# I. Supplementary Materials

## A. Theoretical Analysis

**Theorem 1.** *(Generalization Capability of Learnable Prompt $P$) Given a Spatio-Temporal model $\mathcal{F}_\Theta$, and the input spatio-temporal data $(\mathcal{G}, \mathcal{X})$, where graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$ is spatial component, $\mathcal{X} \in \mathbb{R}^{N \times T}$ is the temporal component. For any domain generalization transformation function $g \colon \mathbb{D}_{specific} \to \mathbb{D}_{common}$, which satisfies $(\hat{\mathcal{G}}, \hat{\mathcal{X}}) = g(\mathcal{G}, \mathcal{X})$,, there exists a learnable prompt $P$ that satisfies:*

$$\mathcal{F}_\Theta\big((\mathcal{G}, \mathcal{X}) + P\big) = \mathcal{F}_\Theta\big(g(\mathcal{G}, \mathcal{X})\big). \tag{1}$$

To further illustrate the domain-generalized transformation $g(\cdot)$ into spatio-temporal scenario, we decompose it into three specific sub-transformations.

**Proposition 1.** *Given the input spatio-temporal data $(\mathcal{G}, \mathcal{X})$, we extract the spatial scanned sequences $\mathbf{S}$, temporal scanned sequences $\mathbf{X}$, and ST-delay scanned sequences $\mathbf{D}$. Any domain-generalizing transformation function $g \colon \mathbb{D}_{specific} \to \mathbb{D}_{common}$ can be decoupled into the composition of the following sub-transformations:*

- ***Spatial pattern transformation:*** *Modifies the spatial node features to obtain the generalized spatial representation $\hat{\mathbf{S}} = g_S(\mathbf{S})$.*
- ***Temporal pattern transformation:*** *Modifies the temporal features to obtain the generalized temporal representation $\hat{\mathbf{X}} = g_T(\mathbf{X})$.*
- ***ST-delay pattern transformation:*** *Modifies the delay-aware features to obtain the generalized spatio-temporal representation $\hat{\mathbf{D}} = g_D(\mathbf{D})$.*

*Proof.* ***Spatial pattern transformations.***

We set spatial embedding difference as $\Delta\mathbf{S} = \hat{\mathbf{S}} - \mathbf{S}$. Then, the embedding of spatial node can be calculate as:

$$\begin{aligned}
\hat{\mathbf{H}} &= (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot \hat{\mathbf{S}} \cdot \mathbf{W} && (2)\\
&= (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot (\mathbf{S} + \Delta\mathbf{S}) \cdot \mathbf{W} && (3)\\
&= (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot \mathbf{S} \cdot \mathbf{W} + (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot \Delta\mathbf{S} \cdot \mathbf{W} && (4)\\
&= \mathbf{H} + (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot \Delta\mathbf{S} \cdot \mathbf{W}, && (5)
\end{aligned}$$

where $\mathbf{W}$ is a linear projection, $\mathbf{A}$ denotes the adjacency matrix, $\epsilon$ is a parameter.

For a learnable prompt $P = [\alpha_1, \ldots, \alpha_F] \in \mathbb{R}^{1 \times D}$, we can perform a similar split:

$$\begin{aligned}
\mathbf{H}_P &= (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot (\mathbf{S} + [1]^N \cdot P) \cdot \mathbf{W} && (6)\\
&= (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot \mathbf{S} \cdot \mathbf{W} + (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot [1]^N \cdot P \cdot \mathbf{W} && (7)\\
&= \mathbf{H} + (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot [1]^N \cdot P \cdot \mathbf{W} && (8)\\
&= \mathbf{H} + [d_i + 1 + \epsilon]^N \cdot P \cdot \mathbf{W}, && (9)
\end{aligned}$$

Where $[1]^N \in \mathbb{R}^{N \times 1}$ denotes a column vector with $N$ ones, and $[d_i + 1 + \epsilon]^N \in \mathbb{R}^{N \times 1}$ denotes a column vector where the value of the $i$-th row is $d_i + 1 + \epsilon$, with $d_i$ representing the degree of node $v_i$. To obtain the same global spatial representation $h_{\mathcal{G},P}$, we have:

$$h_{\mathcal{G},S} = h_{\mathcal{G},P} \to \text{Sum}(\hat{\mathbf{H}}) = \text{Sum}(\mathbf{H}_P), \tag{10}$$

where $\text{Sum}(\mathbf{M}) = \sum_i m_i$ denotes the operation that calculates the sum of each row in the matrix. We can further simplify the above equation as:

$$\begin{aligned}
& h_{\mathcal{G},S} = h_{\mathcal{G},P} && (11)\\
\to\ & \text{Sum}(\hat{\mathbf{H}}) = \text{Sum}(\mathbf{H}_P) && (12)\\
\to\ & \text{Sum}(\mathbf{H} + (\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot \Delta\mathbf{S} \cdot \mathbf{W}) = \text{Sum}(\mathbf{H} + [d_i + 1 + \epsilon]^N \cdot P \cdot \mathbf{W}) && (13)\\
\to\ & \text{Sum}((\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot \Delta\mathbf{S} \cdot \mathbf{W}) = \text{Sum}([d_i + 1 + \epsilon]^N \cdot P \cdot \mathbf{W}), && (14)
\end{aligned}$$

Where the result of $((\mathbf{A} + (1+\epsilon) \cdot \mathbf{I}) \cdot \Delta\mathbf{S}) \in \mathbb{R}^{N \times D}$, and the linear transformation $\mathbf{W} \in \mathbb{R}^{D \times D'}$. We first calculate $\Delta h_{\mathcal{G},P} = \text{Sum}([d_i + 1 + \epsilon]^N \cdot P \cdot \mathbf{W}) \in \mathbb{R}^{D'}$. We can obtain that:

$$\begin{aligned}
\Delta h^i_{\mathcal{G},P} &= \sum_{j=1}^{D} \sum_{k=1}^{N} (d_k + 1 + \epsilon) \cdot \alpha_j \cdot \mathbf{W}_{j,i} && (15)\\
&= \sum_{j=1}^{D} (D + N + N \cdot \epsilon) \cdot \alpha_j \cdot \mathbf{W}_{j,i}, && (16)
\end{aligned}$$

where $h^i_{\mathcal{G},P}$ denotes the value of the $i$-th dimension in $h_{\mathcal{G},P}$, $D = \sum_{k=1}^{N} d_k$ denotes the total degree of all nodes in the whole graph, $\alpha_j$ denotes the $j$-th learnable parameter in $P$, and $\mathbf{W}_{j,i}$ denotes the parameter in $\mathbf{W}$. As for $\Delta h_{\mathcal{G},S} = \text{Sum}((\mathbf{A} + (1+\epsilon)\cdot\mathbf{I})\cdot\Delta\mathbf{S}\cdot\mathbf{W})$, we assume $(\mathbf{A} + (1+\epsilon)\cdot\mathbf{I})\cdot\Delta\mathbf{S} = \mathbf{B} \in \mathbb{R}^{N\times D}$. Then we have:

$$\Delta h^i_{\mathcal{G},S} = \sum_{j=1}^{D}(\sum_{k=1}^{N}\beta_{k,j})\cdot\mathbf{W}_{j,i}, \tag{17}$$

where $\beta_{k,j}$ denotes the learnable parameter in $\mathbf{B}$. According to above analysis, to obtain a same global spatial representation $h_{\mathcal{G},P}$ with a certain $h_{\mathcal{G},S}$, we have:

$$h^i_{\mathcal{G},P} = h^i_{\mathcal{G},S}\ ,\ \text{for every } i \in [1, D'] \tag{18}$$

$$\rightarrow \Delta h_{\mathcal{G},P} = \Delta h^i_{\mathcal{G},S} \tag{19}$$

$$\rightarrow \alpha_j = \frac{\sum_{k=1}^{N}\beta_{k,j}}{D+N+N\cdot\epsilon}\ ,\ j \in [1, D] \tag{20}$$

Therefore, for an arbitrary domain-generalizing transformation $g_S(\cdot)\colon \mathbb{D}_{\text{specific}} \rightarrow \mathbb{D}_{\text{common}}$, there exists a learnable embedding $P$ that satisfies the above conditions and can obtain the generalizable spatial representation for Spatio-Temporal model $\mathcal{F}_\Theta$. The proof process for Temporal Pattern Transformations $g_T(\cdot)$ and ST-delay pattern transformation $g_D(\cdot)$ is similar to the above. □

Proposition 1 demonstrates that our proposed learnable domain adapter provides comprehensive coverage for all spatio-temporal feature transformations. The domain adapter introduces a shared feature modification vector $P \in \mathbb{R}^D$ that is applied uniformly across domain features. This design achieves the same effect as applying independent feature modifications to each domain individually.

### *B. Datasets*

TABLE I: Overview of Datasets Used in Pre-training and Evaluation.

| Pre-training | Dataset | Traffic Metric | Location | #Regions | Data Interval | Time Span | Data Size |
|---|---|---|---|---|---|---|---|
| Yes | CAD4 | Traffic flow | Bay Area, USA | 2352 | 5 min | 2020/01/01–2020/03/15 | 50.8M |
| | CAD7 | Traffic flow | Los Angeles, USA | 1859 | 5 min | 2020/01/01–2020/03/15 | 40.2M |
| | CAD8 | Traffic flow | Los Angeles, USA | 1022 | 5 min | 2020/01/01–2020/03/15 | 22.1M |
| | CAD12 | Traffic flow | Los Angeles, USA | 953 | 5 min | 2020/01/01–2020/03/15 | 20.6M |
| | PEMS04 | Traffic flow | California, USA | 307 | 5 min | 2018/01/01–2020/02/28 | 5.2M |
| | PEMS08 | Traffic flow | California, USA | 170 | 5 min | 2016/07/01–2020/08/31 | 3.0M |
| | NYC-TAXI | Taxi demand | New York City, USA | 263 | 30 min | 2016/01/01–2021/12/31 | 27.7M |
| | TrafficZZ | Traffic speed | Zhengzhou, China | 676 | 30 min | 2022/03/05–2022/04/05 | 0.9M |
| | TrafficHZ | Traffic speed | Hangzhou, China | 672 | 30 min | 2022/03/05–2022/04/05 | 0.9M |
| | TrafficCD | Traffic speed | Chengdu, China | 728 | 30 min | 2022/03/05–2022/04/05 | 1.0M |
| | TrafficJN | Traffic speed | Jinan, China | 576 | 30 min | 2022/03/05–2022/04/05 | 0.8M |
| | METR-LA | Traffic speed | Los Angeles, USA | 207 | 5 min | 2012/03/01–2022/04/30 | 3.6M |
| | PEMS-BAY | Traffic speed | Bay Area, USA | 325 | 5 min | 2017/01/01–2017/04/30 | 11.2M |
| No | CAD3 | Traffic flow | California, USA | 480 | 5 min | 2020/03/01–2020/04/30 | 8.4M |
| | CAD5 | Traffic flow | California, USA | 211 | 5 min | 2020/03/01–2020/04/30 | 3.7M |
| | CHI-TAXI | Taxi demand | Chicago, USA | 77 | 30 min | 2021/01/01–2021/12/31 | 1.3M |
| | NYC-BIKE | Bike trajectory | New York City, USA | 540 | 30 min | 2020/01/01–2020/12/31 | 9.5M |
| | TrafficSH | Traffic speed | Shanghai, China | 896 | 30 min | 2022/03/05–2022/04/05 | 1.3M |
| | PEMS07M | Traffic speed | California, USA | 228 | 5 min | 2017/05/01–2017/08/31 | 2.9M |
| | SZ-DIDI | Traffic index | Shenzhen, China | 627 | 10 min | 2018/01/01–2018/02/28 | 5.3M |
| | CD-DIDI | Traffic index | Chengdu, China | 524 | 10 min | 2018/01/01–2018/02/28 | 4.5M |

The dataset statistics utilized in this study are summarized in Table I. A variety of large-scale, real-world public datasets were employed to ensure comprehensive training and evaluation. These datasets cover multiple categories of traffic-related data and originate from several major metropolitan areas, including New York City, Chicago, Los Angeles, Shanghai, and Shenzhen.

The first column of Table I indicates whether each dataset was utilized during the pre-training phase of the proposed Damba-ST model. For supervised evaluation, portions of the datasets involved in downstream tasks were held out to assess the model's performance in supervised settings. The datasets comprise the following five types of traffic metricits:

- **Traffic Flow Data**: including CAD-X and PEMS-X collected in the United States;
- **Taxi Demand Data**: including X-TAXI from New York Ciitty and Chicago;
- **Traffic Speed Data**: including Traffic-X from major Chinese cities, as well as METR-LA, PEMS-BAY, and PEMS07M from the Los Angeles and Bay Area regions;
- **Bicycle Trajectory Data**: including NYC-BIKE from New York City;
- **Traffic Index Data**: including X-DIDI from Shenzhen and Chengdu.

### *C. Implementation Details*

To ensure a fair and comprehensive comparison, the experimental settings closely follow those of OpenCity [13]. The specific configurations for each experimental scenario and the dataset splits are detailed as follows:

- **In-distribution Prediction Setting**: The proposed Damba-ST is directly evaluated, while baseline models are trained separately on each target dataset. For the NYC-TAXI dataset, data from 2016 to 2020 are used for training, the first two months of 2021 are used for validation, and the remaining ten months of 2021 are used for testing. For the PEMS-BAY dataset, the training, validation, and testing splits follow a 0.5/0.1/0.4 ratio. For the CAD8 and CAD12 datasets, the training, validation, and testing splits follow a 0.8/0.1/0.1 ratio.
- **Zero-shot Prediction Setting**: Damba-ST is directly applied to the test sets, while baseline models are trained and evaluated under a fully supervised paradigm. For the CAD3, CAD5, PEMS07M, and TrafficSH datasets, the training, validation, and testing splits are 0.5, 0.1, and 0.4, respectively. For the CHI-TAXI and NYC-BIKE datasets, the split ratios are 0.2, 0.2, and 0.6. Notably, none of these datasets are seen during the pre-training stage of Damba-ST.
- **Deployment Practicality Setting**: To assess real-world deployment potential, we directly utilize the publicly released large models from their original repositories for evaluation. Following the experimental protocols of UrbanGPT and UniST, the final month of CHI-TAXI data is used as the test set, and the prediction accuracy of the first six forecasting steps is measured across models.

### *D. Baselines*

We have carefully selected 12 recent and advanced models from the spatio-temporal prediction community as our baselines, all of which have demonstrated considerable success in traffic prediction tasks. These models include:

(1) **Recent Spatio-Temporal Foundation Models**:

- **OpenCity** [13]: OpenCity is a foundation model for traffic forecasting that unifies spatio-temporal modeling and zero-shot generalization in a single framework. It combines Transformer-based temporal encoders and graph-based spatial modules to capture both periodic and dynamic traffic patterns across multiple cities. By pre-training on large, heterogeneous traffic datasets, OpenCity learns generalized representations that transfer effectively to unseen regions and long-range forecasting tasks, achieving impressive zero-shot performance even without fine-tuning.
- **UniST** [12]: UniST is a universal spatio-temporal foundation model for urban forecasting that jointly handles graph-based and grid-based data. It uses two-stage training consisting of (1) large-scale pre-training on diverse urban datasets to capture general spatio-temporal patterns and (2) prompt-guided fine-tuning to adapt dynamically to new cities or tasks. UniST employs learned memory modules for spatial and temporal components, enabling robust few-shot and zero-shot generalization across heterogeneous urban scenarios, all within a single unified framework.

(2) **Mamba-based Models:**

- **SpoT-Mamba** [36]: SpoT-Mamba first constructs node embeddings via random walk sequences to capture spatial structure, and then applies Mamba-style temporal scanning to model long-distance temporal dynamics. By combining walk-based spatial encoding and selective state-space temporal modeling, it effectively handles both local and global spatio-temporal patterns, delivering strong performance on traffic forecasting benchmarks.
- **STG-Mamba** [37]: STG-Mamba embeds selective state-space models within a Graph Selective State-Space Block (GS3B) to dynamically identify and model relevant spatio-temporal features. To adaptively refine the evolving graph structure, it introduces Kalman filtering Graph Neural Networks, which integrate multi-granularity embeddings via Kalman-filter-inspired updates. Built as an encoder-decoder consisting of stacked GS3B blocks, STG-Mamba achieves linear computational complexity, outperforming both GNN and Transformer baselines in accuracy.

(3) **Attention-Based Models:**

- **ASTGCN** [38]: ASTGCN integrates spatio-temporal attention with graph convolutions to model traffic flow. It captures recent, daily, and weekly patterns through separate branches, with each branch employing attention-enhanced spatial and temporal convolutions. The fused output improves forecasting accuracy on real-world traffic datasets.

- **STWA** [6]: This model dynamically generates location- and time-specific parameters by leveraging latent stochastic representations, enabling adaptive spatio-temporal modeling. It incorporates a windowed attention mechanism for efficiency and achieves strong performance on standard traffic forecasting benchmarks.
- **PDFormer** [39]: PDFormer models traffic flow by capturing dynamic spatial dependencies with spatial self-attention and explicitly handling propagation delays via a delay-aware feature module, thereby improving long-range spatio-temporal forecasting accuracy.

(4) **GNNs-based Models:**

- **STGCN** [40]: This method applies graph convolution to capture spatial dependencies. Temporal dependencies are modeled using gated 1D convolutions, enabling the model to effectively capture sequential patterns. By stacking spatio-temporal convolutional blocks, STGCN jointly learns spatial and temporal patterns, achieving efficient and accurate traffic prediction without relying on recurrent networks.
- **TGCN** [41]: This model combines Graph Convolutional Networks (GCN) and Gated Recurrent Units (GRU) to model spatial and temporal dependencies in traffic forecasting. The GCN captures the spatial structure of urban road networks, while the GRU models temporal dynamics in traffic data. This hybrid approach enables T-GCN to effectively predict traffic conditions by jointly learning spatial correlations and temporal dynamics.
- **GWN** [8]: This model integrates adaptive graph structure learning with dilated temporal convolutions to simultaneously capture spatial dependencies and long-range temporal patterns, enabling effective spatio-temporal forecasting in traffic data.
- **MTGNN** [42]: This model is designed for multivariate time series forecasting by dynamically learning both graph structure and temporal dependencies. It introduces a graph learning module to infer inter-series relationships and employs dilated temporal convolutions to capture long-range temporal patterns. The model integrates spatial and temporal learning in a unified architecture without requiring a predefined graph.
- **STSGCN** [43]: This method jointly captures spatial and temporal dependencies by constructing spatio-temporal synchronous graphs. It applies graph convolutions over both spatial and short-term temporal connections within a unified block (STSGCL), enabling effective learning of localized spatio-temporal patterns. Multiple STSGCL blocks are stacked to capture long-term spatio-temporal dynamics effectively.