feature in the target domain is reconstructed by recoupling the two branches. The predicted value is obtained based on the reconstructed feature.

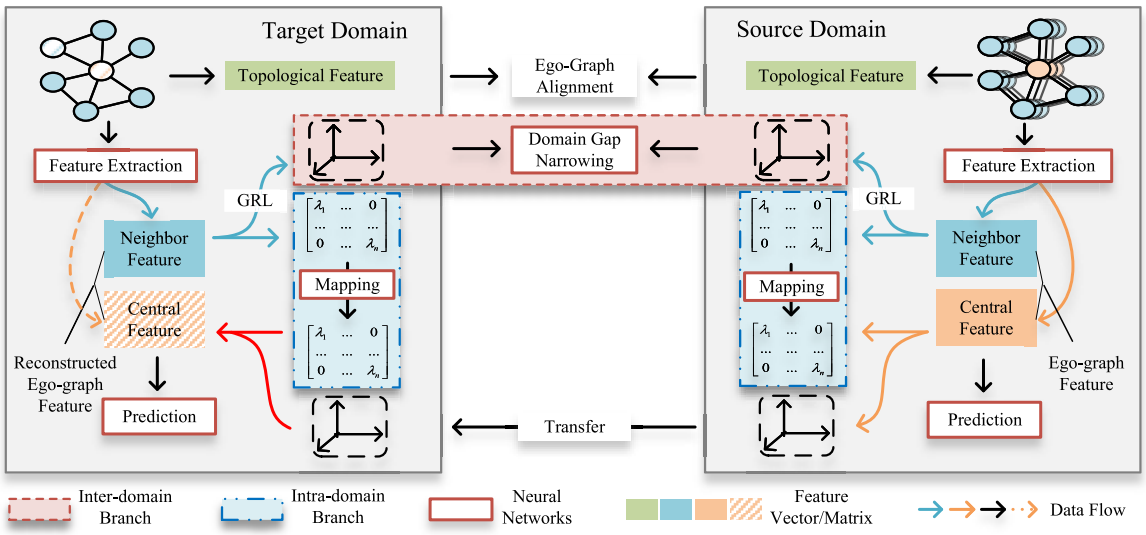


Fig. 3. Architecture of SEDA. Feature extraction neural network and prediction neural network are trained in source domain and applied in target domain.

B. Ego-Graphs Cross-Domain Alignment

In this section, we find the alternative data for missing data based on the topological similarity of ego-graphs. Specifically, we align cross-domain ego-graphs according to the local and global two-level multi-dimensional topological vectors.

At the local level, we adopt degree density $\eta_{i,d}$ and normalized structure entropy $\eta_{i,e}$ to measure the connectivity and equilibrium of the edge distribution [48] of ego-graph $\mathcal{S}_i$:

$$\eta_{i,d} = \frac{2\,|\mathcal{E}_i|}{|\mathcal{V}_i|\,|\mathcal{V}_i - 1|}, \tag{2}$$

$$\eta_{i,e} = -\frac{2\sum_{j=1}^{|\mathcal{V}_i|} I_j \ln I_j - \ln(4(|\mathcal{V}_i| - 1))}{2\ln|\mathcal{V}_i| - \ln(4(|\mathcal{V}_i| - 1))}, \tag{3}$$

where $-\sum_{j=1}^{|\mathcal{V}_i|} I_j \ln I_j$ represents the structure entropy of ego-graph, $I_j = \frac{k_j}{K}$ is the degree weight of each neighbor node, $K = \sum_{n=1}^{|\mathcal{V}_i|} k_n$, k_n is the degree of node $v_n \in \mathcal{V}_i$. $\mathcal{E}_i$ is the edge set of the ego-graph $\mathcal{S}_i$. These local topological metrics have a great correlation with the accessibility of transportation and the development of the area. They can potentially reflect traffic observation characteristics.

At the global level, we use closeness centrality and betweenness centrality to measure the importance of the central node v_i of an ego-graph [48]. Closeness centrality $\eta_{i,c}$ is the reciprocal of the average distances from node v_i to all other nodes $v_j \in \mathcal{V}$ in the whole graph. It can measure how close the node v_i is to the center of the whole graph $\mathcal{G}$. Betweenness centrality $\eta_{i,b}$ measures the importance of the central node v_i by the number of shortest paths through the node. It indicates whether the node is a critical articulation point in the whole graph,

$$\eta_{i,c} = \frac{|\mathcal{V}|}{\sum_{j=1}^{|\mathcal{V}|} d_{ij}}, \tag{4}$$

$$\eta_{i,b} = \frac{2}{N^2 - 3N + 2} \sum_{s \neq t \neq v \in \mathcal{V}} \frac{\sigma_{st}^{v_i}}{\sigma_{st}}, \tag{5}$$

where d_{ij} is the distance between node v_i and node v_j, σ_{st} is the number of shortest paths from node v_s to node v_t, $\sigma_{st}^{v_i}$ is the number of shortest paths from node v_s through node v_i to node v_t. These global topological metrics can help identify the location of central nodes of ego-graphs (near the urban areas, near the suburban area) and measure their importance (trunk road, branch road), which facilitates the search for supplementary knowledge. By combining the above four metrics, we can obtain the two-level multi-dimensional topological vector $\eta_i = [\eta_{i,d}, \eta_{i,e}, \eta_{i,c}, \eta_{i,b}]$ of ego-graph $\mathcal{S}_i$. It's important to note that the vector η_i can be extended flexibly by other observed metrics, such as normalized road class, number of lanes, if these are available.

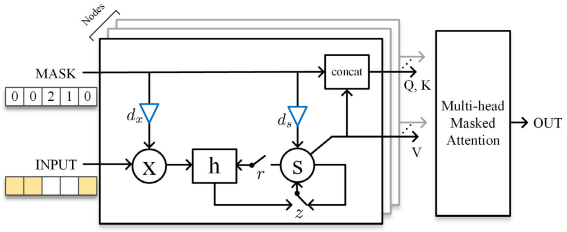


Fig. 4. The architecture of masked attention network. S is the hidden state, h is the current state, *concat* is the concatenation operation, r and z are gates.

Based on the topological vector η_i, each ego-graph in target domain aligns the m most similar ego-graphs in source domain for matching alternative data. Euclidean distance is used to calculate the similarity of topological vectors:

$$\underset{i \in \mathcal{V}^{\mathcal{S}}, j \in \mathcal{V}^{\mathcal{T}}}{sim(i,j)} = \sqrt{\sum\nolimits_{k \in \{d,e,c,b\}} \left(\eta_{i,k}^{\mathcal{S}} - \eta_{j,k}^{\mathcal{T}}\right)^2}, \tag{6}$$

KD-tree [49] is utilized to optimize time complexity of search.

C. Spatiotemporal Feature Extraction With Missing Mask

In this section, we compress the data of aligned ego-graphs into neighbor features and central features. Specifically, we design a masked attention network $Masked\_A(x)$ to extract the features. As shown in Figure 4, a missing mask M and two delay indexes d_x, d_s are added in the Gated Recurrent

Unit to sense data missing, and a Multi-head Masked Attention mechanism is designed to flexibly aggregate neighbor node features based on data missing status. It is simultaneously deployed in both the target and source domains to alleviate the impact of missing data '0' during the feature extraction process.

The missing mask $M_i \in \mathbb{R}^w$ of observation data $X_{i,t-w:t} \in \mathbb{R}^w$ is used to record the time intervals between each data and the last existing observation data. It enables the neural network to sense the status of data missing. If the input data $X_{i,t}$ is missing, we calculate a state delay index d_s and an input delay index d_x according to mask value $M_{i,t}$ to make up the missing data. Index $d_x = e^{-max(0,\sigma(M_{i,t}W_{dx}))}$ is used to weight the last observation data and the mean data of the observed time series, index $d_s = e^{-max(0,\sigma(M_{i,t}W_{ds}))}$ is used to attenuate s_{t-1}, where W_{dx} and W_{ds} are trainable parameters. Therefore, the masked GRU is formulated as:

$$X_r = d_x X_{lo} + (1 - d_x)\bar{X}, \tag{7}$$

$$r = \sigma((X_r|s_{t-1})W_r), \tag{8}$$

$$z = \sigma((X_r|s_{t-1})W_z), \tag{9}$$

$$h = \tanh((X_r|s_{t-1} \otimes r)W_h), \tag{10}$$

$$s_t = d_s z \otimes s_{t-1} + (1 - z) \otimes h, \tag{11}$$

X_{lo} is the last observation data, $\bar{X}$ is the mean value of input data. The output of the hidden state at the last timestamp is the node features $F_{o,i}$ of node i in the ego-graph $\mathcal{S}$.

Furthermore, we divide the node features of an ego-graph into central feature and neighbor feature. The neighbor feature is generated based on a multi-head masked self-attention mechanism, which adaptively aggregates useful features based on the missing status of neighbors. The neighbor feature F_n of ego-graph $\mathcal{S}$ is obtained by:

$$Q = (\bigcup_{j \in \mathcal{N}_i} F_{o,j} | \bigcup_{j \in \mathcal{N}_i} M_j)W_q, \tag{12}$$

$$K = (\bigcup_{j \in \mathcal{N}_i} F_{o,j} | \bigcup_{j \in \mathcal{N}_i} M_j)W_k, \tag{13}$$

$$V = (\bigcup_{j \in \mathcal{N}_i} F_{o,j})W_v, \tag{14}$$

$$head_m = softmax(\frac{\mathrm{QK^T}}{\sqrt{d_k}})V, \tag{15}$$

$$F_n = Concat(head_1, \ldots head_m)W_h, \tag{16}$$

where $\bigcup$ indicates the concatenation operation, K^T is transpose of K, d_k is the dimension of K. The feature of the central node is directly treated as the central feature F_c of the ego-graph, which does not need aggregation. At this point, we have obtained the central feature F_c and neighbor feature F_n of an ego-graph $\mathcal{S}$. The features in source domain $F_c^{\mathcal{S}}$, $F_n^{\mathcal{S}}$ and the neighbor feature in target domain $F_n^{\mathcal{T}}$ are reliable, while the central feature in target domain $F_c^{\mathcal{T}}$ is less reliable due to missing data.

D. Missing Feature Reconstruction by Dual-Branch Cross reCoupling

In this section, we narrow the gap between the two domains and reconstruct the missing feature based on the feature from the source domain. Specifically, we symmetrically decouple the features of aligned ego-graphs to form an intra-domain branch and an inter-domain branch, then cross recouple the two branches to reconstruct missing features with isolating external domain specificity. Adversarial domain adaptation is used to narrow the domain gap in the inter-domain branch, and a transferable relation network $R(x)$ is designed to learn the mapping of domain-specific factors in the intra-domain branch.

The reliable features $\{F_c^{\mathcal{S}}, F_n^{\mathcal{S}}, F_n^{\mathcal{T}}\}$ are first decoupled into orthogonal bases and singular values by Singular Value Decomposition. The singular value λ_i represents the weight of the feature on orthogonal bases of each dimension. $\sum_{i=0}^{n} \lambda_i = \|F_i\|_F^2$ indicates the feature scales. So the singular values can be treated as the domain-specific factors in the intra-domain branch and orthogonal bases are treated as domain-invariant factors in the inter-domain branch. The decomposition is formulated as:

$$\begin{pmatrix} f_{11} & \cdots & f_{1b} \\ \vdots & \ddots & \vdots \\ f_{m1} & \cdots & f_{mb} \end{pmatrix}_i = U_i \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{pmatrix}_i V_i^T \tag{17}$$

where $\left[f_{1j}, \ldots, f_{mj}\right]^T$ is the central feature or neighbor feature of source domain or target domain, U_i is the orthogonal bases, λ is the singular value, b is the size of minibatch.

In the inter-domain branch, we construct an adversarial neural network to narrow the domain gaps of orthogonal bases. Specifically, a discrimination network $D(x)$ is constructed to distinguish whether the orthogonal bases come from the source domain or the target domain. A Gradient Reversal Layer (GRL) is added between the discrimination network $D(x)$ and masked attention network $Masked\_A(x)$, which makes the gradient change to the opposite direction. The gradient reverse is formulated as,

$$GRL(\frac{\partial L}{\partial \theta}) = -\lambda \frac{\partial L}{\partial \theta}, \tag{18}$$

$$\lambda = \frac{2}{1 + \exp(-\gamma\rho)} - 1 \tag{19}$$

where γ is the progression of iterations in training, ρ is the weight parameter. Hence, the network $Masked\_A(x)$ updates its parameters in the opposite direction of network $D(x)$, and forms an adversarial relationship. The outputs of $Masked\_A(x)$ try to "confuse" $D(x)$, and make it unable to make accurate discrimination, thus the purpose of narrowing the domain gap is achieved.

In the intra-domain branch, due to the traffic diffusion phenomena in ego-graphs, the singular values of the central feature are different from those of the neighbor feature. We construct a transferable fully connected relation network $R(x)$ to learn the mapping between the singular values of neighbor feature and central feature in source domain. Then $R(x)$ is deployed in target domain, where it can generate the missing central singular values based on the neighbor singular values. After that, we cross recouple the orthogonal bases of source domain and the singular values generated in target domain to reconstruct the missing central feature:

$$S_{c\_r,j}^{\mathcal{T}} = R(S_{n,j}^{\mathcal{T}}), \tag{20}$$

$$U^{\mathcal{T}}_{c\_r,j} = U^{\mathcal{S}}_{c,i}, \tag{21}$$

$$V^{\mathcal{T}}_{c\_r,j} = V^{\mathcal{S}}_{c,i}, \tag{22}$$

$$F^{\mathcal{T}}_{c\_r,j} = U^{\mathcal{T}}_{c\_r,j} S^{\mathcal{T}}_{c\_r,j} V^{\mathcal{T}}_{c\_r,j}, \tag{23}$$

where $S^{\mathcal{T}}_{n,j}$ is the neighbor singular value matrix of ego-graph $\mathcal{S}^{\mathcal{T}}_j$ in target domain, $U^{\mathcal{S}}_{c,i}$ is the central orthogonal bases of the aligned ego-graph $\mathcal{S}^{\mathcal{S}}_i$ in source domain. $S^{\mathcal{T}}_{c\_r,j}$, $U^{\mathcal{T}}_{c\_r,j}$, $V^{\mathcal{T}}_{c\_r,j}$, $F^{\mathcal{T}}_{c\_r,j}$ are the reconstructed singular value, left singular vector, right singular vector and central feature.

Then the reconstructed feature and the original feature are weighted according to the data missing rate r of ego-graph $\mathcal{S}^{\mathcal{T}}_j$ to obtain the final feature $F^{\mathcal{T}}_{c\_f,j} = rF^{\mathcal{T}}_{c\_r,j} + (1-r)F^{\mathcal{T}}_{c,j}$. When the missing rate is low, more original feature $F^{\mathcal{T}}_{c,j}$ will be retained, otherwise, more reconstructed feature $F^{\mathcal{T}}_{c\_r,j}$ will be fused. It helps preserve the original features and mitigate impact from potential missing data in the source domain. Finally, after combining the central feature $F^{\mathcal{T}}_{c\_f,j}$ and neighbor feature $F^{\mathcal{T}}_{n,j}$ in target domain, the predicted value $\hat{p}_j$ of node v_j can be obtained based on the fully connected prediction network $P(x)$ trained in source domain. The overall algorithm flow is shown in Algorithm 1.

E. Loss Function

The loss function of SEDA consists of three parts: loss $\mathcal{L}_p$ of prediction network $P(x)$, loss $\mathcal{L}_d$ of discrimination network $D(x)$ and loss $\mathcal{L}_r$ of relation network $R(x)$. In prediction network $P(x)$ and relation network $R(x)$, the Mean Square Error (MSE) is adopted as the loss function:

$$\mathcal{L}_p = \frac{1}{\left|\mathcal{V}^{\mathcal{S}}\right|} \sum\nolimits_{i=1}^{|\mathcal{V}^{\mathcal{S}}|} (p_i - \hat{p}_i)^2, \tag{24}$$

$$\mathcal{L}_r = \frac{1}{\left|\mathcal{V}^{\mathcal{S}}\right|} \sum\nolimits_{i=1}^{|\mathcal{V}^{\mathcal{S}}|} (R(S^{\mathcal{S}}_{n,i}) - S^{\mathcal{S}}_{c,i})^2, \tag{25}$$

where $\hat{p}_i$ is the predicted value and p_i is the ground truth. It should be noted that if the ego-graphs in target domain have no data missing, their features will also be involved in the training process. We use cross entropy as the loss function of discrimination network $D(x)$:

$$\mathcal{L}_d = \sum\nolimits_{i=1}^{|\mathcal{V}^{\mathcal{S}}|+|\mathcal{V}^{\mathcal{T}}|} -y_i \log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i), \tag{26}$$

y_i is the result of domain discrimination, and $\hat{y}_i$ is the domain category label. The final loss function is:

$$\mathcal{L} = \alpha\mathcal{L}_p + (1-\alpha)\mathcal{L}_r - \lambda\mathcal{L}_d + \beta \left\|\theta\right\|, \tag{27}$$

where α and β are weight parameters used to trade-off the learning rate, λ is the weight parameter dynamically updated with iteration, $\|\theta\|$ is the L2 regularization term used to prevent overfitting. The update process of parameters $\theta_f, \theta_r, \theta_p, \theta_d$ in masked attention network, relation network, prediction network and discrimination network is:

$$\theta_f \leftarrow \theta_f - l(\alpha\frac{\partial L_p}{\partial \theta_p} + (1-\alpha)\frac{\partial L_r}{\partial \theta_r} - \lambda\frac{\partial L_d}{\partial \theta_d}), \tag{28}$$

$$\theta_r \leftarrow \theta_r - l(1-\alpha)(\frac{\partial L_r}{\partial \theta_r}), \tag{29}$$

$$\theta_p \leftarrow \theta_p - l(\alpha\frac{\partial L_p}{\partial \theta_p}), \tag{30}$$

$$\theta_d \leftarrow \theta_d - l(\frac{\partial L_d}{\partial \theta_d}), \tag{31}$$

We use "Adam" optimizer for optimization.

Algorithm 1 SEDA Algorithm

Input: $\mathcal{G}^{\mathcal{S}}$,$\mathcal{G}^{\mathcal{T}}$, $X^{\mathcal{S}}$, $X^{\mathcal{T}}$, Ground Truth $p^{\mathcal{S}}$ (only for training)
Output: $\hat{p}^{\mathcal{T}}$
$F^{\mathcal{S}}_c, F^{\mathcal{S}}_n = Masked\_A(X^{\mathcal{S}})$
$F^{\mathcal{T}}_c, F^{\mathcal{T}}_n = Masked\_A(X^{\mathcal{T}})$
for each $v^{\mathcal{S}}_i, v^{\mathcal{T}}_j \in \mathcal{V}^{\mathcal{S}}, \mathcal{V}^{\mathcal{T}}$ **do**
    $\eta^{\mathcal{S}}_i, \eta^{\mathcal{T}}_j = [\eta^{\mathcal{S}}_{i,d}, \eta^{\mathcal{S}}_{i,e}, \eta^{\mathcal{S}}_{i,c}, \eta^{\mathcal{S}}_{i,b}], [\eta^{\mathcal{T}}_{j,d}, \eta^{\mathcal{T}}_{j,e}, \eta^{\mathcal{T}}_{j,c}, \eta^{\mathcal{T}}_{j,b}]$
end for
for each $\eta^{\mathcal{T}}_j$ **do**
    Search m $\eta^{\mathcal{S}}_i$ that are most similar to $\eta^{\mathcal{T}}_j$
    Get aligned ego-graphs $S^{\mathcal{T}}_j \leftrightarrow \{S^{\mathcal{S}}_{i1}, \ldots, S^{\mathcal{S}}_{im}\}$
    $U^{\mathcal{T}}_{c,j}, S^{\mathcal{T}}_{c,j}, V^{\mathcal{T}}_{c,j} = SVD(F^{\mathcal{T}}_{c,j})$
    $U^{\mathcal{T}}_{n,j}, S^{\mathcal{T}}_{n,j}, V^{\mathcal{T}}_{n,j} = SVD(F^{\mathcal{T}}_{n,j})$
    for each $F^{\mathcal{S}}_{c,i}, F^{\mathcal{S}}_{n,i}$ in $\{S^{\mathcal{S}}_{i1}, \ldots, S^{\mathcal{S}}_{im}\}$ **do**
        $U^{\mathcal{S}}_{c,i}, S^{\mathcal{S}}_{c,i}, V^{\mathcal{S}}_{c,i} = SVD(F^{\mathcal{S}}_{c,i})$
        $U^{\mathcal{S}}_{n,i}, S^{\mathcal{S}}_{n,i}, V^{\mathcal{S}}_{n,i} = SVD(F^{\mathcal{S}}_{n,i})$
        $\hat{p}^{\mathcal{S}}_i, y^{\mathcal{S}}_i = P(F^{\mathcal{S}}_{c,i}, F^{\mathcal{S}}_{n,i}), D(U^{\mathcal{S}}_{n,i})$
        $S^{\mathcal{S}}_{c\_r,i} = R(S^{\mathcal{S}}_{n,i})$
    end for
    $F^{\mathcal{T}}_{c\_r,j} = mean(U^{\mathcal{S}}_{c,i} S^{\mathcal{T}}_{c\_r,j} V^{\mathcal{S}}_{c,i}), i \in \{i1, i2 \ldots, im\}$
    $F^{\mathcal{T}}_{c\_f,j} = rF^{\mathcal{T}}_{c\_r,j} + (1-r)F^{\mathcal{T}}_{c,j}$
    $\hat{p}^{\mathcal{T}}, y^{\mathcal{T}} = P(F^{\mathcal{T}}_{c\_f,j}, F^{\mathcal{T}}_{n,j}), D(U^{\mathcal{T}}_{n,j})$
end for
$\mathcal{L}_d = CrossEntropy(y^{\mathcal{S}}, 0) + CrossEntropy(y^{\mathcal{T}}, 1)$
$\mathcal{L}_r = \frac{1}{|\mathcal{V}^{\mathcal{S}}|} \sum_{i=1}^{|\mathcal{V}^{\mathcal{S}}|} (R(S^{\mathcal{S}}_{n,i}) - S^{\mathcal{S}}_{c,i})^2$
$\mathcal{L}_p = \frac{1}{|\mathcal{V}^{\mathcal{S}}|} \sum_{i=1}^{|\mathcal{V}^{\mathcal{S}}|} (p^{\mathcal{S}}_i - \hat{p}^{\mathcal{S}}_i)^2$
$\mathcal{L} = \alpha\mathcal{L}_p + (1-\alpha)\mathcal{L}_r - \lambda\mathcal{L}_d + \beta \left\|\theta\right\|$
Back propagation and update model parameters

F. Dataset Similarity Measure

In this section, we present a simple and effective Dataset Similarity measure method based on KL divergence (DSKL) to guide the selection of source domain datasets. Three problems hinder us from calculating the dataset similarity: missing data in target domain, differences in sampling frequency, and mismatch of dataset timestamps. To solve these problems, we calculate the daily average of a dataset to form a typical sequence. The dataset typical sequence $D_i \in \mathbb{R}^{1\times f}$ of the $i\text{-}th$ dataset is formulated as,

$$D_i = \{mean(\sum\nolimits_{u=0}^{n} \sum\nolimits_{t\in\{t \bmod f=0\}} X_{iu,t}), \ldots, mean(\sum\nolimits_{u=0}^{n} \sum\nolimits_{t\in\{t \bmod f=f-1\}} X_{iu,t})\} \in \mathbb{R}^{1\times f}, \tag{32}$$

where $X_{iu,t}$ is the observation data of the $u\text{-}th$ node in the $i\text{-}th$ dataset at timestamp t, mean() is taking the mean of non-zero value, f is the daily sampling frequency of the dataset. We resample source domain to match its frequency to target domain, then move source domain to make the lowest points of the two domains coincide to achieve timestamp alignment.

TABLE II
DATASETS DETAILS

Dataset	# Samples	# Nodes	Sample Rate
PeMS_04	16,992	307	5 minutes
PeMS_08	17,856	170	5 minutes
CD_S	52,560	826	10 minutes
SZ_S	52,560	1,174	10 minutes

To obtain dataset similarity, we first use KL divergence to measure the data distribution similarity between each node pair across two domains. Specifically, we first fill in the missing value of the node typical sequence D_{iu} with the dataset typical sequence D_i, then calculate the top k similarity $top\_k(SimN)$ between node pairs across target and source domains. The mean of the top k similarity is treated as the dataset similarity. It is formulated as,

$$D_{iu} = \{mean(\sum\nolimits_{t\in\{t \mod f=0\}} X_{iu,t}), \ldots, mean(\sum\nolimits_{t\in\{t \mod f=f-1\}} X_{iu,t})\}, \tag{33}$$

$$SimN(D_{iu}, D_{jv}) = \exp(-\sum D_{iu}(x) \log \frac{D_{iu}(x)}{D_{jv}(x)}), \tag{34}$$

$$SimD(D_i, D_j) = \underset{u\in\mathcal{G}_i}{mean}(\underset{v\in\mathcal{G}_j}{top\_k}(SimN(D_{iu}, D_{jv}))), \tag{35}$$

where D_{iu} is the typical sequence of the $u\text{-}th$ node in $i\text{-}th$ dataset, $SimN()$ is the similarity between two node typical sequences, $SimD()$ is the similarity between two datasets, $top\_k()$ is the operation that selects the first k maximum value.

V. EXPERIMENT

A. Datasets and Baselines

We evaluate our proposed SEDA on real public traffic datasets[1] from the US and China. The dataset details are shown in Table II. PeMS_04 and PeMS_08 are the freeway flow/speed/occupancy datasets collected in California, the US. CD_S and SZ_S are the urban traffic speed datasets collected in Chengdu and Shenzhen, China. We choose seven classic and state-of-the-art traffic prediction models as baselines,[2] which include two traditional prediction models, two knowledge transfer-based models, and three incomplete data prediction models. In addition, we also select a state-of-the-art missing data imputation model to form an "Imputation + Prediction" model. Traditional traffic prediction models:

1) DCRNN [10]: It predicts traffic flow by capturing spatiotemporal dependencies using random walks on the graph and autoencoder architectures.
2) AGCRN [11]: It constructs graph structure based on observation data adaptively, and initializes independent convolution parameters for each node.

Knowledge transfer-based traffic prediction models:

3) 3STL [32]: It is a 3-stage traffic prediction framework with transfer learning and predicts traffic flow based on a long short-term memory network on complete datasets.
4) FlashST [31]: It facilitates knowledge transfer via pre-training and fine-tuning, and incorporates a spatio-temporal prompt network to capture invariant knowledge and uses a distribution mapping mechanism to align data distributions.

Incomplete data traffic prediction models:

5) BTTF [28]: It is a Bayesian temporal factorization (BTF) framework for modeling multidimensional time series in the presence of missing values. It integrates low-rank matrix/tensor factorization and vector autoregressive (VAR) process into a single probabilistic graphical model.
6) LSTM-GL-ReMF [22]: It is a regularized matrix factorization model, which chooses Long Short-term Memory model and Graph Laplacian as the temporal and spatial regularizers to enhance prediction performance.
7) GSTAE [30]: It treats imputation and prediction as two parallel tasks and utilizes graph convolutional layers with a self-adaptive adjacency matrix for spatial dependencies modeling and gated recurrent units for temporal learning.

Traffic missing data imputation model:

8) BATF [28]: It is an augmented tensor factorization model that adopted a fully Bayesian framework for automatically learning parameters using variational Bayes.

We consider two missing types: point-wise missing (PM) and continuous missing (CM), following the experiment setting of LSTM-GL-ReMF [22] as shown in Figure 5. In PM, we randomly select $r\%$ data in the original dataset and assign their value as "0". In CM, we treat the data of every two consecutive days of one node as a data unit, then randomly select $r\%$ data units and assign their value as "0".

We test the performance of the models at missing rates 0%-40% in three modes: 1) *Direct Traditional Prediction*, we directly use the incomplete dataset to train and test the traditional traffic prediction model. 2) *Imputation + Prediction*, we use the state-of-the-art missing data imputation model to fill in the missing data, then input it into the traditional traffic prediction model. 3) *Direct Incomplete Data Prediction*, we directly input the missing traffic data into the incomplete prediction model for prediction. Our proposed SEDA is tested by combining the datasets as "source domain→target domain". It is important to note that we do not pre-train other baseline models except FlashST on the source datasets, as those baseline models cannot be deployed on datasets with different graph structures.

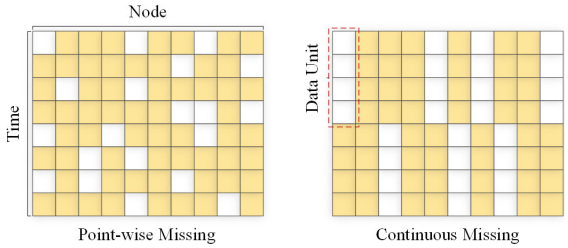


Fig. 5. Point-wise Missing (PM) and Continuous Missing (CM). Yellow squares represents observed data, and white squares represents missing data.

[1]Available at http://pems.dot.ca.gov/ and https://outreach.didichuxing.com/
[2]The codes are available at https://github.com/ChenXu02/traffic_SEDA

TABLE III
MODEL PERFORMANCE COMPARISON ON PEMS_08(FLOW) DATASETS (RMSE/MAE/MAPE)

Missing Rate	DCRNN	AGCRN	3STL	FlashST	LSTM-GL-ReMF	BTTF	GSTAE	**SEDA(ours)**
original	26.16/17.19/9.73	25.48/16.09/8.89	29.00/19.12/10.67	**22.87/15.25/8.35**	31.53/22.34/12.40	33.01/21.95/11.75	26.15/17.32/9.49	26.12/16.72/9.16
10%, PM	47.39/27.86/15.09	26.85/17.08/9.57	36.33/23.95/13.36	26.56/17.71/9.70	33.12/23.85/12.70	33.48/22.23/12.05	26.95/18.10/9.91	**20.11/16.86/9.37**
20%, PM	57.92/38.31/21.31	29.40/19.56/10.67	43.23/28.33/15.92	30.09/20.06/10.99	33.92/24.29/12.98	33.99/22.43/12.07	28.93/19.41/10.63	**26.63/18.24/9.67**
30%, PM	79.06/53.60/28.27	33.73/22.42/12.23	52.11/34.55/19.17	32.94/21.96/12.03	35.53/24.84/13.36	34.08/22.32/12.19	31.02/20.82/11.41	**31.06/19.37/10.22**
40%, PM	109.41/61.64/34.03	38.12/25.94/15.03	66.76/43.97/24.57	38.91/25.94/14.21	37.68/25.21/13.50	35.34/23.34/12.49	32.61/21.95/12.03	**31.61/20.56/10.85**
10%, CM	51.01/33.19/18.22	61.10/33.16/18.07	39.03/25.77/14.36	27.95/18.64/10.21	33.48/23.86/13.56	34.01/22.08/12.05	28.55/19.16/10.50	**24.19/17.03/9.73**
20%, CM	77.21/50.28/27.05	76.75/47.52/26.01	45.72/30.11/16.83	33.32/22.21/12.17	41.63/25.86/14.78	34.05/22.83/12.46	30.68/20.61/11.29	**26.35/19.18/10.07**
30%, CM	92.03/60.40/32.12	92.74/61.34/33.45	54.66/36.01/20.11	38.63/25.75/14.11	47.65/30.34/16.42	40.43/23.93/13.09	32.81/22.14/12.15	**31.55/20.52/10.99**
40%, CM	121.93/82.44/45.01	136.25/83.99/46.14	72.22/47.61/26.57	45.20/30.13/16.51	58.42/34.47/18.55	37.97/25.39/13.41	35.26/23.67/12.97	**35.19/21.93/11.71**

In training, we randomly select 70% of the real traffic dataset as the training set, 15% as the validation set (if required by the model), and 15% as the testing set. The length of the input sequence is 12. We determine the number of network layers and dimensions for all the models under the best performance in the complete data state (refer to their papers). All models are trained for 100 epochs in four 12th Gen Intel(R) Core(TM) i9-12900K CPUs and eight GeForce RTX 3090 GPUs.

In the model setting of SEDA, the time window $w = 12$, the batch size is 64, the range of the ego-graph is 2 hop, the data in source domain is resampled according to the sampling frequency of target domain, each ego-graph in target domain matches 3 ego-graphs in source domain, the initial learning rate $l = 0.002$ and decay by 20% at 10th, 20th, 30th, 60th, 90th epoch. The number of Gated Recurrent Unit layers in $Masked\_A(x)$ is 1, the dimension of hidden state is 64, the dimensions of each layer in predicting network $P(x)$ are 128, 256, 128, the dimensions of each layer in discrimination network $D(x)$ and relation network $R(x)$ are both 128, 128. Hyperparameters $\alpha = 0.6$ and $\beta = 0.7$ are chosen by grid search. We choose three general evaluation indicators: Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percent Error (MAPE):

$$RMSE = \sqrt{\frac{1}{|\mathcal{V}^{\mathcal{T}}|}\sum_{i\in\mathcal{V}^{\mathcal{T}}}(p_i - \hat{p}_i)^2}, \tag{36}$$

$$MAE = \frac{1}{|\mathcal{V}^{\mathcal{T}}|}\sum_{i\in\mathcal{V}^{\mathcal{T}}}|p_i - \hat{p}_i|, \tag{37}$$

$$MAPE = \frac{1}{|\mathcal{V}^{\mathcal{T}}|}\sum_{i\in\mathcal{V}^{\mathcal{T}}}\left|\frac{p_i - \hat{p}_i}{p_i}\right|, \tag{38}$$

where $\hat{p}_i$ is the predicted value and p_i is the ground truth.

B. Experiment Results

Firstly, we test the traditional traffic prediction models in mode "Direct Traditional Prediction" and the incomplete data traffic prediction models in mode "Direct Incomplete Data Prediction". We test our model with "PeMS_04(f)→PeMS_08(f)" and "SZ_S→CD_S". Results in Tables III and IV demonstrate our proposed SEDA has high prediction accuracy and stability under different data missing scenarios. In the original complete data, SEDA's MAPEs are 9.16% and 9.89% in PeMS_08 and CD_S, which is comparable to the performance of the baselines. In data missing 10%-40% scenarios, SEDA shows more obvious advantages. The mean MAPE of SEDA in different missing rates is lower than the best baseline (the underlined) 0.88%, 1.03%, 0.75%, 0.79% in scenarios (PeMS_08, PM), (PeMS_08, CM), (CD_S, PM), and (CD_S, CM). SEDA achieved slightly worse performance on CM than on PM, but still better than other baseline models under the same scenarios. We visualize SEDA's prediction results in CM 20% scenarios in CD_S and PeMS_08 in Figure 6, which shows SEDA can correctly fit the pattern of data, even if data are missing for continuous days (yellow part).

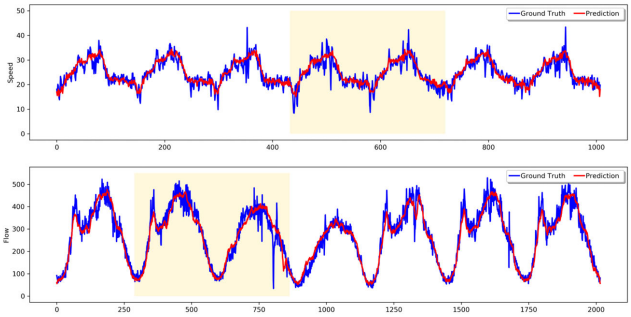


Fig. 6. Visualization of model performance in CM 20% scenarios. The upper is the performance in CD_S, the lower is the performance in PeMS_08(f).

Furthermore, we use the state-of-the-art missing data imputation model BATF to impute the missing traffic data, and input the imputed data into traditional prediction models to form the "Imputation + Prediction" test mode. The average imputation MAPE of BATF is 10.11%, and the performance of the traditional prediction model after imputing is shown in Table V. The results show SEDA achieves better performance with less computation, and its mean MAPE in different missing rates is lower than the best baseline (the underlined) more than 0.56%, 0.18%, 0.59%, and 0.45% in scenario (PeMS_08, PM), (PeMS_08, CM), (CD_S, PM), and (CD_S, CM). The cumulative error of imputation prevents traditional models from achieving their best performance. In few scenarios it even has an adverse effect compared to no imputation. Our SEDA effectively avoids cumulative error.

We also test the model's performance in extreme missing scenarios (50%-90%). We set eight different scenarios according to the dataset (PeMS_08, CD_S), missing type (PM, CM)

TABLE IV
MODEL PERFORMANCE COMPARISON ON CD_S DATASETS (RMSE/MAE/MAPE)

Missing Rate	DCRNN	AGCRN	3STL	FlashST	LSTM-GL-ReMF	BTTF	GSTAE	**SEDA(ours)**
original	3.87/2.41/9.91	3.77/2.39/9.57	4.42/2.97/12.32	**3.31/2.20/8.92**	4.33/2.94/11.88	4.19/2.54/10.83	3.55/2.35/9.53	3.90/2.44/9.89
10%, PM	4.16/2.77/11.32	3.91/2.51/10.19	4.56/3.06/12.71	3.86/2.57/10.43	4.66/3.04/12.19	4.21/2.78/11.10	3.87/2.52/10.20	**3.89/2.45/9.91**
20%, PM	4.63/3.08/12.53	4.12/2.85/11.29	5.30/3.56/14.75	4.31/2.88/11.67	4.74/3.07/12.42	4.36/2.90/11.23	3.93/2.64/10.69	**3.93/2.47/9.99**
30%, PM	5.19/3.32/13.35	4.95/3.01/12.16	6.70/4.50/18.71	4.73/3.15/12.79	4.85/3.11/12.50	4.36/2.89/11.42	4.09/2.71/10.98	**4.07/2.52/10.11**
40%, PM	5.37/3.74/15.29	5.59/3.65/14.09	8.89/5.95/24.73	5.06/3.38/13.72	4.89/3.13/12.59	4.51/3.01/11.57	4.39/2.85/11.29	**4.10/2.54/10.13**
10%, CM	5.09/3.21/12.42	10.32/4.52/18.32	4.56/3.06/12.69	3.95/2.64/10.69	4.67/3.05/12.41	4.28/2.83/11.14	3.84/2.53/10.25	**3.90/2.45/9.93**
20%, CM	5.93/3.86/15.37	12.56/6.31/25.56	5.73/3.83/15.95	4.51/3.01/12.21	4.85/3.12/12.59	4.27/2.87/11.21	4.10/2.66/10.83	**3.95/2.50/10.02**
30%, CM	9.97/5.86/21.73	15.12/8.50/33.16	7.08/4.74/19.71	5.14/3.43/13.92	4.91/3.18/12.99	4.36/2.90/11.47	4.18/2.77/11.14	**3.99/2.52/10.11**
40%, CM	11.11/6.17/23.11	21.96/11.25/45.15	9.61/6.44/26.72	5.77/3.86/15.63	5.06/3.25/13.33	4.33/2.89/11.29	4.29/2.82/11.43	**4.12/2.59/10.29**

TABLE V
EXPERIMENTS IN "IMPUTATION + PREDICTION" MODE ON PEMS_08(FLOW), CD_S DATASETS (RMSE/MAE/MAPE)

Dataset	PeMS_08(f)				CD_S			
Missing Rate	DCRNN	AGCRN	3STL	FlashST	DCRNN	AGCRN	3STL	FlashST
original	26.16/17.19/9.73	25.48/16.09/8.89	29.00/19.12/10.67	22.80/15.19/8.35	3.87/2.41/9.91	3.77/2.39/9.57	4.42/2.97/12.32	3.27/2.20/8.92
10%, PM	30.13/20.29/10.93	28.12/18.81/9.94	30.01/19.77/11.04	25.50/17.00/9.34	3.73/2.58/10.82	3.96/2.65/10.93	4.49/3.02/12.36	3.59/2.41/9.76
20%, PM	32.07/21.71/11.79	29.88/19.69/10.61	31.64/20.81/11.66	27.87/18.58/10.21	4.29/2.82/11.86	4.29/2.91/12.15	4.70/3.16/12.94	3.82/2.56/10.39
30%, PM	34.06/22.65/12.62	32.57/21.85/11.73	32.75/21.55/12.05	30.22/20.15/11.07	4.41/2.91/12.34	4.45/3.06/12.71	4.83/3.25/13.31	4.04/2.71/11.01
40%, PM	35.49/23.52/12.98	36.56/24.54/13.28	32.97/21.70/12.13	32.17/21.31/11.73	4.69/3.13/12.98	4.55/3.05/12.75	4.96/3.32/13.62	4.16/2.79/11.32
10%, CM	30.47/20.36/11.28	28.27/19.09/10.51	32.01/21.11/11.79	25.33/16.89/9.28	4.01/2.65/11.17	4.09/2.67/11.25	4.71/3.15/12.96	3.57/2.39/9.71
20%, CM	33.09/21.85/12.12	33.81/22.64/12.62	34.42/22.71/12.66	28.20/18.80/10.33	4.29/2.87/12.00	4.41/2.92/12.36	4.81/3.24/13.24	3.77/2.53/10.27
30%, CM	35.37/23.85/13.24	38.33/25.13/14.11	37.91/25.01/13.95	30.73/20.49/11.26	4.55/3.09/12.80	4.65/3.06/12.89	4.98/3.35/13.71	4.02/2.70/10.93
40%, CM	38.15/25.29/13.96	38.36/25.21/13.98	40.85/26.95/15.03	33.66/22.44/12.35	4.79/3.23/13.38	4.72/3.12/13.15	5.12/3.45/14.12	4.13/2.77/11.25

and data state (Missing, Imputed). The imputed data is only applied to the traditional models. Figure 7 shows the MAPE of each model at the missing rate of 0%-90% in each scenario. We can see that SEDA achieves significant advantages in almost all scenarios. Especially on high missing rate and the large dataset (e.g. CD_S), SEDA can maintain stable and high accuracy. We believe that the large urban road network has more obvious topological diversity, so that SEDA can accurately and easily find similar knowledge in source domain according to the topological information of ego-graph.

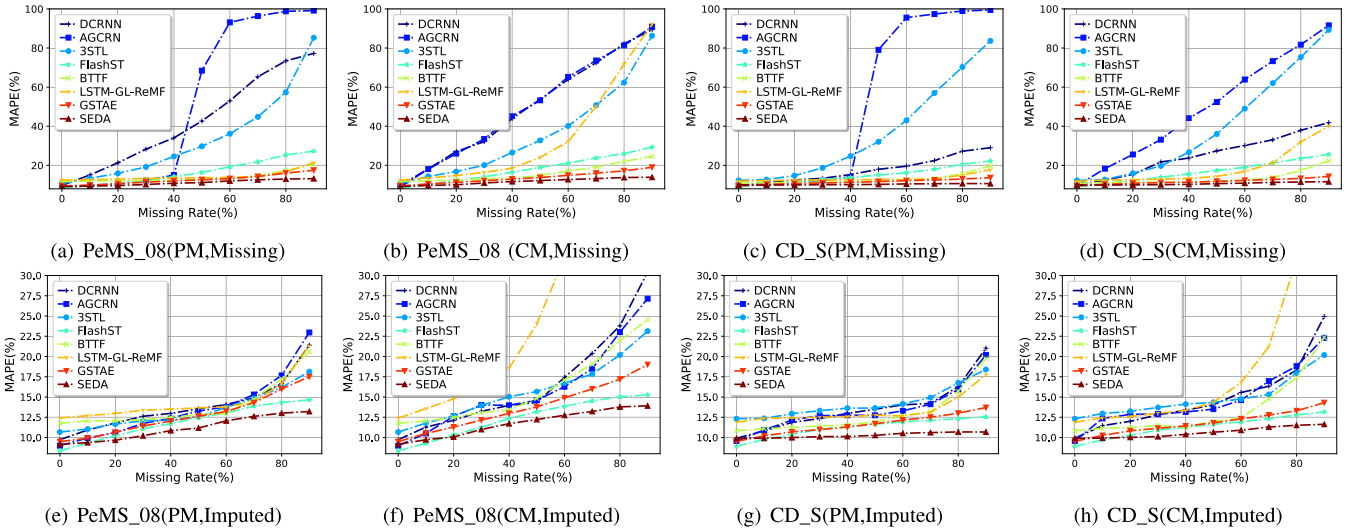
C. Ablation Experiments

To verify the validity of each component of the model, we perform ablation experiments on China urban speed dataset (CD_S). We test the effects of 1) the Domain Adaptation framework (DA), 2) Subgraph Alignment (SA), 3) Feature Decoupling (FD) and 4) Relation Network (RN) and construct four variant models:

- SEDA_DA. We remove the domain adaptation framework, including ego-graphs alignment and missing feature reconstruction. It only uses masked attention network $Masked\_A(x)$ and prediction network $P(x)$ to make predictions on target dataset. This is the basic part of SEDA, which is used to find out whether SEDA really can draw on knowledge from other sources.
- SEDA_SA. We remove the ego-graphs alignment, and make each ego-graph in target domain randomly match 3 ego-graphs in source domain. This variant model is used to verify whether ego-graphs with similar topology have similar features, to better supplement the missing feature.
- SEDA_FD. We remove SVD decomposition and directly feed the undecomposed features into the discrimination network $D(x)$ to narrow domain gap. It is used to verify whether decoupling the orthogonal bases and singular values of features helps the model to process data of different scales. We test the variant model's prediction performance in CD_S with two different scale datasets SZ_S and PeMS_04(s) as source domain.
- SEDA_RN. We remove the relation network $R(x)$ and directly take the neighbor singular values as the central singular values. It is used to test whether $R(x)$ can learn a stable mapping relationship between the singular values of central node and the neighbor nodes in ego-graphs.

We test the above variant models under scenarios: original, PM 20%, 40% and CM 20%, 40% in SZ_S→CD_S.

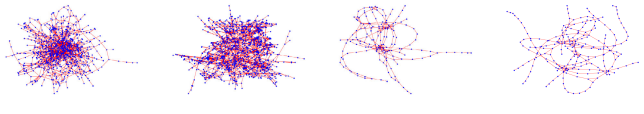
As shown in Table VI, all the variant models can maintain good performance in the original scenario. In SEDA_DA, the performance in the missing scenario is significantly lower than SEDA (Average 3.33%, 5.69% in PM, CM), indicating that our domain adaptation framework is indeed effective in extracting similar knowledge from external sources. Despite the removal of the DA framework, SEDA_DA is still better than other traditional models, mainly due to the advantages brought by the masked attention network $Mask\_A()$. In SEDA_SA, random matching of ego-graphs results in an average 0.60% and 0.72% performance decline in PM and CM. Ego-graphs alignment is necessary so that missing feature can be supplemented with more similar features. We will do more experiments later to test the validity of the ego-graph alignment. In SEDA_FD, datasets with larger scale differences require scale separation more than datasets with small differences. Scale separation



(a) PeMS_08(PM,Missing) (b) PeMS_08 (CM,Missing) (c) CD_S(PM,Missing) (d) CD_S(CM,Missing)

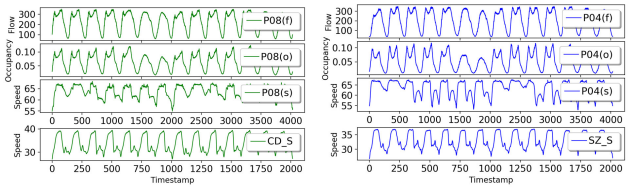
(e) PeMS_08(PM,Imputed) (f) PeMS_08(CM,Imputed) (g) CD_S(PM,Imputed) (h) CD_S(CM,Imputed)

Fig. 7. Comparison of models performance in PM/CM 0-90% scenarios on datasets PeMS_08(f) and CD_S.



(a) CD_S. (b) SZ_S. (c) PeMS_08. (d) PeMS_04.

Fig. 8. Graph structure visualization of datasets.



(a) Datasets of target domains. (b) Datasets of source domains.

Fig. 9. Typical sequences visualization of datasets.

yields an average 0.58% and 0.91% gain in prediction performance when PeMS_04(s) is source domain (See Figure 11 for the experimental results of SEDA (PeMS_04(s) → CD_S)). In SEDA_RN, the results show that learning the mapping relationship between the central singular values and the neighbor singular values is necessary, and it averagely improves the performance by 0.28%, 0.36% in PM, CM. More detailed experiments on the relation network later.

D. Analysis of External Dataset Similarity and Selection

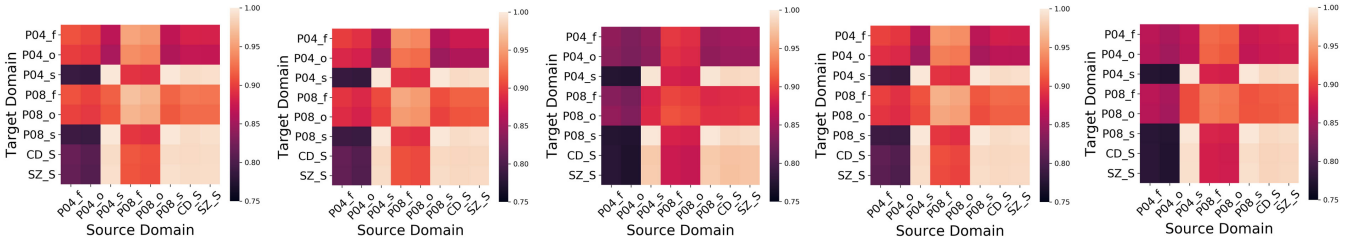
We test whether DSKL can give effective guidance in the selection of source domain through taking PeMS_04, PeMS_08, SZ_S and CD_S as examples. We first visualize their topologies as shown in Figure 8. The topologies of Chinese urban datasets (CD_S, SZ_S) are more complex and have more nodes, while those of American freeway datasets (PeMS_04, PeMS_08) are simpler and have fewer nodes. In addition, we visualize the typical sequences D of the eight datasets as shown in Figure 9. We can intuitively see that datasets of the same type (i.e., flow, speed, and occupancy) have high similarity. Even though the datasets CD_S and SZ_S are collected from two different cities Chengdu and Shenzhen (more than 1700 km apart), their data still shows great similarity, which indirectly proves that SEDA has great implemensibility.

Furthermore, we use DSKL to quantitatively calculate pairwise similarity of the eight datasets, as shown in Figure 10. In addition to the complete datasets, we also calculate the similarity when target domain has 50% and 90% data missing. As can be seen from the heat map, the quantified similarity is consistent with our intuitive observation. Speed datasets show high similarity, even between urban speed data and freeway speed data. The flow and occupancy data show high similarity across data types, because occupancy is defined in a way that is related to flow. The diagonal of the similarity matrix can be understood as the self-similarity of the dataset. If the internal heterogeneity of a dataset is high, its self-similarity is low; otherwise, its self-similarity is high. It is also worth noting that the similarity matrix is asymmetric. This means that transposing the location (in target domain or source domain) of a pair of datasets will have a different effect. Taking SZ_S and PeMS_04(f) as an example, when PeMS_04(f) is target domain, SZ_S can provide more knowledge because it is larger and has more diverse data. Conversely, when SZ_S is target domain, some missing data may not be able to match suitable data in PeMS_04(f). In addition, comparing the five heat maps, although the data missing in target domain has a slight influence on the value of similarity, it does not disturb the similarity order of the datasets. The most similar dataset can still be found easily.

We evaluated the relationship between the DSKL score (dataset similarity) and model performance by using seven datasets as source domains and the CD_S dataset as the target domain. To evaluate the contribution of external data, we compared the experiment results with the baseline model GSTAE, which is a state-of-the-art model designed for incomplete prediction on a single dataset. As shown in Figure 11, the experimental results indicate a close proportional relationship

TABLE VI
MODEL PERFORMANCE COMPARISON OF ABLATION EXPERIMENTS ON SZ_S, (PEMS_04(S))→CD_S DATASETS (RMSE/MAE/MAPE)

Missing Rate	SEDA_DA	SEDA_SA	SEDA_FD	SEDA_RN	SEDA
original	3.93/2.46/9.94	3.96/2.50/10.05	3.89/2.46/9.92, (3.94/2.47/9.99)	3.87/2.45/9.89	**3.90/2.44/9.89**
20%, PM	4.51/2.97/12.02	3.79/2.53/10.40	3.91/2.49/10.08, (4.11/2.61/10.55)	3.89/2.49/10.17	**3.93/2.47/9.99**
40%, PM	5.53/3.65/14.75	4.01/2.66/10.92	4.04/2.57/10.33, (4.23/2.73/10.93)	4.05/2.56/10.51	**4.10/2.54/10.13**
20%, CM	5.49/3.59/14.61	4.03/2.61/10.55	4.01/2.52/10.16, (4.20/2.66/10.77)	4.06/2.57/10.33	**3.95/2.50/10.02**
40%, CM	6.26/4.17/17.07	4.21/2.71/11.19	4.16/2.63/10.64, (4.27/2.87/11.66)	4.20/2.65/10.70	**4.12/2.59/10.29**



(a) Similarity in original. (b) Similarity in PM50%. (c) Similarity in PM90%. (d) Similarity in CM50%. (e) Similarity in CM90%.

Fig. 10. Dataset similarity heat maps in different missing scenarios.

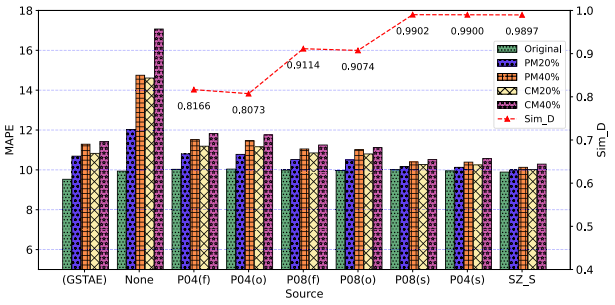
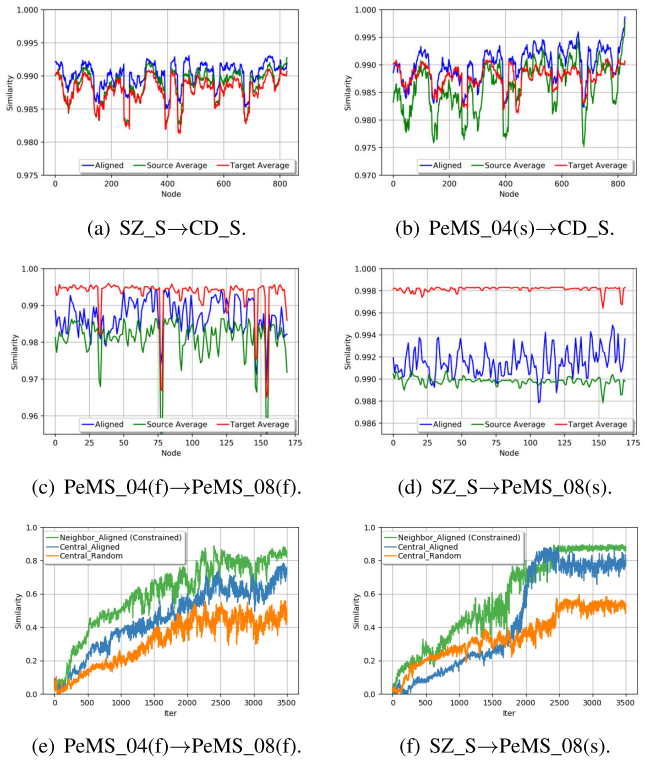


Fig. 11. Performance of SEDA with different source domains.

between DSKL scores and model performance. Higher DSKL scores correlate with greater benefits from knowledge transfer and improved model performance. Compared with GSTAE, we observed that in this case, when the DSKL score exceeds 0.9, our model outperforms GSTAE, suggesting that our model has gained more knowledge from the dataset with high DSKL score than GSTAE from a single dataset alone. This confirms the effectiveness of knowledge transfer. Conversely, when the DSKL score is below 0.9, although the transferred performance still exceeds that of no transfer (source domain is None), it does not surpass GSTAE's performance on a single dataset. This indicates that the knowledge from external datasets with low DSKL score may be not effectively integrated into the target domain prediction and cannot provide sufficient performance improvement. In summary, the DSKL score based on KL divergence can serve as a preliminary indicator to measure the contribution of external data and guide the selection of source domains.

E. Effectiveness Analysis of Ego-Graphs Alignment

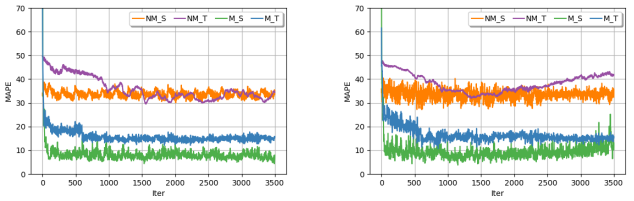
We test the benefits of ego-graphs alignment from both spatiotemporal perspectives. *Temporal perspective*: We calculate



(a) SZ_S→CD_S. (b) PeMS_04(s)→CD_S.
(c) PeMS_04(f)→PeMS_08(f). (d) SZ_S→PeMS_08(s).
(e) PeMS_04(f)→PeMS_08(f). (f) SZ_S→PeMS_08(s).

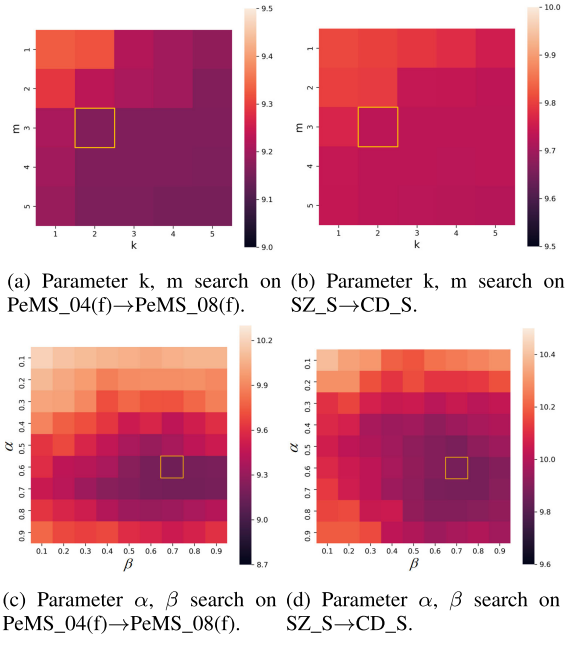
Fig. 12. Performance of ego-graphs alignment.

the distribution similarity of central node data between each ego-graph in target domain and 1) three aligned ego-graphs in source domain, 2) all the ego-graphs in source domain and 3) all the ego-graphs in target domain, to find out whether the ego-graph alignment can extract more similar data in source domain. We test it in SZ_S→CD_S, PeMS_04(s)→CD_S, PeMS_04(f)→PeMS_08(f), and SZ_S→PeMS_08(s) scenarios. The results are shown in Figure 12.(a)-(d), and we can find



(a) SZ_S→CD_S. (b) PeMS_04(s)→CD_S.

Fig. 13. Performance of relation network.



(a) Parameter k, m search on PeMS_04(f)→PeMS_08(f). (b) Parameter k, m search on SZ_S→CD_S.

(c) Parameter α, β search on PeMS_04(f)→PeMS_08(f). (d) Parameter α, β search on SZ_S→CD_S.

Fig. 14. Performance of parameters GridSearch.

that the similarity of aligned ego-graphs (blue lines) is higher than the average similarity between target domain ego-graphs and the whole source domain (green lines). When CD_S is target domain, the similarity of aligned ego-graphs (blue line) is sometimes higher than the average similarity between target domain ego-graphs and the whole target domain (red line), which means that ego-graphs alignment can even extract more similar data in source domain than the average level in target domain. Therefore, ego-graphs alignment can find similar alternative data of missing data.

Spatial Perspective: We record the cosine similarity of orthonormal bases across domains during SEDA training. We only select the base vectors corresponding to the 5 largest singular values (the total weight is more than 96%) in each pair of orthonormal bases. As shown in Figure 12.(e)(f), the green line is the similarity of orthogonal bases of the neighbor features between aligned ego-graphs in two domains. It is constrained by the loss function $\mathcal{L}_d$, so the similarity is the highest. The blue line is the similarity of the orthogonal bases of central feature between aligned ego-graphs. It has no constraint but still approaches the level of the green line. The orange line is the similarity of the orthogonal bases of central feature between the random matched ego-graphs, which shows the worst performance. Therefore, ego-graphs alignment is conducive to central orthogonal bases gap narrowing. This enables SEDA to obtain more suitable orthogonal bases from source domain to reconstruct the missing features.

F. Effectiveness Analysis of Relation Network

The relation network $R(x)$ is designed to learn the mapping relationship between the singular values of central node feature and that of neighbor node feature. We test the MAPE of 1) the generated and real central singular values in source domain ("M_S"), 2) the generated and real central singular values in target domain ("M_T"), 3) the generated central singular values and real neighbor singular values in target domain ("NM_T"), 4) the generated central singular values in target domain and real central singular values in source domain ("NM_S")), in scenario CD_S→SZ_S and CD_S→PeMS_04(s).

As shown in Figure 13, the results show that 1) there is a stable mapping relationship between the neighbor singular value and the center singular value, and it can be learned by the relation network (green line), 2) the relation network trained in source domain can still maintain good performance after being transferred to target domain (blue line), 3) directly using neighbor singular values as the missing singular values leads to a large error (purple line), 4) directly using central singular values in source domain as the missing singular values also leads to a large error (orange line). Hence, our proposed relation network is necessary and can effectively learn the mapping relationship to reduce the error of reconstructing singular values of missing feature.

G. Analysis of Model Parameter

We used the GridSearch [50] method to determine the key parameters of the model on PeMS_04(f)→PeMS_08(f) and SZ_S→CD_S, including k-hop, m most similar ego-graphs of the model and α, β of the loss function. We treat k and m as a set of parameters to be searched simultaneously, since they are both related to the model architecture. α and β are treated as another set of parameters since they are both hyperparameters in the loss function. Specifically, we first delimit the range of m and k, where $m \in \{1, 2, \ldots, 5\}$ and $k \in \{1, 2, \ldots, 5\}$, and freeze α and β both to 0.5. Then, we test the model performance by pairwise matching each m and k, and the results are as shown in Figure 14.(a)(b).

As we can see, the model achieves similar good performance in multiple m, k matching schemes, such as (m = 2, k = 5), (m = 3, k = 2,3,4,5), (m = 4, k = 2,3,4,5). In the case of similar performance, we choose the matching scheme with smaller m and k to avoid introducing too much computation. Therefore, we only take (m = 2, k = 5) and (m = 3, k = 2) into consideration. Furthermore, compared to (m = 2, k = 5) and (m = 3, k = 2), we prefer the matching scheme with smaller k, because k is exponential for the increase of model computation, therefor we finally choose the parameter scheme with m = 3, k = 2.

In the same way, we search for the optimal matching scheme for α, β with m = 3, k = 2. We set $\alpha, \beta \in \{0.1, 0.2, \ldots, 0.9\}$, and test the performance of the model by pairwise matching

TABLE VII
TRAINING TIME(S) OF MODELS

Model	DCRNN	AGCRN	3STL	FBDA	LSTM-GL-ReMF	BTTF	GSTAE	BATF(Imputation)	**SEDA**
CD_S	475.11	170.73	137.29	129.61	288.41	313.47	216.11	337.51	**201.6**
PeMS_08(f)	28.26	11.92	7.87	7.34	6.37	13.52	16.67	15.29	**18.12**

TABLE VIII
NOTATION LIST

Notation	Description
$\mathcal{G}$	graph of a dataset
$\mathcal{V}$, $\mathcal{E}$	set of nodes, edges
v_i, e_{ij}	i-th node, edge connecting v_i to v_j
n	number of nodes
$\mathcal{S}_i$	i-th ego-graph
k	number of hops of an ego-graph
$*^{\mathcal{S}}$,$*^{\mathcal{T}}$	* in source domain, target domain
η	topological vector of an ego-graph
F_c, F_n	central feature, neighbor feature of an ego-graph
M	missing mask
d_x, d_s	delay indexes of input, hidden state
U_i, V_i, S_i	left, right singular matrix, singular value matrix
$\mathcal{L}$	loss function

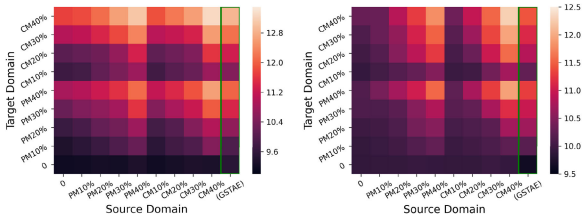
each α and β, the result is shown in Figure 14.(c)(d). We finally determine the optimal $\alpha = 0.6$ and $\beta = 0.7$.

H. Analysis of Time Complexity

The time complexity of SEDA mainly comes from ego-graphs alignment and masked attention network. In the process of ego-graphs alignment, we match m similar source domain neighbor ego-graphs for each target domain ego-graph, the time complexity is $o(m\left|\mathcal{V}^{\mathcal{T}}\right| log\left|\mathcal{V}^{\mathcal{S}}\right|)$. In masked attention network, w time slots need to be calculated. The time complexity is $o(wm\left|\mathcal{V}^{\mathcal{T}}\right| log\left|\mathcal{V}^{\mathcal{S}}\right|)$, where w, m is constants, and $w, m \ll \left|\mathcal{V}^{\mathcal{T}}\right|, \left|\mathcal{V}^{\mathcal{S}}\right|$. The ego-graphs alignment is performed at initialization, so the total time complexity of SEDA is $o(wm\left|\mathcal{V}^{\mathcal{T}}\right| log\left|\mathcal{V}^{\mathcal{S}}\right|)$. We test the real training time of SEDA and baselines in an epoch of CD_S and PeMS_08(f) on 12th Gen Intel(R) Core(TM) i9-12900K CPU and one GeForce RTX 3090 GPU, as shown in Table VII. Overall, SEDA's training time is on the same order of magnitude as other baselines. The excellent performance of SEDA does not result in excessive computational overhead.

I. Discussion on Model Limitations and Future Work

SEDA's predictive performance on complete datasets does not measure up that of traditional state-of-the-art models. This is due to the structure of its feature extraction component. The simple feature extraction network provides SEDA with implementation ease and improved computational speed, but also limits its performance on complete dataset. Moreover, the selection of source domains is crucial for maximizing SEDA's effectiveness. As illustrated in Figure 11, different data types (e.g., flow and speed) result in difference in dataset distribution, which can impede effective knowledge transfer. Additionally, we evaluated the impact of missing data in the source domain on the model's performance. As shown in Figure 15, even within the same data type, high rate of missing data in the source domain adversely affects SEDA's performance, making it inferior to traditional models. The significant disparity in data distribution between two domains or high levels of missing data in the source domain may lead to the failure of knowledge transfer. Therefore, enhancing SEDA's performance under low missing rate and further improving the model's robustness are potential areas for future work.



(a) PeMS_04→PeMS_08 (b) SZ_S→CD_S

Fig. 15. Experiment results with data missing in the source domain. The baseline performance is indicated within the green rectangle.

VI. CONCLUSION

Observation data missing will seriously damage the performance of traffic prediction. In this paper, we designed a novel Spatiotemporal Ego-graphs Domain Adaptation framework to predict traffic state in data missing scenarios. Based on the topological homogeneity, aligned ego-graphs from the external source domain provide alternative data matching with the missing data. Furthermore, a Dual-branch Cross reCoupling method is designed to reconstruct missing features according to the alternative data. It can greatly isolate the interference of the specificity in the external domain while introducing external knowledge. Experimental results on real datasets show that SEDA can achieve excellent prediction performance in the target domain with severe data missing by using the knowledge of external data. It improves the traffic prediction performance by more than 0.45% and 0.86% compared with the state-of-the-art knowledge transfer-based and incomplete prediction baselines. Ablation experiments and visualization analysis have also proved the effectiveness of SEDA components.

REFERENCES

[1] N. Chiabaut and R. Faitout, "Traffic congestion and travel time prediction based on historical congestion maps and identification of consensual days," *Transp. Res. C, Emerg. Technol.*, vol. 124, Mar. 2021, Art. no. 102920.

[2] E. Yao, T. Liu, T. Lu, and Y. Yang, "Optimization of electric vehicle scheduling with multiple vehicle types in public transport," *Sustain. Cities Soc.*, vol. 52, Jan. 2020, Art. no. 101862.

[3] Q. Wang, C. Xu, W. Zhang, and J. Li, "GraphTTE: Travel time estimation based on attention-spatiotemporal graphs," *IEEE Signal Process. Lett.*, vol. 28, pp. 239–243, 2021.

[4] W. Zhang, Q. Wang, D. Shi, Z. Yuan, and G. Liu, "Dynamic order dispatching with multiobjective reward learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18001–18011, Oct. 2022.

[5] K. Ramana et al., "A vision transformer approach for traffic congestion prediction in urban areas," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 4, pp. 3922–3934, Apr. 2023.

[6] Y. Zhang, K. Gao, Y. Zhang, and R. Su, "Traffic light scheduling for pedestrian-vehicle mixed-flow networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 4, pp. 1468–1483, Apr. 2019.

[7] L. Zhao et al., "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[8] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proc. 8th Workshop Syntax Semantics Struct. Stat. Transl. (SSST)*, Oct. 2014, pp. 103–111.

[9] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[10] Y. Huang, Y. Weng, S. Yu, and X. Chen, "Diffusion convolutional recurrent neural network with rank influence learning for traffic forecasting," in *Proc. 18th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./13th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2019, pp. 678–685.

[11] L. Bai, L. Yao, C. Li, and X. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 17804–17815.

[12] C. Chen, J. Kwon, J. Rice, A. Skabardonis, and P. Varaiya, "Detecting errors and imputing missing data for single-loop surveillance systems," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1855, no. 1, pp. 160–167, Jan. 2003.

[13] D. Ni and J. D. Leonard, "Markov chain Monte Carlo multiple imputation using Bayesian networks for incomplete intelligent transportation systems data," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1935, no. 1, pp. 57–67, Jan. 2005.

[14] L. Qu, L. Li, Y. Zhang, and J. Hu, "PPCA-based missing data imputation for traffic flow volume: A systematical approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 512–522, Sep. 2009.

[15] H. Tan, G. Feng, J. Feng, W. Wang, Y.-J. Zhang, and F. Li, "A tensor-based method for missing traffic data completion," *Transp. Res. C, Emerg. Technol.*, vol. 28, pp. 15–27, Mar. 2013.

[16] S. Tak, S. Woo, and H. Yeo, "Data-driven imputation method for traffic data in sectional units of road links," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 6, pp. 1762–1771, Jun. 2016.

[17] J. Yoon, J. Jordon, and M. Schaar, "Gain: Missing data imputation using generative adversarial nets," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5689–5698.

[18] Y. Wang, Y. Zhang, X. Piao, H. Liu, and K. Zhang, "Traffic data reconstruction via adaptive spatial–temporal correlations," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 4, pp. 1531–1543, Apr. 2019.

[19] D. Xu, H. Peng, C. Wei, X. Shang, and H. Li, "Traffic state data imputation: An efficient generating method based on the graph aggregator," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 13084–13093, Aug. 2022.

[20] X. Wu, M. Xu, J. Fang, and X. Wu, "A multi-attention tensor completion network for spatiotemporal traffic data imputation," *IEEE Internet Things J.*, vol. 9, no. 20, pp. 20203–20213, Oct. 2022.

[21] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Sci. Rep.*, vol. 8, no. 1, pp. 1–12, Apr. 2018.

[22] J.-M. Yang, Z.-R. Peng, and L. Lin, "Real-time spatiotemporal prediction and imputation of traffic status based on LSTM and graph Laplacian regularized matrix factorization," *Transp. Res. C, Emerg. Technol.*, vol. 129, Aug. 2021, Art. no. 103228.

[23] A. Baggag et al., "Learning spatiotemporal latent factors of traffic via regularized tensor factorization: Imputing missing values and forecasting," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 6, pp. 2573–2587, Jun. 2021.

[24] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[25] J. Li et al., "A domain adaptation framework for short-term traffic prediction," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Sep. 2021, pp. 3564–3569.

[26] P. Oza, V. A. Sindagi, V. V. Sharmini, and V. M. Patel, "Unsupervised domain adaptation of object detectors: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 6, pp. 4018–4040, Jun. 2024.

[27] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.*, vol. 74, no. 1, p. 47, 2002.

[28] X. Chen, Z. He, Y. Chen, Y. Lu, and J. Wang, "Missing traffic data imputation and pattern discovery with a Bayesian augmented tensor factorization model," *Transp. Res. C, Emerg. Technol.*, vol. 104, pp. 66–77, Jul. 2019.

[29] Y. Qu, Z. Li, X. Zhao, and J. Ou, "Towards real-world traffic prediction and data imputation: A multi-task pretraining and fine-tuning approach," *Inf. Sci.*, vol. 657, Feb. 2024, Art. no. 119972.

[30] A. Wang, Y. Ye, X. Song, S. Zhang, and J. J. Q. Yu, "Traffic prediction with missing data: A multi-task learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 4, pp. 4189–4202, Apr. 2023.

[31] Z. Li, L. Xia, Y. Xu, and C. Huang, "FlashST: A simple and universal prompt-tuning framework for traffic prediction," 2024, *arXiv:2405.17898*.

[32] J. Li, F. Guo, A. Sivakumar, Y. Dong, and R. Krishnan, "Transferability improvement in short-term traffic prediction using stacked LSTM network," *Transp. Res. C, Emerg. Technol.*, vol. 124, Mar. 2021, Art. no. 102977.

[33] X. Chen, J. Yang, and L. Sun, "A nonconvex low-rank tensor completion model for spatiotemporal traffic data imputation," *Transp. Res. C, Emerg. Technol.*, vol. 117, Aug. 2020, Art. no. 102673.

[34] X. Chen, M. Lei, N. Saunier, and L. Sun, "Low-rank autoregressive tensor completion for spatiotemporal traffic data imputation," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 12301–12310, Aug. 2022.

[35] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1–11.

[36] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[37] X. Chen and L. Sun, "Bayesian temporal factorization for multidimensional time series prediction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 4659–4673, Sep. 2022.

[38] Y. Ganin et al., "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2030–2096, 2016.

[39] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–11.

[40] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 4893–4902.

[41] Q. Xue, W. Zhang, and H. Zha, "Improving domain-adapted sentiment classification by deep adversarial mutual learning," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 5, pp. 9362–9369.

[42] Y. Xie, T. Chen, T. Pu, H. Wu, and L. Lin, "Adversarial graph representation adaptation for cross-domain facial expression recognition," in *Proc. 28th ACM Int. Conf. Multimedia*, Oct. 2020, pp. 1255–1264.

[43] X. Shen, Q. Dai, F.-L. Chung, W. Lu, and K.-S. Choi, "Adversarial deep network embedding for cross-network node classification," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 3, pp. 2991–2999.

[44] Q. Zhu, C. Yang, Y. Xu, H. Wang, C. Zhang, and J. Han, "Transfer learning of graph neural networks with ego-graph information maximization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 1766–1779.

[45] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," 2009, *arXiv:0902.3430*.

[46] C. Cortes and M. Mohri, "Domain adaptation in regression," in *Proc. 22nd Int. Conf. Algorithmic Learn. Theory*, Espoo, Finland. Berlin, Germany: Springer, Oct. 2011, pp. 308–323.

[47] X. Chen, S. Wang, J. Wang, and M. Long, "Representation subspace distance for domain adaptation regression," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 1749–1759.

[48] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang, "Complex networks: Structure and dynamics," *Phys. Rep.*, vol. 424, nos. 4–5, pp. 175–308, 2006.

[49] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sep. 1975.

[50] F.-K. Sun, C. Lang, and D. Boning, "Adjusting for autocorrelated errors in neural networks for time series," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 29806–29819.

Chen Xu received the B.S. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2019, where he is currently pursuing the Ph.D. degree in information and communication engineering. His research interests include spatiotemporal data mining and intelligent transportation systems.

Qiang Wang (Member, IEEE) received the Ph.D. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2008. He is currently a Professor with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. His research interests include wireless networks, intelligent transportation systems, VLSI, and machine learning.

Wenqi Zhang (Member, IEEE) received the B.S. and Ph.D. degrees from Beijing University of Posts and Telecommunications, Beijing, China, in 2016 and 2021, respectively. From June 2021 to August 2023, she was engaged in post-doctoral research with the School of Cyberspace Security, Beijing University of Posts and Telecommunications. She is currently a Researcher with the Sony (China) Research Laboratory. Her research interests include intelligent transportation systems, reinforcement learning, and federated learning.

Chen Sun (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from Nanyang Technological University, Singapore, in 2005. From August 2004 to May 2008, he was a Researcher with ATR Wave Engineering Laboratories, Japan, working on adaptive beam-forming and direction-finding algorithms of parasitic array antennas and theoretical analysis of cooperative wireless networks. In June 2008, he joined the National Institute of Information and Communications Technology, Japan, as an Expert Researcher, working on distributed sensing and dynamic spectrum access in TV white space. Since then, he has been contributing to the IEEE 1900.6 standard, IEEE 802.11af standard, and Wi-Fi alliance specifications for Wi-Fi networks in TV white space. In 2012, he joined Sony China as the Research Manager working on IEEE 802.19, ETSI, and 3GPP standards development. He served as the 802.19 standards and rapporteur of ETSI standards. He is currently the Head of Beijing Laboratory, Sony Research and Development Center.