Efficient Prompt Learning for Traffic Prediction

Anonymous Author(s)

ABSTRACT

Accurate traffic prediction is essential for optimizing transportation systems, enhancing resource allocation, and improving overall urban administration. Spatio-temporal graph neural networks (GNNs) have achieved state-of-the-art performance and have been widely used in various spatio-temporal prediction scenarios. However, these prediction methods often exhibit low generalization ability, struggling with distribution shifts caused by spatio-temporal dynamics. To address this challenge, we propose a novel approach to enhance the generalization and adaptation of spatio-temporal GNNs through efficient prompting. Specifically, we introduce a lightweight and efficient prompt tuning framework, named SimpleST, which is model-agnostic and can be applied to various spatio-temporal GNNs. SimpleST facilitates adapting pre-trained spatio-temporal GNNs to novel distributions via a prompt tuning mechanism while keeping the model parameters fixed. This prompt tuning mechanism reduces the overhead and complexity of adaptation, enabling efficient utilization of pre-trained models for out-of-distribution generalization. Extensive experiments conducted on five real-world urban spatio-temporal datasets demonstrate the superiority of our approach in terms of prediction accuracy and computational efficiency.

1 INTRODUCTION

Spatio-temporal prediction in traffic [14] is a vital task that involves forecasting traffic conditions over time and space, enabling the optimization of transportation systems, efficient resource allocation, and improved urban administration. Recently, significant strides have been made in the field of spatio-temporal graph neural networks (GNNs) to achieve state-of-the-art performance in spatio-temporal prediction [13, 22]. Spatio-temporal GNNs leverage the power of graph structures by extending convolutional and attentive neural networks. They excel at capturing spatial relationships and effectively propagating information across the graph, enabling them to model intricate dependencies present in spatio-temporal traffic data [19]. The learning process of these models involves training on spatio-temporal data to learn the model parameters, which are then used to make predictions on unseen testing data.

Significant strides have been achieved in spatio-temporal graph neural architectures, showcasing their adaptability across a wide array of applications. Nonetheless, existing spatio-temporal GNNs still face two key challenges. **(1) Spatio-temporal distribution shift.** A pivotal challenge arises when well-trained models encounter discrepancies in data distribution between their training and testing phases. Due to urban dynamics, or changes in population behavior, the testing data often deviates unexpectedly from the trained distribution. These distribution shifts pose formidable obstacles to model generalization in practical scenarios, necessitating enhanced generalization capabilities for pre-trained models on out-of-distribution (OOD) data. **(2) Intensive computation in model updates.** To better adapt to spatio-temporal changes, recent approaches [3, 9, 29] attempt to employ advanced data-adaptive

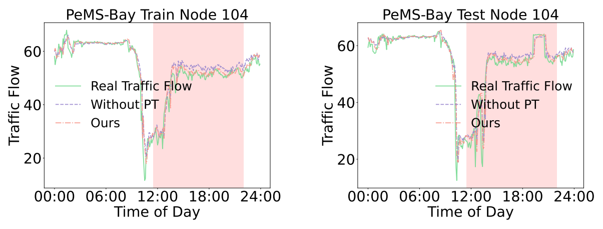


Figure 1: Data distribution shift of PeMS-Bay

R#qiyP-Q3, R#iGjG-Q4

modules or fine-tuning paradigms to accommodate shifted distributions. However, their intricate model structures hinder the accurate capture of spatio-temporal dynamics with limited fine-tuning data. More importantly, fine-tuning all parameters of these models on new data is time-consuming and resource-intensive, impeding rapid adaptation to latency-sensitive and resource-constraint scenarios.

Motivated by the above limitations of existing spatio-temporal prediction methods, in this study, we introduce a model-agnostic spatio-temporal prompt learning paradigm called SimpleST. Our paradigm is specifically designed to efficiently adapt deployed spatio-temporal prediction models in response to distribution shifts in unseen data, which is shown in Figure 1 and Figure 4. Specifically, we draw inspiration from the successful application of prompt-tuning techniques in the field of textual data [26] and propose integrating a specially designed prompt neural network into pre-trained models. Our paradigm enables customized prompts for various downstream tasks, which explicitly guide the predictions of computationally intensive spatio-temporal neural networks. This approach allows more precise control of the prediction process, ensuring that the model focuses on shifted spatio-temporal patterns and captures the inherent complexities of the data. By incorporating our lightweight SimpleST, we empower the equipped spatio-temporal prediction models to efficiently adapt to the frequent distribution shifts with minimal resource cost and facilitate their sustained application in lifelong spatio-temporal prediction tasks.

Our lightweight SimpleST framework brings two key benefits for advancing spatio-temporal prediction: **(1) Adaptability**. Our SimpleST provides a flexible mechanism to adapt the spatio-temporal model's behavior to distribution shift. Our prompt learning component is specifically designed to capture the relevant spatio-temporal context and distribution characteristics from shifted data, allowing the model to align its predictions with the new data distribution. **(2) Efficiency**. It enables fine-tuning specific prompt parameters rather than the entire model, resulting in faster training time. By updating only a subset of parameters in our simple spatio-temporal prompt network, SimpleST enables effective model updates using a limited number of fine-tuning labels.

In a nutshell, the main contributions of this paper are threefold:

- **Practical problem.** This work aims to enhance the adaptability of spatio-temporal pre-trained models to distribution shifts, which are frequently encountered in real-world scenarios characterized by spatial and temporal dynamics. The improved adaptability enables the model to maintain high prediction performance even when faced with previously unseen data.

- **New methodology.** This study presents SimpleST, a model-agnostic spatio-temporal prompt learning paradigm that integrates a simple prompt network with prediction models, allowing for effective adaptation and generalization in diverse spatio-temporal contexts. Additionally, an in-depth analysis is provided to justify the model's capability in alleviating distribution shifts and achieving enhanced efficiency.
- **Extensive experiments.** We perform extensive experiments by adapting SimpleST to diverse spatio-temporal models and thoroughly evaluate its effectiveness and efficiency across multiple real-world datasets. Through comparative analyses with numerous state-of-the-art fine-tuning methods, SimpleST demonstrates superior performance updates in less time. For instance, our method SimpleST achieves an average accuracy improvement of 3-6% while being 1 to 45 times faster than other methods. Our code and data can be accessed at https://anonymous.4open.science/r/SimpleST.

2 RELATED WORK

Traffic Prediction. Accurate traffic prediction is essential for optimizing transportation systems, mitigating congestion, and improving the efficiency of urban mobility. To tackle this challenge, considerable efforts have been devoted to developing various neural network architectures capable of capturing complex spatial and temporal correlations across different time slots and locations. For instance, recurrent neural networks (RNNs) [7, 16], attention mechanisms [15, 25] and temporal convolutional networks [23], have been successfully employed to capture the transitional patterns in spatio-temporal data, further enhancing the predictive capabilities of the models. Recent advancements have focused on designing spatio-temporal (GNNs) [10, 14, 30] to effectively encode spatial correlations among different locations. These approaches leverage the inherent graph structure of the data, allowing for the high-order modeling of interactions between spatially connected nodes, thereby enhancing the accuracy and providing state-of-the-art performance. However, a prevailing challenge faced by most existing spatio-temporal models is their limited capability to handle distribution shifts in contexts of spatio-temporal dynamics.

Prompt-tuning Techniques. The objective of prompt-tuning is to optimize and fine-tune prompts in order to improve the performance and generalization of pre-trained models on specific tasks or domains. Inspired by the success of prompt-tuning in language modeling, researchers have extended its application to various domains, such as computer vision [2, 11] and graph neural networks (GNNs) [3, 4, 8, 20]. For example, Liu et al. [8] unify graph pre-training and downstream tasks into a common task template. Motivated by these advancements, this work introduces a spatio-temporal prompt learning framework that leverages the success of prompt-tuning to customize and refine pre-trained models, providing an effective solution to improve their adaptability in the context of spatio-temporal dynamics, which is critical in real-life urban management.

OOD Problem in Spatio-Temporal Prediction. The OOD challenge in spatial-temporal prediction presents a daunting hurdle in accurately forecasting events within dynamically evolving spatial and temporal systems. In domains like traffic forecasting [3, 9, 12, 27] and climate modeling [18], encountering OOD scenarios can lead to unreliable predictions and diminished model efficacy. To address this, researchers have proposed solutions. For instance, AGCRN [3] advocates for a data-adaptive module to handle distribution shifts across different time segments, while ASTGCN [9] implements spatial-temporal graph convolutional networks with attention mechanisms to capture dynamic shifts in data distributions. However, these approaches are hindered by inefficiencies and resource intensive in updating models due to their intricate structures like multi-layer GCNs. Moreover, they lack adaptability across diverse traffic prediction methodologies.

3 METHODOLOGY

In this section, we present our model-agnostic spatio-temporal prompt learning paradigm: SimpleST, and the overall framework is shown in Figure 2. SimpleST incorporates the temporal convolutional network (TCN) to effectively process the temporal dynamics within the data, and extract patterns and relationships over time. Afterwards, the processed data is fed into GNN-based pre-trained models, enabling efficient adaptation and generalization to diverse spatio-temporal contexts.

In the following sections, we provide descriptions of spatio-temporal data, spatio-temporal graph and task formalization in Section 3.1. Then we provide detailed illustrations of our method SimpleST in Section 3.2. Afterwards, we provide efficiency analysis of SimpleST in Section 3.3. Finally, we provide an illustration of SimpleST in Section 3.4.

3.1 Spatio-Temporal Prediction

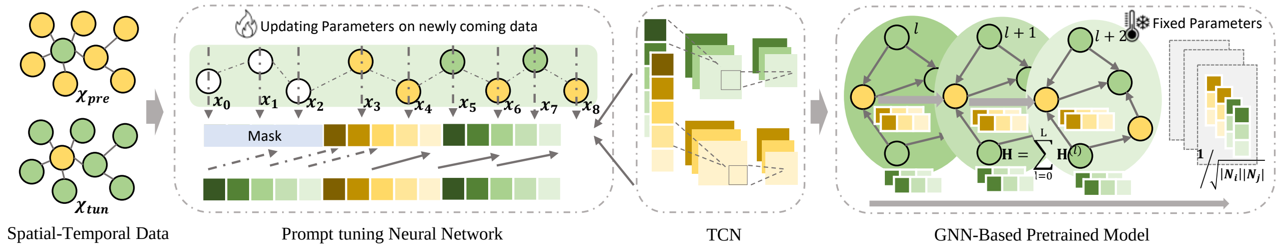
Spatio-temporal (ST) prediction focuses on forecasting data that is spatially and temporally distributed in the urban space [17].

Prompt. Prompt in this paper refers to a mechanism that guides the predictions of computationally intensive spatio-temporal neural networks. This prompt is specifically designed to capture relevant spatio-temporal context and distribution characteristics from shifted data. Therefore, the term "prompt" is utilized to refer to a specific guiding mechanism tailored for enhancing the adaptability and generalization of spatio-temporal prediction models in response to distribution shifts, distinct from prompts commonly used in language models like LLMs.

R#DtiT-Q1, R#Y8nt-Q2

Spaio-Temporal Data. Formally, the target spatio-temporal data can be denoted by a three-way tensor $\mathcal{X} \in \mathbb{R}^{R\times T\times F}$, where R denotes the number of spatial regions (*e.g.* urban districts, road segments), T denotes the number of time slots (*e.g.* quarters, hours, days), and F denotes the dimensionality of the concerned features (*e.g.* the number of crime types). An element $\mathcal{X}_{r,t,f} \in \mathbb{R}$ denotes the value of the f-th feature for the r-th spatial region in the t-th time slot. And $\mathcal{X}_t \in \mathbb{R}^{R\times F}$ is used to represent the time-specific matrix.

Spaio-Temporal Graph. Besides the spatio-temporal data $\mathcal{X}$, it is common to work with a spatio-temporal graph that records the correlations between the regions and the time slots. This graph can be represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ is a set of nodes


Updating Parameters on newly coming data
Fixed Parameters
$\mathcal{X}_{pre}$
$\mathcal{X}_{tun}$
Mask
x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
l $l+1$ $l+2$
$\mathbf{H} = \sum_{l=0}^{L} \mathbf{H}^{(l)}$
$\frac{1}{\sqrt{|N_i||N_j|}}$
Spatial-Temporal Data
Prompt tuning Neural Network
TCN
GNN-Based Pretrained Model

R#iGJG-Q2

Figure 2: Architecture Overview of the Spatio-Temporal Prompt Learning Paradigm SimpleST. The framework consists of four components: spatial-temporal data, prompt tuning neural network, the TCN network and GNN-based pre-trained models. White node denotes the masked nodes in the spatio-temporal graph; And its embedding vector is denoted as mask in the Figure. And for the input of spatial-temporal data, for $\mathcal{X}_{pre}$, the node with green color is center point and nodes around it with yellow color are neighbour nodes; for $\mathcal{X}_{tun}$, the node with yellow color is center point and nodes around it with green color are neighbour nodes. And vectors of them marked with corresponding colors lierk green or yellow respectively.

representing urban regions, and $\mathcal{E}$ is the set of edges that encode both the spatial interrelations and the temporal transition relations between the region nodes. The node set $\mathcal{V}$ is associated with a node feature matrix $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{|\mathcal{V}|}\} \in \mathbb{R}^{|\mathcal{V}|\times d}$, where $\mathbf{x}_i \in \mathbb{R}^d$ represents the feature vector of the node v_i.

Task Formalization. Based on the above definitions, the spatio-temporal prediction task is to predict the future values of the ST tensor $\mathcal{X}$, with the help of the historical records of $\mathcal{X}$ and the ST graph $\mathcal{G}$. This can be formalized as learning a spatio-temporal model $g(\cdot)$ with parameter set Θ_g as follows:

$$(\mathcal{X}_{t+1}, \cdots, \mathcal{X}_{t+T'}) = g(\mathcal{X}_{t-T+1}, \cdots, \mathcal{X}_t, \mathcal{G}; \Theta_g) \quad (1)$$

3.2 Spatio-Temporal Prompt Tuning

3.2.1 Pre-training and Tuning Paradigm. In real urban scenarios, spatio-temporal data typically exhibits daily variations, resulting in a dynamic and evolving distribution. However, these variations pose a challenge for the existing ST models trained using static method, limiting their ability to accurately predict new data that is temporally distant from the historical training samples. To fill this gap, our SimpleST framework adopts the pre-training and tuning paradigm. It requires the ST model, pre-trained on fixed historical data, to continuously adapt to newly updated data. This enables the model to maintain its predictive accuracy over time, effectively capturing the evolving nature of the spatio-temporal data. In this pre-training and tuning paradigm, we split the entire ST data into four subsets $\mathcal{X} = (\mathcal{X}_{pre}, \mathcal{X}_{val}, \mathcal{X}_{tun}, \mathcal{X}_{tst})$, as follows:

$$\begin{aligned} \mathcal{X}_{pre} &= \mathcal{X}_{t-T+1:t-T+T_{val}}, & \mathcal{X}_{val} &= \mathcal{X}_{t-T+T_{val}+1:t-T+T_{pre}}, \\ \mathcal{X}_{tun} &= \mathcal{X}_{t-T+T_{pre}+1:t}, & \mathcal{X}_{tst} &= \mathcal{X}_{t+1:t+T'}, \end{aligned} \quad (2)$$

where the pre-training data $\mathcal{X}_{pre}$, validation data $\mathcal{X}_{val}$, tuning data $\mathcal{X}_{tun}$, and test data $\mathcal{X}_{tst}$ are arranged in chronological order. The index t denotes the time slot separating the test data from the other two sets. T indicates the total number of time slots in the combined pre-training and tuning data, T_{pre}, T' represents the length of the training and test data, respectively. Following the pre-training and tuning paradigm, ST models are pre-trained on $\mathcal{X}_{pre}$, fine-tuned on $\mathcal{X}_{tun}$, and ultimately evaluated on $\mathcal{X}_{tst}$.

A common approach following this paradigm is the pre-train-and-fine-tune method [8], which involves fine-tuning a pre-trained model on newly updated data for model adaptation. However, the fine-tuning process can be computationally expensive, particularly for heavy but accurate ST models based on GNNs. In light of the effectiveness and efficiency of prompt tuning [21] in adapting pre-trained models to unseen data, our SimpleST proposes a prompt tuning approach for ST prediction, enabling efficient adaptation. Specifically, instead of fine-tuning the entire ST model on new data, SimpleST fixes the pre-trained model during the tuning process, and optimizes a prompt network for model adaption. This prompt tuning method can be described as the objective below:

$$\begin{aligned} &\arg\min_{\Theta_h} \mathcal{L}(g(h(\mathcal{X}_{tun}, \mathcal{G}; \Theta_h); \Theta_g), \mathcal{X}_{tun}), \\ &\text{with } \Theta_g = \arg\min_{\Theta_g} \mathcal{L}(g(\mathcal{X}_{pre}; \Theta_g), \mathcal{X}_{pre}), \end{aligned} \quad (3)$$

where $h(\cdot)$ and $g(\cdot)$ denote the prompt neural network and the original ST model, respectively. Their contained parameters are referred as Θ_h, Θ_g, respectively. $\mathcal{L}(\cdot)$ denotes the loss function such as the squared mean error. This equation presents the two-step optimization process of our prompt tuning framework. In the first phase, the original ST model $g(\cdot)$ is optimized using the pre-training data $\mathcal{X}_{pre}$. Here $g(\cdot)$ can be any proposed ST model. In the second phase, our SimpleST optimizes the plug-in prompt neural network $h(\cdot)$ using the newly updated data $\mathcal{X}_{tun}$. It mitigates the distribution shift of the new data $\mathcal{X}_{tun}$ by learning a transformation, adapting $\mathcal{X}_{tun}$ to the pre-trained ST model $g(\cdot)$ customized to the old data $\mathcal{X}_{pre}$. Compared to fine-tuning the entire ST model $f(\cdot)$, our prompt neural network $g(\cdot)$ is able to achieve comparable or even better performance with much less training steps, due to its lightweight network architecture which is easier to optimize.

3.2.2 Time-aware Prompt Network. To capture the essential temporal dependencies that impact the region relationships in spatio-temporal data, we integrate a temporal convolutional network (TCN) into our prompt neural network. This fusion introduces dynamism into the transformed features. In formal terms, the operation of our SimpleST, which combines the prompt neural network with TCN, can be described as follows:

$$\begin{aligned} \tilde{\mathcal{X}}_{r,t} &= \mathbf{W}_4 \bar{\mathbf{H}}_{r,t} + \mathcal{X}_{r,t}; & \bar{\mathbf{H}}_r &= \sigma(\mathbf{W}_3 \tilde{\mathbf{H}}_r + \mathbf{b}_2), \\ \tilde{\mathbf{H}}_r &= \sigma(\delta(\mathbf{W}_2 * \mathbf{H}_r + \mathbf{b}_1)); & \mathbf{H}_{r,t} &= \mathbf{W}_1 \mathcal{X}_{r,t}, \end{aligned} \quad (4)$$

where $\tilde{\mathcal{X}} \in \mathbb{R}^{R \times T_{tun} \times F}$ denotes the spatio-temporal data transformed by our prompt network, with $T_{tun} = T - T_{pre}$ denoting the number of time slots in the tuning data. $\bar{\mathbf{H}} \in \mathbb{R}^{R \times T_{tun} \times d}$ denotes the intermediate embedding with hidden dimensionality d. The results of the TCN, which convolves the original T_{tun} temporal dimensions into T'_{tun} dimensions, are denoted by $\tilde{\mathbf{H}} \in \mathbb{R}^{R \times T'_{tun} \times d}$. $\mathbf{H} \in \mathbb{R}^{R \times T_{tun} \times d}$ denotes the initial embeddings for all regions and time slots. The learnable parameters of our prompt neural network are $\mathbf{W}_4 \in \mathbb{R}^{F \times d}$, $\mathbf{W}_3 \in \mathbb{R}^{T_{tun} \times T'_{tun}}$, $\mathbf{W}_2 \in \mathbb{R}^{(T_{tun} - T'_{tun} + 1) \times 1}$, $\mathbf{W}_1 \in \mathbb{R}^{d \times F}$, and $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^{d}$. And $*$ denotes the convolution operator. $\sigma(\cdot), \delta(\cdot)$ denote the ReLU activation and the dropout function, respectively. A skip connection is utilized in the final layer of our prompt network, to directly utilize the original ST data. The output $\tilde{\mathcal{X}}$ of our prompt network has the same dimensionality as the original ST data $\mathcal{X}$, and thus can be seamlessly used by any spatio-temporal models for performance enhancement.

Algorithm 1: The **SimpleST** Learning Algorithm

```
Input: The spatial-temporal graph $\mathcal{G}$, the maximum epoch
    number $E$, the learning rate $\eta$.
Output: Traffic flow or crime rate $\mathbf{H}$, trained parameters in
    $\Theta_f$ of Prompt neural network, and $\Theta_g$ of
    GNN-based neural network.
1  Initialize all parameters in $\Theta_g$ and $\Theta_f$;
2  Split the date into train, validation, test and prompt
     $\mathcal{X} = (\mathcal{X}_{pre}, \mathcal{X}_{val}, \mathcal{X}_{tun}, \mathcal{X}_{tst})$;
3  Train the framework SimpleST by Equation 3;
4  for $epoch = 1, 2, ..., E$ do
5      Train the GNN-based pre-train model via $\mathcal{X}_{pre}$;
6      for $\theta_g \in \Theta_g$ do
7          $\theta_g \leftarrow \theta_g - \eta \cdot \frac{\partial \mathcal{L}}{\partial \theta_g}$
8      end
9      Validate the GNN-based pre-trained model via $\mathcal{X}_{val}$;
10 end
11 for $epoch = 1, 2, ..., E'$ do
12     Freeze parameters of GNN-baed pre-train model and
         update the prompt neural network via Equation 3 via
         $\mathcal{X}_{tun}$;
13     Compute the MAE loss $\mathcal{L}$ following Equation 3;
14     Minimize the loss $\mathcal{L}$ by Equation 7 using gradient
         decent with learning rate $\eta$;
15     for $\theta_f \in \Theta_f$ do
16         $\theta_f \leftarrow \theta_f - \eta \cdot \frac{\partial \mathcal{L}}{\partial \theta_f}$
17     end
18 end
19 Return $\mathbf{H}$ and all parameters $\Theta_g$ and $\Theta_f$.
```

3.3 In-depth Discussion

This section makes further discussions to study two research questions: (1) How does the prompt network alleviate the distribution shift of spatio-temporal data? (2) How is the efficiency of the proposed SimpleST in comparison to the fine-tuning method and spatio-temporal prediction baselines?

Prompt Network as Data Projector. Our SimpleST model leverages vanilla spatio-temporal prediction loss functions, such as mean absolute error (MAE), to optimize and maximize the accuracy of the final spatio-temporal (ST) prediction task. While no explicit constraints are placed on mitigating distribution shifts, we demonstrate that our prompt network design is inherently trained to function as a data editor for the original ST data. This is evident through the following observations with the corresponding condition:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial \mathcal{L}}{\partial \tilde{\mathcal{X}}} \cdot \frac{\partial \tilde{\mathcal{X}}}{\partial \theta} = \frac{\partial \mathcal{L}}{\partial (\mathcal{X} + \nabla \mathcal{X})} \cdot \frac{\partial \nabla \mathcal{X}}{\partial \theta}, \tag{5}$$

where $\nabla \mathcal{X} = \mathbf{W}_4 \bar{\mathbf{H}}$ denotes the adjustable output of our prompt network. By incorporating a skip connection to include the original data $\mathcal{X}$, $\nabla \mathcal{X}$ functions as an editing component. The equation presented decomposes the gradient of the loss function $\mathcal{L}$ with respect to a specific learnable parameter θ from the prompt network into two components: the gradient of $\mathcal{L}$ with respect to the edited ST data, and the gradient of the editing value with respect to the parameter θ. In essence, the training objective of our SimpleST is to learn a spatio-temporal data editor that yields improved performance.

The Effectiveness of Prompt Network on Alleviation the Distribution Dismatch of Training and testing. The efficacy of prompt learning networks in addressing distribution shifts in spatio-temporal prediction tasks can be theoretically explained through a rigorous analysis of how these networks function as data editors to align training and testing distributions. We delve deeper into the theoretical underpinnings of the prompt tuning network and its role in mitigating distribution mismatches: R#iCJG-Q1

About the prompt tuning mechanism, the prompt tuning process can be represented as $\mathcal{X}_{t+1}, ..., \mathcal{X}_{t+T'} = g(h(\mathcal{X}_{t-T+1}, ..., \mathcal{X}_t, \mathcal{G}; \theta_h), \mathcal{G}; \theta_g)$; Here, $g(.)$ denotes teh spatio-temporal model. $\mathcal{X}_{t+1}, ..., \mathcal{X}_{t+T'}$ is the model prediction based on the input data and the prompt network. $\mathcal{G}$ is the spatio-temporal graph. θ_g denotes the parameters of the pretrained model. θ_h denotes the parameters of the prompt network. During this process, the gradient update of the original model is shown as $\theta'_g = \theta_g - \nabla_{\theta_g} \mathcal{L}$ and the gradient update of the prompt network is shown as $\theta'_h = \theta_h - \nabla_{\theta_h} \mathcal{L}$. Here $\mathcal{L}$ denotes the loss of the spatio-temporal framework.

About mitigating distribution shifts: The prompt network acts as a data editor by modifying the model's predictions to better match the testing distribution. This can be formulated as:

$$\begin{aligned} \mathcal{X}_{t+1}, ..., \mathcal{X}_{t+T'} &= g(h(\mathcal{X}_{t-T+1}, ..., \mathcal{X}_t, \mathcal{G}; \theta_h), \mathcal{G}; \theta_g) \\ &= g(h(\mathcal{X}_{t-T+1}, ..., \mathcal{X}_t, \mathcal{G}; \theta'_h - \nabla_{\theta_h} \mathcal{L}), \mathcal{G}; \theta_g) \end{aligned} \tag{6}$$

Thus, the gradient of the prompt network guides the model to update its parameters in a way that minimizes the discrepancy between the training and testing distributions, thereby mitigating distribution mismatches.

Model Efficiency Analysis. We demonstrate the efficiency advantages of our SimpleST from two perspectives: a comparison of the number of parameters and a comparison of time complexity. Firstly, we observed that the optimization operations in both the pre-training phase and the tuning phase are nearly identical. The only difference lies in the parameter set used, as illustrated below:

$$\begin{aligned} &\text{Pre-training:} && \theta := \theta - \eta \cdot \frac{\partial \mathcal{L}}{\partial \theta}, \text{ with } \theta \in \Theta_{\mathrm{g}}, \\ &\text{Prompt Tuning:} && \theta := \theta - \eta \cdot \frac{\partial \mathcal{L}}{\partial \theta}, \text{ with } \theta \in \Theta_{\mathrm{h}}. \end{aligned} \tag{7}$$

The two phases employ the same MAE loss function $\mathcal{L}$ [23] and the same learning rate η. However, there is a significant difference in the number of parameters, as empirically $|\Theta_g| > C \times |\Theta_h|$ where $|\Theta|$ denotes the number of parameters and $C \geq 10$. This confers a substantial efficiency advantage to the tuning phase of our SimpleST, not only by reducing the number of optimization operations but also by facilitating easier convergence during the optimization process.

Secondly, the pre-trained model is GNN-based which has a higher time complexity due to its costly graph information propagation paradigm. While our lightweight prompt tuning network consists of only 2-layer TCN and 2 fully-connected layers. Specifically, the time complexity of the GNN-based pre-trained model is $O(|\mathcal{E}| \times L \times d)$, where $|\mathcal{E}|$ denotes the number of edges, and L denotes the number of graph layers. In contrast, the prompt tuning neural network $f(\cdot)$ only requires $O(L' \times d^2)$ for prompt training, where L' represents the number of MLP layers, and d is the hidden dimensionality.

In conclusion, our analysis of the number of parameters and time complexity shows that the proposed prompt learning model outperforms GNN-based ST models in terms of tuning efficiency. This makes it a promising framework for large-scale data in real-world spatial-temporal prediction scenarios.

3.4 Algorithm Illustration

In this section, we provide the detailed algorithm of our framework SimpleST in Algorithm 1 in Appendix. In Algorithm 1, our framework SimpleST provides detailed algorithmic steps. Firstly, we initialize all parameters (Algorithm 1, Step 1). Next, we train the GNN-based model by updating the parameters Θ_g until it becomes proficient. The prompt tuning neural network is trained iteratively while keeping the GNN-based model fixed. To enhance the performance of the prompt tuning neural network, we employ the MAE loss [23], as utilized in previous studies on traffic prediction. The MAE loss is computed using Equation 3. We iteratively update the prompt tuning neural network (Equation 7) until convergence is achieved. Finally, the procedure concludes by returning the parameters Θ_g and Θ_f.

4 EXPERIMENTS

We assess the performance of our SimpleST through evaluations on two spatio-temporal prediction tasks: traffic prediction and crime forecasting. The experiments aim to address following questions:

- **RQ1**: How does our SimpleST compare to various methods in terms of performance?
- **RQ2**: What is the impact of each component on the performance of our SimpleST?
- **RQ3**: How efficient is of our SimpleST compared with other state-of-the-art methods?
- **RQ4**: What effects do different hyperparameter settings have on the performance of our SimpleST?

Table 1: Data Description of 5 Traffic Prediction Datasets

Traffic Data	Point-based Datasets				
Datasets	PeMSD04	PeMSD08	PeMSD03	PeMSD07	PeMS-Bay
Sensors	307	170	358	883	325
Data	16,992	17,856	26,208	28,224	52,116
Interval	5 minutes	5 minutes	5 minutes	5 minutes	5 minutes

4.1 Experimental Setup

Datasets Description. We evaluate the performance of our model on traffic prediction using five real-world traffic flow datasets: PeMSD04, PeMSD07, PeMSD03, PeMSD08, and PeMS-Bay. These datasets capture traffic flow data collected at 5-minute intervals. Following established practices [3, 6], we construct the urban spatial graph based on the road network for each dataset. Detailed statistics for each dataset are presented in Table 1 in Appendix. We also provide details of data splitting in Appendix.
Compared Methods: (1) ASTGNN [9]: AGCRN's core concept lies in utilizing an attention-based graph convolution network to grasp temporal and spatial dynamics and correlations. (2) AGCRN [3]: Its proposal involves incorporating node-adaptive parameter learning and a data-adaptive graph generation module to capture node-specific patterns and address data shift challenges in traffic prediction tasks. (3) MTGNN [23]: Its objective is to capture one-way relationships using graph learning techniques. Throughout this process, additional factors such as traffic dynamics or weather patterns are also taken into account. (4) STG-NCDE [5]: It employs two separate modules, one for temporal processing and the other for spatial processing. These modules are then seamlessly integrated into a unified framework to capture saptial and temporal correlations. We also compare our method SimpleST with CauSTG [29]. CauSTG leverages the inherent spatio-temporal relationships within the data. It enables the transfer of invariant ST relations to out-of-distribution (OOD) scenarios. We compare our approach to Invariant Risk Minimization (IRM) [1], another out-of-distribution (OOD) method that aims to estimate invariant correlations across multiple training distributions. We provide hyperparameter settings of these methods in Appendix.

4.2 Overall Effectiveness

We evaluate the effectiveness of our SimpleST by comparing it to baselines on the task of urban spatio-temporal prediction. SimpleST in these experiments utilizes ASTGCN, AGCRN, MTGNN and STG-NCDE as the base or backbone models. The reason that we choose these four base models is that these four methods incorporate a data adaptive module or other modules to alleviate the data distribution issue.We also evaluate the performance of SimpleST with CauSTG and IRM. The results are presented in Table 2 (urban spatio-temporal prediction datasets including PeMSD04, PeMSD08, PeMSD03, PeMSD07 and the large-scale dataset PeMS-Bay). Table 5 shows the results with the base MTGNN in terms of crime datasets of New York City and Chicago. We make the following observations for experiment results: R#Y8nt-Q3, R#iGjG-Q3

- **Superior performance**: Our SimpleST framework consistently outperforms almost all other baseline methods across five urban spatio-temporal prediction datasets and two crime datasets. This demonstrates the effectiveness of our prompt learning network in capturing distribution shifts between the pretrained and tuning data. In contrast, existing baselines experience a decline in performance due to the distribution gap. These advantages can be primarily attributed to our carefully designed spatio-temporal prompt tuning paradigm with the successful injection of spatio-temporal context distilled from the downstream data. Furthermore, the superior performance of our method in crime prediction results surpassing other baseline models underscores its exceptional generalization capabilities across diverse time series data types, such as crime data. R#Y8nt-Q3, R#iGjG-Q3

- **Significant improvements on large-scale data**: It is important to highlight that our SimpleST exhibits a considerably larger performance gap compared to the baselines when evaluated on the large-scale dataset PeMS-Bay. This outcome can be attributed to the heightened difficulty faced by non-adaptive baselines in handling the substantial distribution shift present in PeMS-Bay, which encompasses a broader time range. In contrast, our SimpleST effectively addresses this challenge by adeptly adapting itself to the shifted domain through its prompt tuning network.
- **Variations among different backbone models**: Our prompt tuning paradigm in SimpleST effectively mitigates the distribution shift issue for all backbone models. And MTGNN and AGCRN incorporate TCN modules to capture temporal dynamics, while ASTGCN incorporates an auxiliary temporal modeling view using the GRU network. Such diverse modeling techniques enhance the generalization ability of these models, resulting in a smaller disparity in performance.
- **Limitations of the OOD Methods**: Our findings indicate that integrating the OOD methods CauSTG and IRM with base models does not consistently yield performance improvements. In some instances, performance even decreased. This suggests that directly combining OOD methods with pretrained urban spatio-temporal prediction models may not be an effective strategy for addressing the OOD challenge. We attribute this observation to the significant parameter disparity between CauSTG and certain base models like AGCRN. This imbalance potentially hinders the sufficient training of CauSTG's parameters on tuning datasets.

4.3 Efficiency Comparison

We study the efficiency of our SimpleST framework by comparing it to three tuning techniques: training randomly-initialized ST models on the tuning data (i.e., Base), tuning the OOD methods on the tuning data (i.e., CauSTG and IRM) and fine-tuning pretrained ST models on the tuning data (i.e., Finetune). The tuning time from start to model convergence is recorded. The evaluation is done on a server with 48 cores of Intel(R) Xeon(R) CPU Max 9468 CPU max 2101 MHz and CPU min 800 MHz, and 8 NVIDIA H100 80GB HBM3 GPUs. In Table 3, we show the tuning time and GPU costs of using different backbone models, using two-week tuning data, respectively. In Table 7 (shown in Appendix), we present the running time of tuning models with 1-day, 1-week, and 2-week tunning data. MTGNN is employed as the backbone in this experiment. We also provide efficiency results of crime prediction in Table 6. From the results, we have the following conclusions:

1) Efficiency and GPU cost of prompt tuning: The advantageous tuning efficiency of our SimpleST model is evidenced by its ability to significantly reduce the tuning time and GPU cost required on different datasets. The higher efficiency and lower resource cost are achieved by focusing the tuning process on a smaller parameter set specific to the prompt network. By doing so, we effectively reduce the computational overhead associated with optimization calculations. Moreover, the reduced parameter set of the prompt network helps constrain the solution space of the model. This constrained solution space facilitates easier convergence during the training process. As a result, our SimpleST model can achieve optimal performance with fewer iterations and computations compared to models with larger parameter spaces. Overall, the streamlined tuning process of our SimpleST model not only saves computational resources but also enables faster convergence and more efficient utilization of data. This improvement in tuning efficiency contributes to the overall effectiveness and practicality of our model across different datasets and problem domains. When it comes to crime prediction, our method continues to excel with superior efficiency and reduced resource costs compared to the baseline approaches. R#Y8nt-Q3, R#iGjG-Q3

2) Comparing fine-tuning to training from scratch: While both compared tuning techniques optimize the same number of model parameters, we generally observe higher tuning efficiency with the fine-tuning method. This can be attributed to the pretraining process, which provides better starting points for the fine-tuning method, enabling faster convergence. However, there are instances where training from scratch outperforms fine-tuning. These cases reflect the uncertainty of whether the pretrained model state is advantageous or detrimental for training on the tuning data.

3) Efficiency under different tuning data size: As the amount of tuning data increases, specifically for durations of 1 day, 1 week, and 2 weeks, the tuning time for all three methods naturally increases due to the growing complexity of the data. However, the efficiency advantage of our SimpleST model becomes even more apparent and pronounced in these scenarios. Notably, our SimpleST model displays remarkable efficiency by maintaining its advantage and, in some cases, even enhancing it when dealing with larger tuning sets. This efficiency advantage is a result of the model's streamlined architecture and its ability to effectively leverage the provided prompt network. By focusing on tuning a smaller parameter set specific to the prompt network, our model reduces the computational burden associated with optimization calculations. As a consequence, our SimpleST model demonstrates its efficiency by effectively handling larger tuning sets while maintaining its performance and convergence speed. This efficiency advantage is especially crucial in real-world applications where dealing with substantial amounts of data is common and time is a critical factor. In summary, our SimpleST model showcases its efficiency by effectively managing the increased complexity and tuning time that come with larger tuning sets. This efficiency advantage further underscores the practicality and effectiveness of our model in real-world scenarios.

4) Impact of backbone models: By referring to Table 3 , it becomes evident that SimpleST effectively expedites the tuning process for various ST models, irrespective of their size. This attribute holds significant value in real-world applications. From the results, we observed that our prompt tuning neural network significantly improved the efficiency of different baselines, saving the time cost by approximately 1 time to 45 times. Our method, SimpleST, demonstrates a more significant efficiency improvement when applied to

Table 2: Overall Effectiveness Comparison

Data		PeMSD04			PeMSD08			PeMSD03			PeMSD07			PeMS-Bay		
Base Models	Metrics	MAE ↓	RMSE ↓	MAPE ↓	MAE ↓	RMSE ↓	MAPE ↓	MAE ↓	RMSE ↓	MAPE ↓	MAE ↓	RMSE ↓	MAPE ↓	MAE ↓	RMSE ↓	MAPE ↓
ASTGNN	Base	20.33	32.66	13.38%	16.95	26.88	10.72%	15.32	25.26	15.44%	23.65	35.34	9.26%	2.28	4.53	5.44%
	Base + CauSTG	20.17	32.54	13.26%	16.31	26.03	10.43%	15.06	24.93	14.72%	22.87	34.19	9.12%	2.18	4.49	4.76%
	Base + IRM	20.02	32.50	13.30%	16.27	25.85	10.49%	15.05	24.90	14.73%	22.76	34.01	9.08%	2.17	4.50	4.78%
	Base + Finetune	20.01	32.47	13.24%	**15.98**	**25.27**	**10.04%**	14.98	24.86	14.69%	22.72	33.99	9.06%	2.15	4.47	**4.52%**
	Base + **Ours**	**19.47**	**31.85**	**13.08%**	16.24	25.73	10.38%	**14.76**	**24.79**	**14.53%**	**22.05**	**33.97**	**8.95%**	**1.96**	**4.31**	4.64%
	Best Improve	*4.42%*	*2.54%*	*2.29%*	*6.07%*	*4.47%*	*3.28%*	*3.79%*	*1.90%*	*6.26%*	*7.26%*	*4.03%*	*3.46%*	*16.33%*	*5.10%*	*17.24%*
AGCRN	Base	21.03	33.93	13.71%	17.36	27.41	10.75%	16.76	26.22	15.89%	21.02	33.66	8.94%	1.84	4.02	4.22%
	Base + CauSTG	21.67	34.93	14.15%	17.44	27.46	10.73%	16.63	28.93	15.75%	21.23	33.68	9.02%	2.00	4.59	4.68%
	Base + IRM	21.22	33.56	13.65%	17.20	26.75	10.65%	17.72	26.02%	15.68%	21.03	33.61	8.90%	1.83	3.95	4.10%
	Base + Finetune	21.04	33.52	13.63%	17.17	26.60	10.61%	16.69	25.99	15.66%	20.95	33.57	8.87%	1.80	3.93	4.07%
	Base + **Ours**	**20.98**	**32.75**	**13.54%**	**16.16**	**25.20**	**10.39%**	**16.54**	**25.49**	**15.52%**	**20.83**	**33.51**	**8.82%**	**1.72**	**3.79**	**3.83%**
	Best Improve	*3.29%*	*6.66%*	*4.51%*	*7.92%*	*8.97%*	*3.46%*	*1.33%*	*13.50%*	*2.38%*	*1.92%*	*0.51%*	*2.27%*	*16.28%*	*21.11%*	*22.19%*
MTGNN	Base	20.39	32.61	13.34%	17.58	26.92	10.96%	15.85	25.93	15.07%	20.92	33.68	8.85%	2.01	4.32	4.65%
	Base + CauSTG	20.26	32.38	13.39%	17.17	26.34	11.65%	15.58	26.27	15.07%	20.86	33.57	8.93%	1.71	3.93	3.85%
	Base + IRM	19.84	31.83	13.56%	16.03	25.33	10.40%	15.59	25.34	14.85%	20.82	33.45	8.85%	1.83	3.94	3.97%
	Base + Finetune	19.44	31.85	13.05%	15.94	25.29	10.57%	15.36	25.39	15.25%	20.80	33.44	8.80%	1.80	3.91	4.01%
	Base + **Ours**	**19.42**	**31.54**	**13.02%**	**15.52**	**24.69**	**10.06%**	**14.87**	**24.73**	**14.29%**	**20.72**	**33.37**	**8.76%**	**1.70**	**3.81**	**3.77%**
	Best Improve	*3.29%*	*3.39%*	*0.53%*	*13.27%*	*9.03%*	*15.81%*	*6.59%*	*6.23%*	*6.72%*	*0.97%*	*0.93%*	*1.92%*	*18.24%*	*13.39%*	*23.34%*
STG-NCDE	Base	19.87	32.09	13.26%	16.32	25.78	11.07%	15.90	26.16	15.26%	20.87	33.73	8.99%	1.97	4.26	4.33%
	Base + CauSTG	19.77	31.86	13.21%	16.23	25.69	11.21%	15.43	25.90	15.11%	20.76	33.43	8.83%	1.90	3.98	4.12%
	Base + IRM	19.50	31.82	13.15%	16.20	25.30	10.92%	15.73	25.92	15.15%	20.74	33.40	8.86%	1.88	3.90	4.02%
	Base + Finetune	19.46	31.80	13.13%	16.18	25.21	10.84%	15.64	25.88	15.02%	20.72	33.36	8.85%	1.87	3.87	3.96%
	Base + **Ours**	**19.42**	**31.78**	**13.10%**	**15.64**	**24.76**	**10.17%**	**15.07**	**24.32**	**14.17%**	**20.68**	**33.29**	**8.78%**	**1.80**	**3.83**	**3.86%**
	Best Improve	*2.32%*	*0.97%*	*1.22%*	*4.35%*	*4.12%*	*10.23%*	*5.51%*	*5.57%*	*7.69%*	*0.92%*	*1.32%*	*2.39%*	*9.44%*	*11.23%*	*12.18%*

Table 3: Efficiency of Total Training Time (Minutes) and GPU Memory Cost (GB)

Data		PeMSD04		PeMSD08		PeMSD03		PeMSD07		PeMS-Bay	
Base Models	Metrics	Time ↓	GPU Cost ↓	Time ↓	GPU Cost ↓	Time ↓	GPU Cost ↓	Time ↓	GPU Cost ↓	Time ↓	GPU Cost ↓
ASTGNN	Base	118.733	6.523	189.652	2.811	343.887	9.451	334.768	19.922	230.127	10.431
	Base + CauSTG	207.895	7.108	220.712	5.315	467.891	9.841	515.723	20.808	243.783	12.367
	Base + IRM	101.334	6.678	145.782	2.901	276.910	9.305	290.018	19.920	180.272	10.510
	Base + Finetune	92.796	6.674	130.260	2.868	259.797	9.303	280.456	19.914	179.191	10.431
	Base + **Ours**	**74.535**	**6.417**	**107.947**	**2.453**	**229.952**	**8.623**	**256.164**	**16.864**	**109.532**	**8.892**
	Best Improve	*1.789*	*10.768%*	*1.045*	*116.673%*	*1.035*	*14.125%*	*1.013*	*23.387%*	*1.101*	*39.080%*
AGCRN	Base	24.996	5.785	22.5561	2.346	122.978	6.873	52.073	16.793	187.091	7.301
	Base + CauSTG	214.493	6.361	189.026	3.047	595.509	8.020	394.927	20.529	741.769	10.133
	Base + IRM	22.101	5.789	15.193	2.403	109.898	6.892	35.624	16.810	167.893	7.235
	Base + Finetune	19.012	5.783	13.522	2.346	103.768	6.883	30.303	16.793	156.886	7.207
	Base + **Ours**	**10.347**	**4.012**	**8.034**	**1.758**	**67.101**	**4.777**	**20.125**	**11.449**	**117.547**	**6.876**
	Best Improve	*19.730*	*58.549%*	*22.528*	*73.322%*	*7.875*	*67.887%*	*18.624*	*79.308%*	*5.310*	*47.368%*
MTGNN	Base	20.401	6.195	16.538	1.844	27.716	3.531	59.940	15.416	144.893	6.137
	Base + CauSTG	187.205	10.459	143.846	4.838	94.649	7.277	234.976	17.733	81.591	9.402
	Base + IRM	29.713	6.193	8.249	1.837	29.557	3.520	45.303	15.423	123.785	6.134
	Base + Finetune	23.036	6.189	8.432	1.836	16.381	3.516	20.541	15.420	107.665	6.121
	Base + **Ours**	**5.604**	**4.393**	**3.103**	**1.717**	**10.782**	**3.233**	**13.959**	**7.625**	**68.472**	**5.971**
	Best Improve	*32.406*	*138.083%*	*45.357*	*187.771%*	*7.778*	*125.085%*	*15.833*	*132.564%*	*1.116*	*57.461%*
STG-NCDE	Base	399.218	34.398	115.371	19.439	636.225	39.857	1424.158	51.410	550.535	36.342
	Base + CauSTG	155.076	34.398	40.488	19.678	86.426	40.041	382.163	51.707	206.044	35.063
	Base + IRM	120.276	34.556	62.373	20.153	89.024	39.937	298.365	52.901	173.373	36.534
	Base + Finetune	116.521	34.531	59.273	20.113	81.052	39.857	271.985	52.410	151.682	36.342
	Base + **Ours**	**38.991**	**33.221**	**24.039**	**18.304**	**50.116**	**38.455**	**190.779**	**49.727**	**150.144**	**30.195**
	Best Improve	*2.977*	*4.019%*	*3.799*	*10.010%*	*11.695*	*4.125%*	*6.465*	*6.383%*	*2.667*	*20.994%*

Table 4: Ablation study for the proposed SimpleST with the base MTGNN on the large-scale traffic data PeMS-Bay.

Datasets	PeMS-Bay (1 week)			PeMS-Bay (2 weeks)			PeMS-Bay (3 weeks)			PeMS-Bay (4 weeks)		
Metrics	MAE ↓	RMSE ↓	MAPE ↓	MAE ↓	RMSE ↓	MAPE ↓	MAE ↓	RMSE ↓	MAPE ↓	MAE ↓	RMSE ↓	MAPE ↓
SimpleST	1.63	3.68	3.63%	1.65	3.71	3.67%	1.68	3.90	3.79%	1.70	3.81	3.77%
w/o TCN	1.65	3.70	3.66%	1.67	3.73	3.70%	1.71	3.94	3.83%	1.74	3.92	3.85%
w/o MLP	1.69	3.72	3.69%	1.70	3.76	3.72%	1.75	3.96	3.87%	1.80	4.12	3.94%
w/o data initial	1.70	3.74	3.71%	1.72	3.78	3.75%	1.78	3.99	3.91%	1.86	4.11	3.97%

more complex backbone models like STG-NCDE. This is likely due to the substantial difference in the number of parameters between the original methods and our approach.

R#Y8nt-Q3, R#iGjG-Q3

Table 5: Overall Performance of MTGNN on Crime Prediction in terms of NYC and CHI Datasets

	New York City		Chicago	
	MAE	MAPE	MAE	MAPE
Base	0.9015	0.5476	1.0584	0.5301
Base + CauSTG	0.8785	0.5388	1.0581	0.5289
Base + IRM	0.8772	0.5392	1.0585	0.5296
Base + Finetune	0.8765	0.5387	1.0576	0.5294
Base + Ours	**0.8624**	**0.5200**	**1.0439**	**0.5193**
Best Improve	4.5339%	5.3077%	1.3890%	2.0797%

R#Y8nt-Q3, R#iGjG-Q3

Table 6: Efficiency (Minutes) and GPU Memory Cost (GB) of MTGNN on Crime Prediction in terms of NYC and CHI

	New York City		Chicago	
	Time	GPU	Time	GPU
Base	4.015	5.764	3.788	5.624
Base + CauSTG	8.112	8.232	7.998	8.678
Base + IRM	3.566	5.881	3.887	5.904
Base + Finetune	3.243	5.881	3.687	5.904
Base + Ours	**2.241**	**4.612**	**1.956**	**4.732**
Best Improve	2.620	78.491%	3.089	83.390%

5) Impact of the OOD Methods: The combination of base models with the OOD methods CauSTG and IRM exhibit varying efficiency and resource cost performance depending on the complexity of the base model. *Simpler Models (AGCRN, MTGNN)*: Combining CauSTG with these models leads to higher resource consumption and lower performance improvements, with longer training times. *More Complex Models (STG-NCDE)*: CauSTG integration with these models results in shorter training times and some performance improvement. We also explored integrating other OOD methods [24, 28] into our base models. However, these methods require edge relations to address the OOD issue, which is not compatible with backbone models.

By referring to the results presented in Table 3, it becomes evident that our SimpleST model effectively expedites the tuning process for various spatio-temporal (ST) models. This attribute holds significant value in real-world applications where time efficiency is crucial. In Table 3, we can observe the positive impact of our prompt tuning neural network on the efficiency of different baseline models. The tuning process with our SimpleST model is significantly faster than baselines, achieving a speedup of approximately 1 to 45 times. This significant reduction in tuning time validates the advantage of the simple structure employed in our model, which effectively saves time during the optimization process. These results highlight the practical benefits of our SimpleST model in real-world scenarios, where time efficiency plays a critical role in model development and deployment. By accelerating the tuning process, our model enables researchers and practitioners to iterate and experiment more quickly, leading to faster model development and improved decision-making capabilities. In summary, the findings presented in Table 3 demonstrate that our SimpleST model effectively expedites the tuning process for various ST models, resulting in significant time savings. This time efficiency attribute is of great importance in real-world applications, reaffirming the value and practicality of our approach.

4.4 Ablation Study

To assess the effectiveness of each component in our SimpleST framework, we conduct ablation experiments on both tasks. The evaluation results for traffic prediction and crime prediction can be found in Table 4, respectively. We examine the following ablated variants:

R#MauK-Q2

1) w/o TCN: This particular variant is specifically designed to encapsulate temporal dynamics, making it crucial for capturing evolving traffic patterns over time. Consequently, we have observed a decline in performance when this component is omitted, underscoring its significance in accurately modeling and predicting dynamic traffic behaviors.

2) w/o MLP: Delving deeper into the intricate components of the prompt-tuning mechanism could illuminate the unique roles each plays in enhancing the model's performance. Specifically, in this variant, we remove the multiple fully-connected layers within the prompt network. Across various scenarios, a significant decline in performance becomes evident, emphasizing the critical role of these transformation layers in effectively adjusting to the distribution shift inherent in the newly generated data.

3) w/o Skip: A more comprehensive examination of the distinct components within the prompt-tuning mechanism could offer enhanced clarity regarding their precise impacts on the model's performance. In this modified model, the skip connection in the final layer of our prompt neural network has been removed. This alteration presents challenges for the prompt network in generating suitable spatio-temporal data inputs for the pretrained backbone model $g(\cdot)$. The evaluation outcomes validate this setback, as evidenced by a notable decline in performance across various scenarios.

In addition to evaluating the effectiveness of the components in our SimpleST, Table 4 further confirms that tuning data covering a larger temporal range generally results in a larger distribution shift, leading to more pronounced performance degradation for the pretrained model. This observation reinforces the motivation behind our SimpleST, which aims to develop an effective prompt tuning approach to adapt to temporal shifts successfully. Through comparing the ablated versions with the full version of our SimpleST, the inclusion of all components in SimpleST enhances its robustness against the increase in distribution disparity.

4.5 Hyperparameter Study

In this section, we investigate the influence of different hyperparameter settings on prediction accuracy and tuning time. The evaluation is carried out on both traffic flow prediction using the PeMSD04 and PeMSD08 datasets. The results, in terms of MAE, are presented in Figure 3 and Figure 5 (shown in Appendix). Specifically, we examine the following hyperparameters:

- **Embedding dimensionality**: The optimal performance for our SimpleST model is achieved with an embedding dimensionality of 32 for both tasks. The periodic and noise-free characteristics of traffic data mitigates the risks of severe under-fitting and over-fitting, respectively. In terms of efficiency, a larger embedding dimensionality leads to a substantial increase in tuning time. A model with a dimension of 32 strikes a balance, requiring a moderate amount of tuning time.

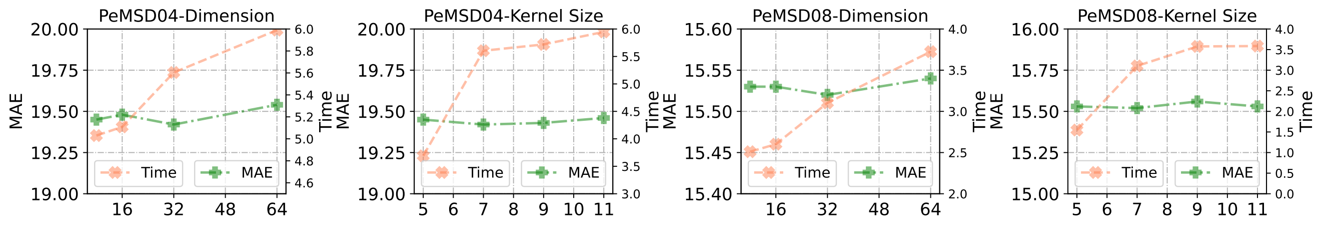


Figure 3: Hyperparameter study of SimpleST with the base MTGNN on traffic prediction

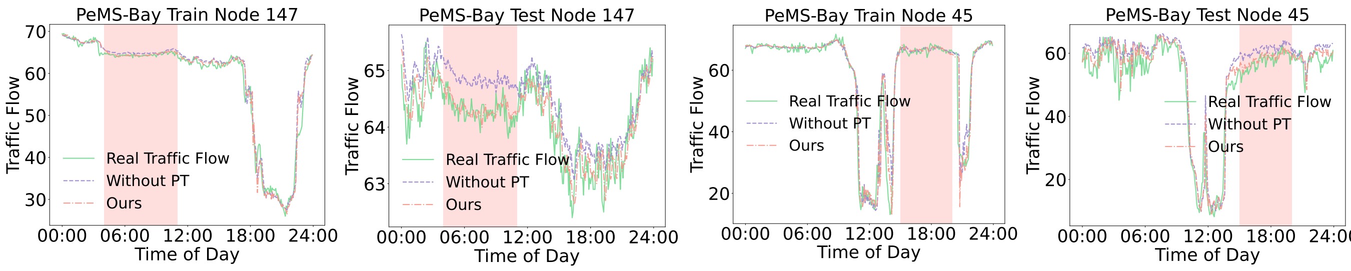


Figure 4: Case study of SimpleST with the base MTGNN on PeMS-Bay to show data distribution shift

- **Kernel size of TCN**: This parameter determines the number of consecutive time slots considered in the temporal relation modeling of our prompt network. Based on the results, a kernel size of 7 demonstrates performance advantages in certain cases. Similar to the embedding dimensionality. Additionally, a kernel size of 7 proves to be highly efficient in terms of tuning time within our SimpleST framework.
  This parameter determines the number of consecutive time slots considered in the temporal relation modeling of our prompt network. Based on the results, a kernel size of 7 demonstrates performance advantages in certain cases. Similar to the embedding dimensionality. Additionally, a kernel size of 7 proves to be highly efficient in terms of tuning time within our SimpleST framework.

4.6 Case Study

In this section, we assess the effectiveness of the proposed SimpleST framework in mitigating spatio-temporal distribution shifts by examining specific cases. Figure 4 illustrates the variation in traffic flow throughout the day for two region nodes: 147 and 45. The left plot for each region represents the training data, while the right plot represents the test data. Each plot includes three curves: the ground truth traffic flow, the predicted values obtained using the ablated model without prompt tuning (referred to as "Without PT"), and the predicted values obtained using the full version of our SimpleST. We summarize the key observations as follows:

R#qiyP-Q3, R#iGjG-Q4

In the training data, both our SimpleST and the ablated model without prompt tuning exhibit high prediction accuracy compared to the ground truth curves. However, the ground truth traffic data from the test set for the same regions demonstrates a noticeable distribution shift. Specifically, the curve in the red region for node 147 shows significant oscillations, while the curve in the red region for node 45 exhibits a distinct rising trend that differs from the training data. To be specific, Node 147 has largely different traffic patterns in train and test data. Conversely, node 45 exhibits a distinct rising trend in the test data, deviating notably from the patterns observed during training. This highlights how traffic flows differ within a given region during morning rush hours versus nighttime, as well as the contrast in traffic volumes between holidays and regular workdays. In comparison to SimpleST, predictions made by the ablated model without prompt tuning display clear inaccuracies. This observation validates the robust capability of our SimpleST framework in effectively addressing distribution shifts. By successfully handling changes in the data distribution, our model demonstrates its adaptability and ability to generalize well to different real-life urban management scenarios.

5 CONCLUSION

In this study, we introduce a simple yet powerful spatio-temporal prompt learning paradigm aimed at enhancing the robustness and generalization ability of spatio-temporal prediction models in the presence of dynamic distribution shifts. Our framework incorporates prompt tuning, which involves generating informative spatio-temporal prompt context that captures the underlying patterns and dynamics in the downstream urban data. Through comprehensive empirical evaluations across various spatio-temporal prediction tasks, we have demonstrated the remarkable effectiveness of our spatio-temporal prompt learning framework. By leveraging this framework, we significantly improve the resilience of pre-trained models to distribution shifts and enhance adaptability to new data.

REFERENCES

[1] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant Risk Minimization. *arXiv* (2019).

[2] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. 2022. Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274* (2022).

[3] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems* 33 (2020), 17804–17815.

[4] Yuzhou Chen, Ignacio Segovia-Dominguez, et al. 2021. TAMP-S2GCNets: coupling time-aware multipersistence knowledge representation with spatio-supra graph convolutional networks for time-series forecasting. In *ICLR*.

[5] Jeongwhan Choi, Hwangyong Choi, Jeehyun Hwang, and Noseong Park. 2022. Graph neural controlled differential equations for traffic forecasting. In *International Conference on Artificial Intelligence (AAAI)*, Vol. 36. 6367–6374.

[6] Zulong Diao, Xin Wang, Dafang Zhang, Yingru Liu, Kun Xie, and Shaoyao He. 2019. Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting. In *International Conference on Artificial Intelligence (AAAI)*, Vol. 33. 890–897.

[7] Ziluo Ding, Rui Zhao, Jiyuan Zhang, Tianxiao Gao, Ruiqin Xiong, Zhaofei Yu, and Tiejun Huang. 2022. Spatio-temporal recurrent networks for event-based optical flow estimation. In *International Conference on Artificial Intelligence (AAAI)*, Vol. 36. 525–533.

[8] Taoran Fang, Yunchao Zhang, Yang Yang, and Chunping Wang. 2022. Prompt tuning for graph neural networks. *arXiv preprint arXiv:2209.15240* (2022).

[9] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. [n. d.]. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. In *International Conference on Artificial Intelligence (AAAI)*. International Conference on Artificial Intelligence (AAAI), 922–929.

[10] Xu Han and Shicai Gong. 2022. LST-GCN: Long Short-Term Memory Embedded Graph Convolution Network for Traffic Flow Forecasting. *Electronics* 11, 14 (2022), 2230.

[11] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022. Visual prompt tuning. In *European Conference on Computer Vision*. Springer, 709–727.

[12] Jiawei Jiang, Chengkai Han, Wayne Xin Zhao, and Jingyuan Wang. 2023. PDFormer: Propagation Delay-aware Dynamic Long-range Transformer for Traffic Flow Prediction. *arXiv preprint arXiv:2301.07945* (2023).

[13] Guangyin Jin, Yuxuan Liang, Yuchen Fang, Jincai Huang, Junbo Zhang, and Yu Zheng. 2023. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *arXiv preprint arXiv:2303.14483* (2023).

[14] Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hongyu Yang, and Pyang Li. 2022. Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In *International Conference on Machine Learning (ICML)*. PMLR, 11906–11917.

[15] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. Stan: Spatio-temporal attention network for next location recommendation. In *WWW*. 2177–2185.

[16] Zhongjian Lv, Jiajie Xu, Kai Zheng, Hongzhi Yin, Pengpeng Zhao, and Xiaofang Zhou. 2018. Lc-rnn: A deep learning model for traffic speed prediction.. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, Vol. 2018. 27.

[17] Zheyi Pan, Yuxuan Liang, Weifeng Wang, Yong Yu, Yu Zheng, and Junbo Zhang. 2019. Urban traffic prediction from spatio-temporal data using deep meta learning. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. 1720–1730.

[18] Stephen H Schneider and Robert E Dickinson. 1974. Climate modeling. *Reviews of Geophysics* 12, 3 (1974), 447–493.

[19] Zezhi Shao, Zhao Zhang, Fei Wang, and Yongjun Xu. 2022. Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. 1567–1577.

[20] Mingchen Sun, Kaixiong Zhou, Xin He, Ying Wang, and Xin Wang. 2022. Gppt: Graph pre-training and prompt tuning to generalize graph neural networks. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. 1717–1727.

[21] Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. 2023. All in One: Multi-Task Prompting for Graph Neural Networks. In *SIGKDD*.

[22] Lijing Wang, Aniruddha Adiga, Jiangzhuo Chen, Adam Sadilek, Srinivasan Venkatramanan, and Madhav Marathe. 2022. Causalgnn: Causal-based graph neural networks for spatio-temporal epidemic forecasting. In *International Conference on Artificial Intelligence (AAAI)*, Vol. 36. 12191–12199.

[23] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. [n. d.]. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. 753–763.

[24] Yutong Xia, Yuxuan Liang, Haomin Wen, Xu Liu, Kun Wang, Zhengyang Zhou, and Roger Zimmermann. 2024. Deciphering spatio-temporal graph forecasting: A causal lens and treatment. *Advances in Neural Information Processing Systems* 36 (2024).

[25] Mingxing Xu, Wenrui Dai, Chunmiao Liu, Xing Gao, Weiyao Lin, Guo-Jun Qi, and Hongkai Xiong. 2020. Spatial-temporal transformer networks for traffic flow forecasting. *arXiv preprint arXiv:2001.02908* (2020).

[26] Yuan Yao, Bowen Dong, Ao Zhang, Zhengyan Zhang, Ruobing Xie, Zhiyuan Liu, Leyu Lin, Maosong Sun, and Jianyong Wang. 2022. Prompt tuning for discriminative pre-trained language models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

[27] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *International Joint Conferences on Artificial Intelligence (IJCAI)*. ijcai.org, 3634–3640.

[28] Zeyang Zhang, Xin Wang, Ziwei Zhang, Haoyang Li, Zhou Qin, and Wenwu Zhu. 2022. Dynamic graph neural networks under spatio-temporal distribution shift. *Advances in neural information processing systems* 35 (2022), 6074–6089.

[29] Zhengyang Zhou, Qihe Huang, Kuo Yang, Kun Wang, Xu Wang, Yudong Zhang, Yuxuan Liang, and Yang Wang. 2023. Maintaining the status quo: Capturing invariant relations for ood spatiotemporal learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3603–3614.

[30] Jiawei Zhu, Qiongjie Wang, Chao Tao, Hanhan Deng, et al. [n. d.]. AST-GCN: Attribute-augmented spatiotemporal graph convolutional network for traffic forecasting. *IEEE Access* 9 ([n. d.]), 35973–35983.

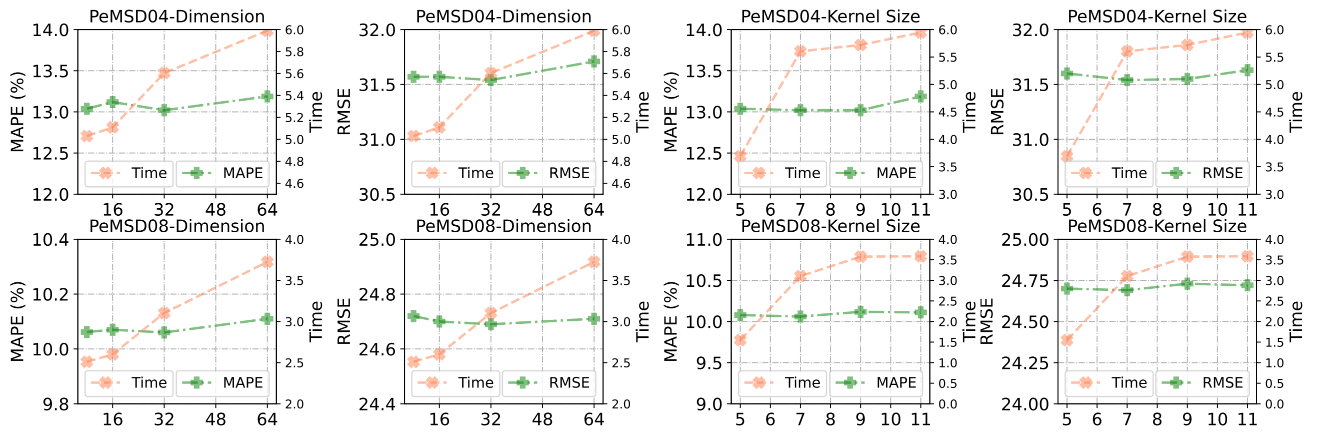


Figure 5: Hyperparameter study on traffic prediction with the base MTGNN

APPENDIX

Data Splitting for Experiments

The traffic prediction process is segmented into four distinct phases: training, finetune, prompt tuning and testing. Initially, 60% of the earliest available traffic data is designated as the training dataset. And 20% of data is used as validation data. The last 20% of data is used as the test dataset. Subsequently, the traffic data from the last two-week period of the train dataset is utilized for finetuning and prompt tuning for multiple tables and figures referenced in the study, apart from Table 7 and Table 4. Moreover, for Table 7 and Table 4, an additional division is implemented. In this case, data from the last 1 day and 1 week of train datasets are allocated for finetune and prompt tuning to test the generalization ability of our method SimpleST on different time segments of urban spatio-temporal prediction.

Table 7: Tuning time (minutes) of MTGNN comparison with different amount of tuning data for traffic prediction.

Datasets	PeMSD04			PeMSD07			PeMSD03			PeMSD08		
Time line	1 day ↓	1 week ↓	2 weeks ↓	1 day ↓	1 week ↓	2 weeks ↓	1 day ↓	1 week ↓	2 weeks ↓	1 day ↓	1 week ↓	2 weeks ↓
Time for Training Scratch	2.382	13.913	20.401	9.093	30.905	59.940	10.560	22.130	27.716	1.071	3.477	7.152
Time for Finetune	2.848	14.620	23.036	8.002	40.865	20.541	10.227	20.754	16.381	2.025	2.945	8.432
Time for Prompt Tuning	**1.302**	**1.952**	**5.604**	**5.694**	**12.220**	**13.959**	**3.030**	**3.071**	**10.782**	**0.489**	**1.371**	**3.103**
Faster x than Scratch	82.950%	612.756%	264.044%	59.694%	152.905%	329.400%	248.515%	620.612%	157.058%	119.018%	153.611%	130.487%
Faster x than Prompt	118.740%	648.975%	311.420%	40.885%	234.411%	47.152%	237.525%	575.806%	51.929%	314.110%	114.807%	171.737%

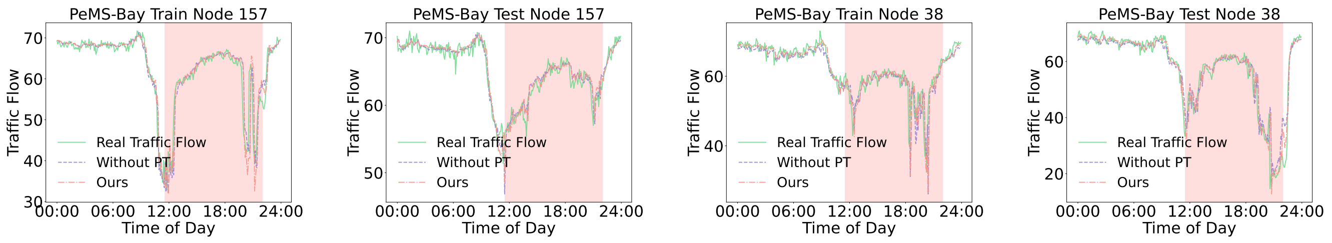
More Cases of Data Shiftness

In the training phase, both our SimpleST and the modified model lacking prompt tuning demonstrate strong predictive accuracy when compared to the actual data patterns. However, a significant shift in data distribution is noticeable in the ground truth traffic data from the test set for the same regions. Notably, node 157 displays pronounced fluctuations in the red region, while node 38 in the same area exhibits a distinct upward trend diverging from the training data. Specifically, Node 157 showcases markedly different traffic patterns between the training and test datasets. Conversely, in the test data, node 38 displays a clear upward trajectory, deviating notably from the training patterns. This illustrates how traffic patterns vary within the same region during peak morning hours versus nighttime, as well as the differences in traffic volume between holidays and regular workdays. Compared to SimpleST, the modified model without prompt tuning produces notably inaccurate predictions. This underscores the robustness of our SimpleST framework in effectively mitigating distribution shifts, showcasing its adaptability and generalization capacity across various real-world urban management contexts.

R#qiyP-Q3, R#iGjG-Q4

Hyperparameter Settings

ASTGNN [9]: We experiment with different numbers of terms in the Chebyshev polynomial, denoted as $K = 3$ and set the kernel size along the temporal dimension to 3. All graph convolution layers utilize 64 convolution kernels, while the temporal convolution layers also employ 64 convolution kernels. The batch size is set to 64, and the learning rate to 0.0001.
AGCRN [3]: It employs a dual-tier AGCRN structure to comprehend both the node-centric spatial intricacies and temporal evolutions. In terms of configuration, we designate 32 units for the hidden layers across all AGCRN units alongside a consistent batch size of 64. The learning rate is set as 0.003.



R#qiyP-Q3, R#iGjG-Q4

Figure 6: Case study of SimpleST with the base MTGNN on PeMS-Bay to show data distribution shift

MTGNN [23]: For fair comparison, all compared algorithms have hidden dimensionality modified from the range [8,16,32,64] to achieve their best performance as reported results at 32. The learning rate η is initialized as 0.003 with weight decay 0.3. For GNN-based models, the number of GCN layer is 3.

STG-NCDE [5]: Following standard practice in this domain [14], we employ a 12-sequence-to-12-sequence forecasting setting. This means it predicts the next 12 graph snapshots after observing the preceding 12 snapshots.

CauSTG [29]: The learning rate is $1e-3$. And the number of TCN kernels are 4. The dimensions of TCN kernels are 12, 6 and 3. Hidden dimensions of GNN are 64. And optimizer is Adam.

IRM [1]: The setting of the scale is 1. And it is combined with the backbone loss with the weight 1.

R#MauK-Q1

SimpleST: For our prompt tuning network, the number of the TCN Layer is 2 and the number of MLP layer is set as 2. The learning rate is set as $1e-3$ and batch size is set as 32. The kernel size of the TCN Layer is set as 7 during which our framework SimpleST obtains the best performance from the range of [5,7,9,11]. Following existing settings of traffic prediction, we utilize historical 12 time steps with 5 minutes a step to predict future 12 time steps on point-based datasets (PeMSD04, PeMSD08, PeMSD03, PeMSD07 and PeMS-Bay).

Stability of Long-Term Prediction

R#MauK-Q4

Our method, despite the focus on short-term gains, inherently possesses advantageous characteristics that contribute to its long-term stability and effectiveness when applied to continuously evolving data over extended durations. (1) Our method's flexibility and adaptability enable it to adjust to changing data patterns over time, enhancing its capability to maintain stable predictions across extended periods. (2) The robust design of our framework, including the incorporation of advanced learning mechanisms and adaptive features, fortifies its stability when faced with evolving data dynamics. (3) The ability of our method to generalize well to diverse datasets and scenarios further bolsters its long-term stability by ensuring consistent performance across varying conditions.