



Prompt-Based Spatio-Temporal Graph Transfer Learning

Junfeng Hu*
National University of Singapore
Singapore
junfengh@u.nus.edu

Xu Liu
National University of Singapore
Singapore
liuxu@comp.nus.edu.sg

Zhencheng Fan
University of Technology Sydney
Sydney, NSW, Australia
zhencheng.fan@student.uts.edu.au

Yifang Yin
Institute for Infocomm Research,
A*STAR
Singapore
yin_yifang@i2r.a-star.edu.sg

Shili Xiang
Institute for Infocomm Research,
A*STAR
Singapore
sxiang@i2r.a-star.edu.sg

Savitha Ramasamy
Institute for Infocomm Research,
A*STAR & IPAL, CNRS@CREATE
Singapore
ramasamysa@i2r.a-star.edu.sg

Roger Zimmermann
National University of Singapore
Singapore
rogerz@comp.nus.edu.sg

Abstract

Spatio-temporal graph neural networks have proven efficacy in capturing complex dependencies for urban computing tasks such as forecasting and kriging. Yet, their performance is constrained by the reliance on extensive data for training on a specific task, thereby limiting their adaptability to new urban domains with varied task demands. Although transfer learning has been proposed to remedy this problem by leveraging knowledge across domains, the cross-task generalization still remains under-explored in spatio-temporal graph transfer learning due to the lack of a unified framework. To bridge the gap, we propose Spatio-Temporal Graph Prompting (STGP), a prompt-based framework capable of adapting to multi-diverse tasks in a data-scarce domain. Specifically, we first unify different tasks into a single template and introduce a task-agnostic network architecture that aligns with this template. This approach enables capturing dependencies shared across tasks. Furthermore, we employ learnable prompts to achieve domain and task transfer in a two-stage prompting pipeline, facilitating the prompts to effectively capture domain knowledge and task-specific properties. Our extensive experiments demonstrate that STGP outperforms state-of-the-art baselines in three tasks—forecasting, kriging, and extrapolation—achieving an improvement of up to 10.7%.

CCS Concepts

• **Information systems** → *Spatial-temporal systems*.

Keywords

Transfer learning; prompt tuning; spatio-temporal data mining

*Work done when the author is a research intern at Institute for Infocomm Research, A*STAR, Singapore.



This work is licensed under a Creative Commons Attribution International 4.0 License.

CIKM '24, October 21–25, 2024, Boise, ID, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0436-9/24/10
<https://doi.org/10.1145/3627673.3679554>

ACM Reference Format:

Junfeng Hu, Xu Liu, Zhencheng Fan, Yifang Yin, Shili Xiang, Savitha Ramasamy, and Roger Zimmermann. 2024. Prompt-Based Spatio-Temporal Graph Transfer Learning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*, October 21–25, 2024, Boise, ID, USA. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3627673.3679554>

1 Introduction

The rise of urban computing has spurred the development of sensor technologies for monitoring urban conditions. The collected data, often represented as spatio-temporal graphs, has significantly changed our perception of understanding city environments, leading to various applications, such as forecasting [23] and kriging [2]. Recently, spatio-temporal graph neural networks (STGNNs) have stood as a dominant paradigm in urban computing, owing to their ability to capture complex spatio-temporal dependencies [19, 31, 38–40, 44]. The success of STGNNs primarily depends on their training within a singular task and dataset, which necessitates the availability of extensive data for effective optimization. However, this reliance on abundant data constrains their generalizability when a new urban domain with a multi-task requirement emerges [12].

Transfer learning, which focuses on transferring knowledge across *domains* and *tasks*, emerges as a promising machine learning approach in scenarios where training data is limited [47]. It aims to leverage knowledge learned from a data-rich source domain to perform diverse downstream tasks in a data-scarce target domain. Nowadays, researchers have realized the data scarcity issue in spatio-temporal graphs and proposed a variety of solutions [35, 42]. As shown in Fig. 1(a)-(b), these models are first pre-trained on a source domain and then fine-tuned on a target domain. To achieve effective domain transfer, they are further supported by a transfer module. For instance, CrossTReS [14] captures domain shift by a weighting mechanism that filters out patterns exclusively in the source domain. TPB [25] stores shareable patterns across the domains by a memory bank. Despite the intriguing performance obtained by these systems, they are specifically designed for the task

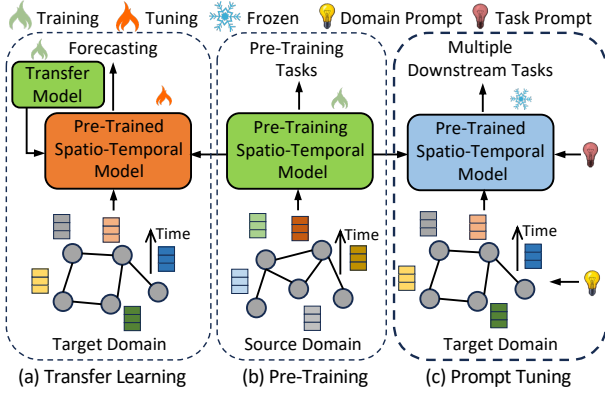


Figure 1: Comparison of existing transfer learning approaches with our prompt tuning. (a) The pre-trained model is adapted to a target domain, aided by a domain transfer model. (b) The spatio-temporal model is pre-trained on a data-rich source domain. (c) STGP leverages prompts to modify data, achieving domain and task transfer, respectively.

of forecasting by incorporating task-specific architectures. Consequently, they fall short in achieving task transfer due to their sensitivity to changes in task specifications [3]. In this work, we aim to move towards general task-agnostic frameworks that are capable of performing multiple tasks in the target domain.

To tackle the problem of task transfer, an emerging technique known as prompt tuning has gained prominence [3]. This concept, originating from Natural Language Processing (NLP), first unifies different tasks into one task template with a single training objective. Then, it leverages pre-trained Transformers that are task-agnostic and employs prompts to instruct task-specific knowledge. Compared to fine-tuning, prompt tuning gains more efficiency by freezing the pre-trained model. Meanwhile, it still yields performance that is comparable to, or even surpasses the fine-tuning methods [21]. Recently, the community has found that prompt tuning can also perform domain transfer by incorporating domain-specific prompts into the input data [6]. Inspired by this paradigm, we argue that prompting can similarly be introduced for transfer learning in spatio-temporal graphs, primarily for two key reasons. First, prompting can suit different spatio-temporal tasks flexibly by leveraging a unified task template and task-agnostic architectures. Second, it provides an innovative idea to achieve both domain and task transfer efficiently from a data perspective: keeping the model frozen while adapting the data space of the target domain.

However, applying prompt tuning to spatio-temporal graphs is non-trivial due to the following challenges. Firstly, to enable task transfer, the design of the model and the knowledge captured by it through pre-training should be task-agnostic. Taking language models [27] as an example, they unify the language tasks into a template of word prediction and utilize Transformers as the learning model. The template and the network architecture ensure its applicability and the learning of latent space shared by various tasks [30]. Nevertheless, *how do we design a model and a task template for spatio-temporal graphs that preserves shareable dependencies among tasks* is under-explored. Secondly, existing prompting methods treat domain transfer and task transfer as distinct problems, introducing

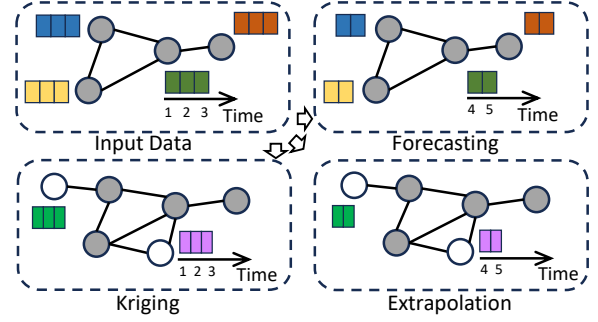


Figure 2: Spatio-temporal graph downstream tasks.

different solutions to address them by using either natural language or learnable prompts [6, 33]. However, spatio-temporal graphs lack an effective way to incorporate language that can prompt domain and task knowledge. Moreover, it is less apparent what mechanism the learnable prompts should take to guide both transfer objectives. Thus, *how to learn prompts that capture both domain knowledge and task-specific properties* is also a challenge waiting to be solved.

To address the aforementioned challenges, we propose a novel Spatio-Temporal Graph Prompting framework, termed STGP, with the goal of facilitating domain transfer while enabling capabilities for multiple downstream tasks. First, we unify spatio-temporal graph tasks into a single task template, inspired by STEP [32], and introduce a task-agnostic network that aligns with this template. Moreover, we introduce a two-stage prompting approach. As depicted in Fig. 1(c), the prompting method sequentially learns domain transfer and task transfer in its respective prompting stages.

In particular, to address the first challenge, we unify spatio-temporal graph tasks into a data reconstruction template, where each task is represented as predictions of data at specific masked spatio-temporal positions. To align with the template, we introduce a task-agnostic network, empowered by a Transformer-based encoder and a gated spatio-temporal decoder, that takes in input data to predict masked signals. In this work, we adopt predictions of randomly masked data as the model’s pre-training task to facilitate the learning of shareable spatio-temporal dependencies. Then, as illustrated in Fig. 2, we apply the pre-trained model to two common downstream tasks—forecasting and kriging—and introduce a new task—extrapolation. The extrapolation task aims at predicting future signals for nodes lacking historical data, which holds significant practical importance. For example, it enables the use of limited sensors to forecast future traffic conditions across an entire city.

Regarding the second challenge, we decouple the transfer of domain and task knowledge and propose a prompting pipeline consisting of two stages. The initial stage focuses on domain transfer, adapting the target domain’s data space to the source domain used for model pre-training. In practice, we freeze the pre-trained model, forward the domain-prompted data into the encoder, and train the prompts using the pre-training task. In the second stage, domain prompts are fixed and task prompts operate on the representations produced by the encoder. The task-prompted representations are fed into the decoder, enabling it to differentiate downstream tasks and reach optimal performance for each task. Our model is more efficient than current spatio-temporal transfer learning methods by

leveraging domain prompts to transfer knowledge without the need for fine-tuning the pre-trained model. Meanwhile, by employing task prompting, our STGP shows versatility across multiple downstream tasks, yielding superior performance compared to other baselines. To summarize, our contributions lie in three aspects:

- We introduce STGP, a novel framework for spatio-temporal graph transfer learning that leverages a unified template and a task-agnostic network to transfer knowledge across multiple tasks in the target domain, such as forecasting, kriging, and extrapolation.
- We propose an innovative two-stage prompting strategy, empowering STGP to capture domain knowledge and task-specific properties for effective domain and task transfer.
- We conduct extensive experiments across four public datasets. The results and discussions highlight the superior performance of STGP and the efficacy of each component of the model.

2 Related Work

2.1 Spatio-Temporal Graph Neural Networks

STGNNs have become one of the dominant paradigms for urban computing [12], leading to a series of applications, such as forecasting and kriging. DCRNN [19] and STGCN [44] are two pioneer methods for forecasting, which employ GNNs to capture spatial relations and deploy sequential models to model temporal dependencies. Graph WaveNet [40] proposes a matrix learning module to model hidden structural information, which becomes an influential idea for following methods [18, 31, 32]. In terms of the task of kriging, a growing number of models have been recently developed to address it. For instance, KCN [2] applies GNNs to capture dependencies between existing and unknown nodes, while IGNNK [38] further considers temporal relations. DualSTN [10] learns joint correlations between time and space. SATCN [39] and INCREASE [46] propose to capture statistical graph information. However, STGNNs heavily rely on sufficient training data and are designed exclusively for a specific task. In this study, we are devoted to addressing these challenges through the proposed prompt-based framework.

2.2 Spatio-Temporal Transfer learning

To improve the performance of STGNNs in scenarios with data scarcity, transfer learning methods are employed for spatio-temporal data. The model undergoes initial pre-training on a data-rich source domain and is then transferred to a target domain where data is scarce. Additionally, a transfer module is introduced to facilitate domain adaptation. RegionTrans [35], MetaST [29], and CrossTREs [14] focus on grid data, learning region correlations between target and source cities. In the graph setting, ST-GFSL [26] introduces to generate network parameters specialized for the target domain. TPB [25] learns temporal patterns shareable across the domains. TransGTR [15] proposes to generate graph structures for the target domain that exhibit structural distributions akin to the source domain. Although these approaches introduce various transfer mechanisms to enhance model performance, most of them are optimized through a complex meta-learning framework such as Reptile [28] or MAML [5] that require fine-tuning of the pre-trained model, which significantly diminishes their efficiency. Moreover, their exclusive focus on forecasting, using task-specific architectures, results in a lack of versatility for multiple downstream tasks.

2.3 Prompt Tuning

Originating from Natural Language Processing, prompt tuning methods have gained widespread use for adapting pre-trained language models to a range of downstream tasks [21]. These methods formulate the pre-training and downstream tasks into a single template of word prediction and employ learnable or natural language prompts to instruct task-related properties. In recent times, researchers have discovered that prompting shows potential in domain transfer by guiding the model with domain knowledge [6]. Aside from the language tasks, prompting methods have also gained attention from various research areas such as Computer Vision [11], Graph Learning [33], and Time Series Modeling [22, 41]. Despite their success, prompt-based methods remain under-explored in the context of spatio-temporal learning, with few proposed methods for graph [20] and grid data [45] that focus on domain transfer only. Nevertheless, the transfer learning scenario demands simultaneous management of domain transfer and multi-task capabilities, thereby amplifying the complexity of the challenge.

3 Preliminaries

We outline the notations for spatio-temporal graphs, followed by formal definitions of transfer learning and the three specific downstream tasks we aim to address.

3.1 Data Notations

Definition 1 (Graph) Sensors can be organized as a graph represented by $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$, where \mathcal{V} is the set of nodes, \mathcal{E} is the set of edges, and $A \in [0, 1]^{|\mathcal{V}| \times |\mathcal{V}|}$ is the normalized adjacency matrix describing the weights between nodes.

Definition 2 (Spatio-Temporal Data) Signals gathered by sensors in the graph are represented as $X \in \mathbb{R}^{N \times T \times d_x}$, where N indicates the number of nodes collecting data, T represents a time window, and d_x denotes the attribute count. We denote the data captured by node i at time t as $x_i^t \in \mathbb{R}^{d_x}$.

3.2 Problem Definitions

Definition 3 (Spatio-Temporal Graph Transfer Learning) Consider a model pre-trained on P source domains with data over an substantial period T_S (e.g., one year), denoted as $\mathcal{G}^S = \{\mathcal{G}_1^S, \dots, \mathcal{G}_P^S\}$. When a new target domain \mathcal{G}^T with $T_T \ll T_S$ (e.g., three days) emerges, the model is required to adapt to this new domain by utilizing knowledge acquired from the source domains and to perform various spatio-temporal graph learning tasks.

Problem 1 (Forecasting) Given historical signals of T time steps of N nodes from a target domain, forecasting aims to transfer the pre-trained model $f(\cdot)$ to predict the data of subsequent T' steps:

$$X^{1:T} \xrightarrow{f(\cdot)} X^{T+1:T+T'}, \quad (1)$$

where $X^{T+1:T+T'} \in \mathbb{R}^{N \times T' \times d_x}$ is the future readings.

Problem 2 (Kriging) The goal of kriging is to adapt the model $f(\cdot)$ to predict data of M unobserved locations given the N observed nodes at the same time window T :

$$X_{1:N} \xrightarrow{f(\cdot)} X_{N+1:N+M}, \quad (2)$$

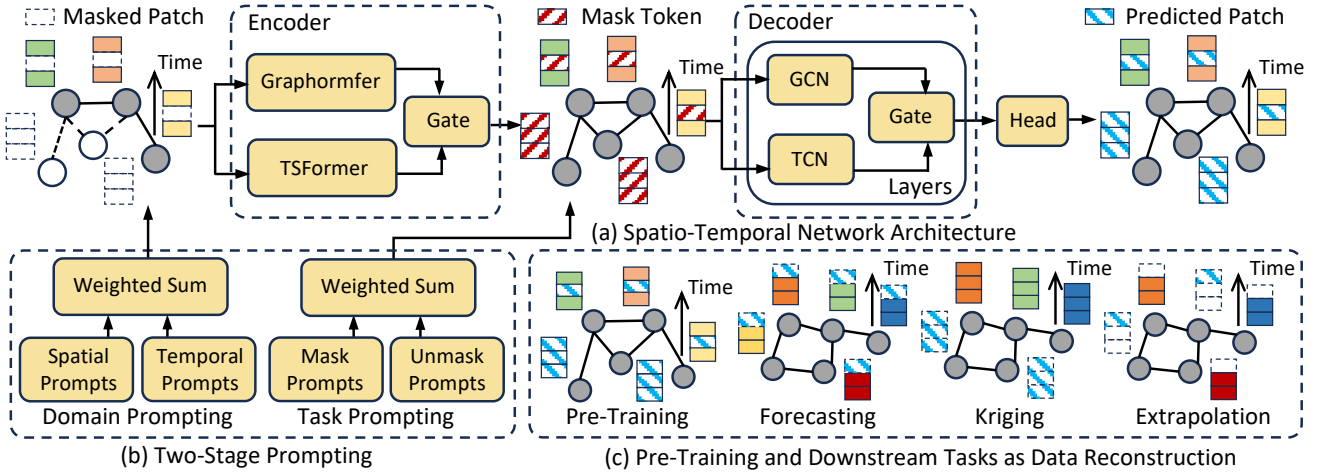


Figure 3: The overview of our Spatio-Temporal Graph Prompting framework. (a) The network architecture is task-agnostic and aligns with the unified template. (b) Domain and task prompts are learned sequentially through the two-stage prompting pipeline. (c) The tasks are consolidated within a data reconstruction template, where they are represented by masked locations.

where $X_{N+1:N+M} \in \mathbb{R}^{M \times T \times d_x}$ is the data of unobserved locations.

Problem 3 (Extrapolation) Given historical signals from N observed nodes, extrapolation involves using the model $f(\cdot)$ to predict T' future steps of M unobserved locations:

$$X_{1:N}^{1:T} \xrightarrow{f(\cdot)} X_{N+1:N+M}^{T+1:T+T'} \quad (3)$$

where $X_{N+1:N+M}^{T+1:T+T'} \in \mathbb{R}^{M \times T' \times d_x}$ denotes the future data of unobserved locations.

4 Methodology

In this paper, we aim to introduce a spatio-temporal graph transfer learning framework, empowered by prompt tuning, that can be applied to multiple tasks. To achieve this goal, we first consolidate the pre-training and downstream tasks into a unified template as illustrated in Fig. 3(c). Next, we design a network architecture that aligns with the template (Fig. 3(a)) and propose a two-stage prompting pipeline for domain and task transfer (Fig. 3(b)), respectively. In the following sections, we detail each component of STGP.

4.1 Unification Spatio-Temporal Graph Tasks

The effectiveness of applying a pre-trained model to diverse downstream tasks greatly relies on a unified task template, which ensures the learning of shareable latent space across tasks. However, it remains unclear how to unify these tasks for spatio-temporal graphs. As suggested by [13], different tasks place differing emphases on spatial and temporal dimensions. For instance, forecasting predominantly relies on the temporal evolution of historical signals, while kriging focuses more on the spatial relationships between observed and unobserved nodes. Consequently, existing models have to adopt task-specific training objectives and architectures to align with these properties [26, 39], which constrains their applicability on multiple downstream tasks.

Inspired by STEP [32] and GraphMAE [9], we reformulate the pre-training and downstream tasks as the reconstruction of masked

data. As illustrated in Fig. 3(c), this unified template involves predicting masked data that requires the extraction of useful spatio-temporal dependencies from unmasked signals. In this way, these tasks can be defined when they are characterized by masks placed at specific positions. For example, forecasting predicts masked signals at future time steps, while kriging aims to infer masked data that represents unobserved nodes. We adopt the predictions of randomly masked data as the pre-training task, which will be detailed below. Since the data masking relies solely on the task definition, without prioritizing spatial or temporal factors, our unified template ensures equal emphasis on dependencies across both dimensions, leading to consistent performance across all tasks.

4.2 Pre-Training and Spatio-Temporal Networks

Our pre-training approach involves predicting randomly masked data, propelling the model to identify shareable spatio-temporal dependencies for downstream tasks. To facilitate this, we introduce a spatio-temporal graph network featuring an autoencoder that comprises a Transformer-based encoder to capture complex dependencies within unmasked data and a gated decoder focusing on the predictions of masked data.

4.2.1 Patching and Masking. As individual node signals contain low information density, reconstructing each signal separately could lead to the learning of trivial solutions [32]. This issue can be more acute in the transfer setting when a target domain involves limited data. Therefore, we employ data patching [25] to aggregate adjacent data into patches. Concretely, given sequential data X_i from node i , we divide it into non-overlapping patches denoted by $\hat{X}_i \in \mathbb{R}^{T_p \times L d_x}$, where T_p is the number of patches and L represents patch size. Then, we employ a linear projection to embed each patch into feature space, represented as $S_i \in \mathbb{R}^{T_p \times d_h}$. Note that we will use T to denote T_p for simplicity in the following sections.

Next, we pre-train the model on source domains by randomly masking the patches. Contrasting with the graph or time series models that utilize a noisy masking method [9, 22], spatio-temporal

random masking, which affects the two dimensions simultaneously, presents more complexity. As the Transformers only take in unmasked patches without padding [32], it is crucial to maintain a uniform patch number within each dimension to ensure a valid tensor structure, a requirement that noisy masking does not satisfy. Therefore, we divide the masking into spatial and temporal parts, where patches of a node or at a time step are all masked. Specifically, given a set of N nodes and of T steps, we randomly sample two sub-sets from them, denoted as \mathcal{M}_s and \mathcal{M}_t . Then, a patch s_i^t with its indexes in any of the sub-sets is marked as masked. The remaining unmasked patches, denoted by $S_{\bar{\mathcal{M}}} \in \mathbb{R}^{N_u \times T_u \times d_h}$ with $S_{\bar{\mathcal{M}}} = \{s_i^t | i \notin \mathcal{M}_s \text{ and } t \notin \mathcal{M}_t\}$, $N_u = N - |\mathcal{M}_s|$ and $T_u = T - |\mathcal{M}_t|$, are the inputs of the encoder. We set a total masking ratio of 75%, following [8, 32]. The scarcity of unmasked patches in the model aids in better capturing general spatio-temporal correlations.

4.2.2 Transformer-Based Encoder. Transformers have become dominant in pre-training frameworks due to their attractive learning capabilities [16, 17, 36]. Additionally, masking further enhances their training efficiency on large-scale datasets by only feeding in unmasked patches. Nevertheless, employing a single Transformer to model both spatial and temporal dimensions still results in considerable time overhead, due to its complexity of $O(N_u^2 \times T_u^2)$. To overcome this issue, we model the two dimensions respectively by utilizing TSFormer [32] to model temporal correlations and Graphormer [43] to capture spatial dependencies. Note that both models incorporate positional embeddings into their inputs. Following [15], we enhance TSFormer by adding time of day (TOD) and day of week (DOW) embeddings, bolstering its capacity to capture periodic knowledge in the data-scarce target domain. In the bottom layer, representations learned from the two models are fused by a gating mechanism formulated as:

$$\begin{aligned} H_{\bar{\mathcal{M}}} &= Z \odot \text{Enc}_s(S_{\bar{\mathcal{M}}}) + (1 - Z) \odot \text{Enc}_t(S_{\bar{\mathcal{M}}}), \\ Z &= \sigma(\text{Enc}_s(S_{\bar{\mathcal{M}}})W_1 + \text{Enc}_t(S_{\bar{\mathcal{M}}})W_2 + b), \end{aligned} \quad (4)$$

where $\text{Enc}_s(\cdot)$, $\text{Enc}_t(\cdot)$ are the two Transformers, $\sigma(\cdot)$, \odot denote sigmoid and element-wise product, W_1 , W_2 , b are learnable parameters. In this way, the time complexity of our encoder reduces to $O(N_u^2 + T_u^2)$, thereby achieving greater efficiency.

4.2.3 Gated Spatio-Temporal Decoder. The decoder first recovers full patches $H \in \mathbb{R}^{N \times T \times d_h}$ from $H_{\bar{\mathcal{M}}}$, where masked positions are replaced by a shared learnable mask token $h_{\mathcal{M}}$:

$$h_i^t = \begin{cases} h_{\mathcal{M}}^t & i \notin \mathcal{M}_s \text{ and } t \notin \mathcal{M}_t \\ h_{\mathcal{M}} & i \in \mathcal{M}_s \text{ or } t \in \mathcal{M}_t \end{cases}. \quad (5)$$

Then, the decoder operates on H to predict signals at masked positions. Here, the decoder architecture should be task-agnostic, ensuring its adaptability across all tasks. Unfortunately, we cannot simply employ forecasting or kriging STGNNs as in other transfer learning frameworks [15, 25, 26]. This limitation arises because most STGNNs prioritize the learning of either spatial or temporal dependencies, which can compromise their effectiveness on other tasks, as suggested by our experiments. Thus, we propose a gated spatio-temporal graph decoder to circumvent the issue. Each decoder layer uses a graph convolutional network (GCN) and a temporal convolutional network (TCN) for parallel spatial and

temporal modeling, followed by a gating mechanism for feature fusion. The process can be described by:

$$\begin{aligned} H^l &= Z \odot \text{TCN}(H^{l-1}) + (1 - Z) \odot \text{GCN}(H^{l-1}, A), \\ Z &= \sigma(\text{TCN}(H^{l-1})W_1 + \text{GCN}(H^{l-1}, A)W_2 + b), \end{aligned} \quad (6)$$

where l represents the layer index. In this way, the gating mechanism dynamically balances spatial and temporal patterns, facilitating the model on various tasks with different characteristics. Note that we still add the time embeddings at its first layer. Finally, we employ a linear prediction head to obtain patch predictions. We adopt mean absolute error (MAE) to optimize the model and calculate it solely on masked patches, in accordance with [7, 32].

4.3 Two-Stage Prompting

In our transfer learning scenario, a pre-trained model needs to adapt to multiple downstream tasks in a target domain. Current prompting methods often introduce distinct solutions for domain transfer and task transfer [6, 33]. In contrast, we aim to achieve both goals within our single framework. Unfortunately, trivially optimizing a single bunch of prompts is inadequate for accomplishing the two goals. This is because domain transfer is designed to learn general domain knowledge, whereas task transfer focuses on capturing task-specific characteristics. Motivated by this observation, we introduce an innovative two-stage prompting pipeline that achieves these goals through a sequential, two-step approach.

4.3.1 Prompt Bank. We use a prompt bank as a basic prompting unit, where each bank contains a set of prompt vectors, defined as $P = (p_1, \dots, p_{N_p}) \in \mathbb{R}^{N_p \times d_h}$. The prompted vector for a given patch h_i^t can be learned by a weighted sum:

$$\text{PT}(h_i^t) = h_i^t + \sum_{j=1}^{N_p} \alpha_j p_j, \quad \alpha_j = \begin{cases} \sigma(h_i^t p_j^\top) & \sigma(h_i^t p_j^\top) > \varphi \\ 0 & \sigma(h_i^t p_j^\top) \leq \varphi \end{cases}, \quad (7)$$

where φ is a threshold and $\text{PT}(\cdot)$ is the prompting function. Compared to methods such as single embedding prompting [24], the prompt bank provides better abilities to transform patches in the latent space, which generally yields better performance [33].

4.3.2 Domain Prompting. In the first stage, prompts are designed to capture the correlations between the domains and dynamically adapt the target domain data. To facilitate the capturing of domain knowledge and general spatio-temporal dependencies, we optimize the prompts by the same pre-training task with a large masking ratio and take the prompted patches as the encoder's inputs. We utilize different prompt banks for Graphormer and TSFormer, enabling them to transfer spatial and temporal domains, respectively. Note that positional embeddings are added to patches before prompting as prior knowledge. The process is defined as:

$$S_s^* = \text{PT}_s(S_{\bar{\mathcal{M}}} + PE_e), \quad S_t^* = \text{PT}_t(S_{\bar{\mathcal{M}}} + PE_e), \quad (8)$$

where S_s^* , S_t^* are the prompted inputs of the two Transformers, $\text{PT}_s(\cdot)$, $\text{PT}_t(\cdot)$ are prompting functions, and PE_e are positional embeddings of the encoder. In this stage, only the prompts are trainable while the other modules are frozen without fine-tuning, which increases efficiency over other transfer learning methods.

4.3.3 Task Prompting. The task prompts are learned respectively for each downstream task to prompt out task-specific properties. We incorporate the prompts into the hidden representations learned by the encoder. Then, the decoder takes in the task-prompted representations to predict target signals at masked positions based on the learned target domain dependencies. We utilized separate prompt banks for mask tokens and unmasked patches, thereby facilitating the capturing of relationships between them:

$$H_{\mathcal{M}}^* = \text{PT}_m(H_{\mathcal{M}} + PE_d), \quad H_{\bar{\mathcal{M}}}^* = \text{PT}_u(H_{\bar{\mathcal{M}}} + PE_d), \quad (9)$$

where $\text{PT}_m(\cdot)$ and $\text{PT}_u(\cdot)$ are promptings for mask tokens and unmasked patches, PE_d is the embeddings of the decoder. Note that in this stage, we further freeze the domain prompts. As downstream tasks feature different masked position patterns, task prompting can benefit the model by identifying these distinctions, thereby achieving optimal performance for each task.

4.4 STGP Learning Procedure

We present the overall learning algorithm of STGP in Algo. 1. Given the encoder f_{θ_e} , the decoder f_{θ_d} , and the head f_{ϕ} parameterized by θ_e , θ_d , and ϕ , we first train them through predictions of randomly masked patches. Then, keeping them frozen, the domain prompts π_{dm} are optimized using the pre-training task for domain transfer. Finally, further with π_{dm} held fixed, task prompts π_{tk} are trained for a task. The head can optionally be tuned at this stage.

Algorithm 1 STGP Pre-Training and Prompting

Input: Source and target domain data $\mathcal{G}^S, \mathcal{G}^T$. Encoder f_{θ_e} , decoder f_{θ_d} , head f_{ϕ} . Domain and task prompts π_{dm} and π_{tk} .

Output: Optimal parameters $\theta_e, \theta_d, \phi, \pi_{dm}$ and π_{tk} .

```

1: // Pre-training  $f_{\theta_e}, f_{\theta_d}$  and  $f_{\phi}$  with  $\mathcal{G}^S$ 
2: while not converged do
3:    $S_{\bar{\mathcal{M}}}, S_{\mathcal{M}} \leftarrow \text{PatchingMasking}(\mathcal{G}^S)$ 
4:    $\hat{S}_{\mathcal{M}} = f_{\phi, \theta_e, \theta_d}(S_{\bar{\mathcal{M}}})$   $\triangleright$  Data reconstruction, Eq. 4 - 6
5:   Calculate loss  $\text{MAE}(\hat{S}_{\mathcal{M}}, S_{\mathcal{M}})$  and optimize  $\theta_e, \theta_d$ , and  $\phi$ .
6: // Domain prompting using  $\mathcal{G}^T$ 
7:  $f_{\theta_e}, f_{\theta_d}, f_{\phi} \leftarrow \text{Freeze}(\theta_e, \theta_d, \phi)$ 
8: while not converged do
9:    $S_{\bar{\mathcal{M}}}, S_{\mathcal{M}} \leftarrow \text{PatchingMasking}(\mathcal{G}^T)$ 
10:   $S_{\bar{\mathcal{M}}}^* \leftarrow \text{DomainPrompting}(S_{\bar{\mathcal{M}}}, \pi_{dm})$   $\triangleright$  Eq. 8
11:   $\hat{S}_{\mathcal{M}} = f_{\phi, \theta_e, \theta_d}(S_{\bar{\mathcal{M}}}^*)$   $\triangleright$  Eq. 4 - 6
12:  Calculate loss and optimize  $\pi_{dm}$ .
13: // Task prompting
14:  $f_{\theta_e}, f_{\theta_d}, f_{\phi}, \pi_{dm} \leftarrow \text{Freeze}(\theta_e, \theta_d, \phi, \pi_{dm})$ 
15: while not converged do
16:   $S_{\bar{\mathcal{M}}}, S_{\mathcal{M}} \leftarrow \text{PatchingMasking}(\mathcal{G}^T)$   $\triangleright$  Downstream task masking
17:   $S_{\bar{\mathcal{M}}}^* \leftarrow \text{DomainPrompting}(S_{\bar{\mathcal{M}}}, \pi_{dm})$   $\triangleright$  Eq. 8
18:   $H_{\bar{\mathcal{M}}} = f_{\theta_e}(S_{\bar{\mathcal{M}}}^*)$   $\triangleright$  Eq. 4
19:   $H \leftarrow \text{RecoverFullPatches}(H_{\bar{\mathcal{M}}})$   $\triangleright$  Eq. 5
20:   $H^* \leftarrow \text{TaskPrompting}(H, \pi_{tk})$   $\triangleright$  Eq. 9
21:   $\hat{S}_{\mathcal{M}} = f_{\phi, \theta_d}(H^*)$   $\triangleright$  Eq. 6
22:  Calculate loss and optimize  $\pi_{tk}$   $\triangleright$   $\phi$  can be tuned optionally
23: return Trained models  $f_{\theta_e}, f_{\theta_d}, f_{\phi}$  and prompts  $\pi_{dm}, \pi_{tk}$ 

```

5 Experiment

5.1 Experimental Settings

Datasets. We evaluate our STGP using four public traffic flow datasets: METR-LA, PEMS-BAY [19], Chengdu, and Shenzhen [1].

Baselines. We consider 19 baselines belonging to three categories.

- **Statistical Methods:** HA, MEAN, KNN, ARIMA [37] calculate the statistical properties of input data to predict node signals.
- **Typical Neural Networks:** DCRNN [19], GWN [40], FOGS [31], DSTAGNN [18], STEP [32], KCN [2], IGNNK [38], SATCN [39], DualSTN [10], and INCREASE [46] are models for forecasting or kriging. To apply them in the transfer learning scenario, we optimize them using the Reptile [28] meta-learning framework.
- **Transfer Learning Models:** AdaRNN [4], ST-GFSL [26], DSAT-Net [34], TPB [25], TransGTR [15] are state-of-the-art time series or spatio-temporal graph methods for transfer learning.

Evaluation Settings and Implementation. We adopt the same setting as previous works [25, 26], where the datasets are divided into source, target, and test datasets. The source datasets comprise three datasets for pre-training. We sequentially split them into training and validation segments with a ratio of 8 : 2, where the model is selected on the validation part. Then only the first three-day data in the remaining dataset is regarded as the target dataset for domain and task prompting. The subsequent three-day data is utilized for model selection and the rest of the segment is adopted to compute metrics. We adopt mean absolute error (MAE) and root mean squared error (RMSE) as the metrics.

Hyperparameters are the same in all settings for STGP. Each patch, representing one-hour data, has a length of $L = 12$, with $T = 25$ patches in total. Both Graphormer and TSFormer contain 4 layers with a hidden size of 128. The decoder contains 6 spatio-temporal gated layers with a hidden size of 32. We also add skip and residual connections into the decoder. The prediction head is a two-layer linear projection with a size of 128 and 256. All prompt banks involve 25 vectors with a size of 128. For extrapolation, we fine-tune the prediction head when optimizing the task prompts, as we empirically find it can yield better results. We run each experiment 5 times to report results. Our model is implemented by PyTorch and trained on one NVIDIA A100 GPU with 32GB memory. The source code is available at <https://github.com/hjf1997/STGP>.

5.2 Performance Evaluation

Forecasting. We conduct forecasting using the similar setting of [25], where 1-day historical data (i.e., 24 patches) is utilized to predict subsequent signals of 1 hour (i.e., 1 patch). To assess Reptile's impact on spatio-temporal transfer learning, we present the results of two baselines directly trained within the target domain. Table 1 reports the metrics, from which we have the following observations. (1) STGP consistently attains superior results on multiple datasets, reducing the metrics of the second-best by up to 5.6% on MAE and 4.5% on RMSE. This performance demonstrates the effectiveness of STGP. (2) TransGTR performs better on some metrics. The probable reason is that it utilizes knowledge distillation to transfer long-term temporal relations from the source domain, which is absent in our model. (3) Simply equipping a model with Reptile only leads

Table 1: Forecasting performance of transfer learning on the four datasets. Each target city only has limited data, whereas the remaining three datasets are regarded as the source domain. The best and second-best are indicated in bold and underlined. Marker * indicates statistical significance, denoting that the p-value from the Student’s T-test is less than 0.01.

Model	Target City	METR-LA								PEMS-BAY							
	Metrics Horizon	MAE (↓)				RMSE (↓)				MAE (↓)				RMSE (↓)			
		15 m	30 m	60 m	avg.	15 m	30 m	60 m	avg.	15 m	30 m	60 m	avg.	15 m	30 m	60 m	avg.
HA	Target Only	3.62	4.23	5.13	4.26	7.33	8.56	10.17	8.54	2.42	2.89	3.70	2.93	5.46	6.52	8.25	6.59
ARIMA		3.22	3.70	5.00	3.65	6.28	7.80	9.80	7.78	2.28	2.49	3.66	2.47	4.52	5.35	7.45	5.36
DCRNN		3.03	3.73	4.81	3.77	5.78	7.34	9.34	7.27	1.81	2.36	3.24	2.38	3.39	4.75	6.37	4.70
GWN		3.08	3.70	4.73	3.69	5.79	7.30	9.10	7.31	1.94	2.40	3.08	2.40	3.60	4.67	6.19	4.67
DCRNN	Reptile	3.01	3.65	4.67	3.69	5.62	7.16	8.96	7.03	1.83	2.43	3.36	2.44	3.36	4.71	6.59	4.68
GWN		3.11	3.75	4.73	3.76	5.87	7.31	9.10	7.21	1.99	2.45	3.14	2.55	3.55	4.64	6.23	4.65
DSTAGNN		3.30	4.10	4.95	4.12	5.90	7.73	9.56	7.74	1.85	2.51	3.59	2.48	3.41	4.79	6.66	4.80
FOGS		3.26	4.11	4.88	4.14	5.95	7.50	9.47	7.51	1.89	2.38	3.37	2.29	3.49	4.54	6.01	3.52
STEP	Transfer	3.20	3.99	4.76	4.00	5.70	7.14	8.53	7.12	1.72	2.30	3.08	2.31	3.21	4.38	5.83	4.40
AdaRNN		3.05	3.68	4.51	3.66	5.66	7.15	8.60	7.10	1.79	2.33	3.04	2.38	3.38	4.60	5.98	4.58
ST-GFSL		<u>3.00</u>	3.79	4.58	3.80	5.72	7.21	8.67	7.18	1.77	2.20	2.95	2.24	3.27	4.50	5.92	4.45
DSATNet		3.03	3.66	4.51	3.63	5.70	7.15	8.78	7.08	1.64	2.16	2.88	2.18	3.26	4.36	5.89	4.30
TPB	Transfer	3.07	3.80	4.66	3.83	5.69	<u>7.03</u>	8.52	6.95	<u>1.62</u>	2.12	2.83	2.15	3.24	<u>4.33</u>	5.76	<u>4.23</u>
TransGTR		3.01	<u>3.64</u>	<u>4.44</u>	<u>3.62</u>	<u>5.60</u>	7.12	<u>8.49</u>	<u>6.89</u>	1.60	<u>2.13</u>	<u>2.79</u>	<u>2.14</u>	3.04	4.35	<u>5.68</u>	4.26
STGP	Transfer	2.97*	3.54*	4.23*	3.52*	5.48*	6.77*	8.19*	6.67*	1.74*	<u>2.13</u>	2.70*	2.12	<u>3.21*</u>	4.18*	5.46*	4.14*
Std.		.003	.010	.008	.009	.029	.011	.019	.011	.017	.014	.012	.011	.013	.015	.014	.017

Model	Target City	Chengdu								Shenzhen							
	Metrics Horizon	MAE (↓)				RMSE (↓)				MAE (↓)				RMSE (↓)			
		15 m	30 m	60 m	avg.	15 m	30 m	60 m	avg.	15 m	30 m	60 m	avg.	15 m	30 m	60 m	avg.
HA	Target Only	2.69	3.13	3.65	3.01	3.69	4.57	5.27	4.40	2.17	2.66	3.04	2.54	3.33	4.05	4.63	3.89
ARIMA		2.97	3.28	4.32	3.19	3.78	4.64	5.51	4.58	2.35	2.99	3.60	2.90	4.32	4.73	5.58	4.62
DCRNN		2.27	3.05	3.60	2.82	3.26	4.38	5.07	4.03	2.05	2.61	3.10	2.49	3.17	3.98	4.60	3.79
GWN		2.25	2.96	3.42	2.74	3.20	4.26	4.90	3.94	1.86	2.46	2.84	2.27	2.76	3.69	4.28	3.41
DCRNN	Reptile	2.31	3.16	3.96	2.97	3.33	4.55	5.42	4.28	1.94	2.57	3.07	2.38	2.84	3.81	4.52	3.53
GWN		2.19	2.81	3.21	2.62	3.12	4.08	4.65	3.79	1.88	2.46	2.82	2.27	2.77	3.68	4.26	3.40
DSTAGNN		2.36	3.01	3.40	2.85	3.21	4.20	4.97	3.99	1.98	2.43	2.95	2.42	2.90	3.69	4.27	3.38
FOGS		2.23	2.80	3.31	2.61	3.18	4.30	4.77	4.05	1.96	2.36	2.80	2.14	2.88	3.61	4.35	3.33
STEP	Transfer	2.21	2.70	3.20	2.49	3.05	3.99	4.29	3.76	1.95	2.40	2.55	2.17	2.73	3.44	3.80	3.22
AdaRNN		2.18	2.91	3.40	2.73	3.09	3.97	4.82	3.68	1.92	2.48	2.88	2.28	2.85	3.63	4.26	3.38
ST-GFSL		2.10	2.80	3.35	2.59	3.02	3.88	4.60	3.57	1.90	2.36	2.71	2.19	2.70	3.53	4.19	3.26
DSATNet		2.06	2.70	3.03	2.46	3.02	4.01	4.53	3.66	1.86	2.34	2.64	2.14	2.73	3.51	4.00	3.22
TPB	Transfer	2.08	<u>2.63</u>	3.02	2.45	2.98	3.84	4.34	3.52	<u>1.85</u>	2.32	2.61	<u>2.13</u>	2.70	<u>3.45</u>	3.91	3.20
TransGTR		<u>2.05</u>	2.65	<u>2.80</u>	<u>2.41</u>	<u>2.95</u>	<u>3.82</u>	<u>4.26</u>	<u>3.49</u>	1.89	<u>2.30</u>	<u>2.47</u>	2.15	<u>2.69</u>	3.49	<u>3.79</u>	<u>3.19</u>
STGP	Transfer	1.98*	2.54*	2.74*	2.30*	2.85*	3.72*	4.02*	3.37*	1.82*	2.27*	2.42*	2.08*	2.66*	3.39*	3.69*	3.11*
Std.		.012	.010	.010	.014	.011	.010	.016	.013	.008	.012	.011	.013	.017	.013	.010	.012

Table 2: Kriging performance of transfer learning on the datasets. Trdu./Indu. denote transductive/inductive settings.

Model	Target City	METR-LA				PEMS-BAY				Chengdu				Shenzhen					
		Metrics		MAE (↓)		RMSE (↓)		MAE (↓)		RMSE (↓)		MAE (↓)		RMSE (↓)		MAE (↓)		RMSE (↓)	
	Setting	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.
MEAN	Target Only		10.28		14.86		5.49		9.28		7.42		9.78		8.81		11.08		
KNN			9.11		12.88		4.20		7.55		7.01		9.40		8.89		11.17		
KCN	Reptile		5.69	7.76	8.72	11.78	3.89	4.76	6.91	8.20	6.64	7.03	8.99	9.31	5.70	8.28	8.08	10.98	
IGNNK			5.67	7.60	9.33	11.68	3.94	4.30	7.16	7.22	6.71	7.56	8.90	9.94	5.41	8.20	7.58	10.90	
SATCN			<u>4.81</u>	7.51	<u>7.85</u>	11.59	3.48	3.85	6.51	6.84	6.61	7.14	9.03	9.42	5.53	<u>8.15</u>	7.89	10.87	
DualSTN			5.62	<u>7.12</u>	9.60	<u>11.46</u>	3.54	3.74	6.82	<u>6.75</u>	<u>5.95</u>	<u>6.42</u>	<u>8.07</u>	<u>8.71</u>	5.37	8.18	7.66	10.91	
INCREASE			5.03	7.38	8.24	11.54	<u>3.46</u>	<u>3.72</u>	<u>6.41</u>	6.78	6.14	6.63	8.53	8.80	<u>5.34</u>	8.16	7.61	<u>10.85</u>	
STGP			4.53*	6.98*	7.75*	11.39	3.39*	3.66*	6.28*	6.68*	5.47*	6.33*	7.70*	8.64	5.25*	8.08*	7.50	10.79	
Std.	Transfer		.022	.062	.037	.081	.013	.033	.018	.040	.033	.055	.014	.053	.019	.083	.049	.088	

Table 3: Extrapolation performance of transfer learning on the datasets. The results are averaged over all 12 future horizons.

Model	Target City	METR-LA				PEMS-BAY				Chengdu				Shenzhen			
		MAE (↓)		RMSE (↓)		MAE (↓)		RMSE (↓)		MAE (↓)		RMSE (↓)		MAE (↓)		RMSE (↓)	
		Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.	Trdu.	Indu.
MEAN	Target Only	10.64		15.27		5.56		9.31		7.53		9.74		8.91		11.63	
GWN	Reptile	5.41	9.26	9.69	15.10	4.32	5.01	8.09	9.86	5.83	7.21	8.25	9.61	4.02	8.47	5.74	11.40
DCRNN		<u>5.10</u>	9.41	<u>9.19</u>	15.57	<u>3.64</u>	5.12	<u>6.97</u>	10.14	6.23	7.10	8.62	9.58	<u>3.85</u>	8.64	<u>5.31</u>	11.54
IGNNK		6.67	9.14	10.33	14.63	3.90	5.05	7.13	9.90	6.52	7.43	8.81	9.71	4.41	8.52	6.47	11.38
SATCN		7.29	9.13	12.41	14.88	4.65	5.13	9.04	10.13	7.09	7.20	9.51	9.66	4.87	8.70	7.15	11.58
ST-GFSL	Transfer	7.73	9.31	12.31	14.99	4.27	4.59	7.61	7.92	6.51	6.79	8.74	9.41	4.75	8.55	6.99	11.45
TPB		7.97	<u>9.01</u>	12.50	14.42	4.40	4.53	7.79	<u>7.83</u>	<u>6.12</u>	6.67	<u>8.37</u>	9.27	4.66	8.38	6.96	<u>11.23</u>
TransGTR		7.33	8.87	11.22	<u>14.12</u>	4.55	4.66	7.95	8.10	6.42	<u>6.62</u>	8.65	<u>9.10</u>	4.58	8.45	6.81	11.36
STGP	Transfer	5.04	8.51*	9.03*	13.49	3.41*	4.13*	6.47*	7.52*	5.46*	6.51*	7.67	8.91*	3.61*	8.22*	5.04*	11.07
Std.		.083	.094	.084	.159	.010	.045	.021	.088	.019	.086	.024	.091	.057	.051	.074	.090

to marginal improvement. This highlights the need for transfer learning systems in spatio-temporal graphs amid data scarcity.

Kriging. We conduct experiments on both transductive and inductive settings for kriging. In the transductive setting, the model has access to the structure of unobserved nodes during training, thereby presenting a greater challenge. Conversely, the inductive setting secludes these nodes throughout the training and utilizes them solely for evaluation. We set the ratio of observed and unobserved nodes to 2 : 1 and report the results in Table 2. It shows that (1) Our STGP consistently obtains the best results, reducing MAE and RMSE by up to 7.0% and 8.0%. This improvement underscores its superiority. (2) Several neural networks struggle to outperform statistical models on the datasets. This highlights the challenge of kriging when models predict nodes without any signals amidst limited training data.

Extrapolation. We conduct evaluations following the same experimental settings. As a new task, we compare STGP against compatible baselines. As Table 3 shows, our analysis reveals that (1) STGP consistently surpasses the baselines across metrics by a significant margin. In particular, STGP achieves a reduction in MAE and RMSE by up to 8.8% and 10.7%, marking the highest performance among the tasks. (2) STGP yields higher standard divisions on extrapolation and kriging. As they require predicting node signals solely based on graph structures, the restricted spatial information poses challenges for the model in avoiding local minima during the optimization.

5.3 Ablation Study

We conduct ablation studies on the three tasks by using METR-LA as the target domain. For kriging and extrapolation, we adopt the inductive setting. The results are reported in Fig. 4.

Prompt Tuning. We remove all prompts, evaluating the pre-trained model on the target domain directly (i.e., zero-shot setting) or fine-tuning it, denoted as **zero** and **ft**. We observed that both settings yield worse performance. This suggests that domain shift impedes a pre-trained model from directly adapting to a target domain. Meanwhile, the effectiveness of fine-tuning is adversely affected due to limited target data and discrepancies between pre-training and downstream task objectives. On the contrary, STGP unifies the tasks, using prompts for effective domain and task transfer.

Domain Prompting. We remove temporal and spatial prompt banks (**w/o dp**) or replace them with a single bank (**r/p sdp**). Our

results show that separate banks achieve the best performance. The explanation is that the two dimensions involve domain shifts with different characteristics, which should be modeled individually.

Task Prompting. We remove task prompts (**w/o tp**) or replace them with a single bank (**r/p stp**). It shows that STGP requires both banks for optimal results. This is because masked patches are filled by a mask token, which is different from patches with extracted dependencies from the encoder. The two prompt banks benefit the decoder in predicting masked patches based on captured relations.

Gated Decoder. We analyze the efficacy of our gated decoder by substituting it with a task-specific model, Graph WaveNet (**r/p gwn**). This variant shows comparable forecasting results but experiences a significant deterioration in other tasks. As the design of these forecasting models prioritizes capturing of temporal dependencies [18, 40, 44], the absence of temporal relations in the rest tasks—owing to the replacement of all target nodes with mask tokens—impedes their effectiveness. The problem can also appear when applying kriging methods [10, 39] to forecasting. Conversely, our decoder places equal importance on both spatial and temporal dimensions and dynamically fuses these dependencies via a gating mechanism, which renders it adaptable to a variety of tasks.

5.4 Efficiency Analysis

To assess the efficiency of STGP’s prompting, we compute its complexities. Suppose a bank contains N_p prompts with dimension d_h , the total parameters for the two prompting stages are $O(4N_p d_h)$. For time complexity, assume the model has L layers, the

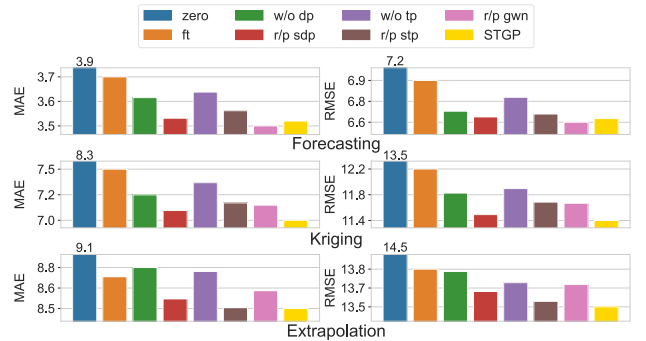
**Figure 4: Model analysis with METR-LA as the target domain.**

Table 4: Tunable parameters and FLOPs comparison. STGP-dp and STGP-tp represent domain and task prompting.

Methods	Params $\times 10^3$	FLOPs $\times 10^9$			
		METR-LA	PEMS-BAY	Chengdu	Shenzhen
ST-GFSL	878	10.82	19.55	38.49	50.36
TPB	567	10.52	19.13	37.92	49.75
TransGTR	1,137	16.48	28.79	54.99	71.51
STGP-dp	3	0.16	0.27	0.43	0.51
STGP-tp	3	0.33	0.52	0.84	1.00

encoder’s complexity is approximated as $O(LTN^2d_h + LNT^2d_h)$. As the decoder is a combination of GCN and TCN, the complexity is $O(LN^2d_h + Ltkd_h^2)$, where k is the kernel size. The prompting by Eq. 7 incurs an extra cost of $O(3N_pTNd_h)$, which is minimal given $N_p \ll LT \ll LN$. We observe that, in terms of dataset size, the parameter cost is invariant while the time cost scales linearly.

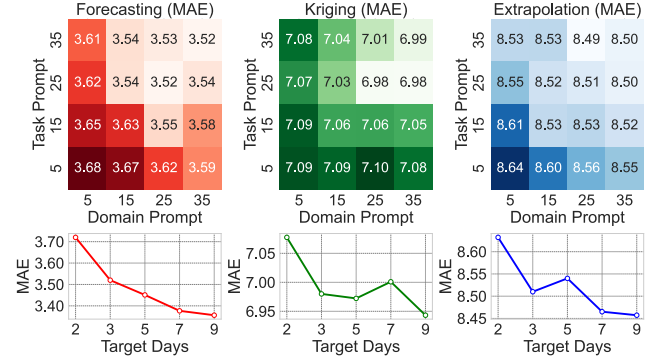
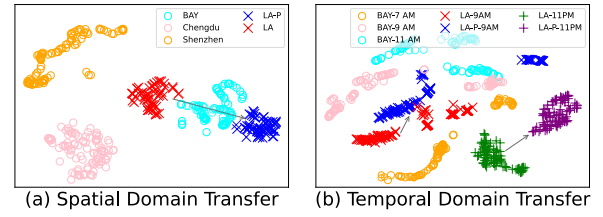
To empirically compare our model with transfer learning baselines, we present the number of tunable parameters and floating point operations (FLOPs) for forecasting. We set the batch size to 1 and report FLOPs of one iteration. As shown in Table 4, the results reveal that TransGTR, requiring training of TSFormer on the target domain, incurs the most cost. ST-GFSL, employing a generator for parameter prediction, also shows high computational demand. TPB, which tunes WaveNet, has the least demand among baselines. In contrast, STGP not only exceeds the performance of these baselines but also significantly reduces the need for parameters and FLOPs, by factors of approximately 100 and 25, respectively.

5.5 Hyperparameter Study

We first evaluate the number of domain and task prompts, using METR-LA as the target domain. The first row of Fig. 5 displays the MAE results, with lighter colors denoting superior performance. The results reveal that optimal performance requires a substantial number of both prompts. A large number of one prompt type coupled with a small number of the other type hinders the model’s performance. This finding suggests distinct roles for the two prompts: domain prompts address domain transfer, while task prompts focus on task-specific properties. Next, we discuss the effect of available data on the target domain. As the second row shows, the MAE decreases with the increase of training data. This suggests that data scarcity is clearly a bottleneck for model performance, necessitating the introduction of effective transfer learning methods like STGP.

5.6 Domain Transfer Analysis

We innovatively utilize prompt tuning to achieve domain transfer for spatio-temporal graphs. To analyze its efficacy, we use METR-LA as the target domain and visualize hidden representations of patches. For spatial domain transfer, we randomly select 100 nodes from each of the datasets and average representations of a node at all time steps. As shown in Fig. 6 (a), after prompting (LA-P), the METR-LA points (LA) shift away from Chengdu and Shenzhen while staying close to PEMS-BAY (BAY). This might be explained by the proximity between Los Angeles and the Bay area. For temporal transfer, we average nodes at the same time and visualize points at different times. Fig. 6 (b) shows that METR-LA points at 9 AM


Figure 5: Effect of prompt numbers and the number of target days for training on the METR-LA dataset.

Figure 6: Visualization of domain transfer. \circ represents source domain; \times , $+$ denote target domain. The arrows indicate the representation shift of a target domain.

move towards PEMS-BAY at 9 AM, while the 11 PM points show no clear trend. As the model is pre-trained on a source domain, these findings suggest that domain prompting can effectively transform target data to align with the latent space of the source domain.

6 Conclusion

In this paper, we present STGP, a novel prompt-based approach for spatio-temporal graph transfer learning. We develop a unified template alongside a task-agnostic network architecture suitable for multiple tasks. We then propose a two-stage prompting pipeline tailored for domain and task transfer. Comprehensive experiments and discussions demonstrate that STGP effectively captures domain knowledge and task-specific properties, consistently achieving state-of-the-art performance across diverse datasets and tasks. Regarding future work, we find that the downstream tasks unified by STGP fall under predictive learning, where known data is utilized to predict unknown data in a regression setting. Although received less attention, spatio-temporal graphs also encompass other applications, such as classification. Exploring the applicability of our unified template and network architecture to these tasks presents an intriguing direction for future research.

Acknowledgments

This research is supported by Singapore Ministry of Education Academic Research Fund Tier 2 under MOE’s official grant number T2EP20221-0023. It is also part of the programme DesCartes and is supported by the National Research Foundation, Prime Minister’s Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

References

- [1] 2020. Didi chuxing gaia initiative. <https://gaia.didichuxing.com>.
- [2] Gabriel Appleby, Linfeng Liu, and Li-Ping Liu. 2020. Kriging convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 3187–3194.
- [3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.
- [4] Yuntao Du, Jindong Wang, Wenjie Feng, Sinno Pan, Tao Qin, Renjun Xu, and Chongjun Wang. 2021. Adarnn: Adaptive learning and forecasting of time series. In *Proceedings of the 30th ACM international conference on information & knowledge management*. 402–411.
- [5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*. PMLR, 1126–1135.
- [6] Chunjiang Ge, Rui Huang, Mixue Xie, Zihang Lai, Shiji Song, Shuang Li, and Gao Huang. 2023. Domain adaptation via prompt learning. *IEEE Transactions on Neural Networks and Learning Systems* (2023).
- [7] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 16000–16009.
- [8] Zhenyu Hou, Yufei He, Yukuo Cen, Xiao Liu, Yuxiao Dong, Evgeny Kharlamov, and Jie Tang. 2023. GraphMAE2: A Decoding-Enhanced Masked Self-Supervised Graph Learner. In *Proceedings of the ACM Web Conference 2023*. 737–746.
- [9] Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. 2022. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 594–604.
- [10] Junfeng Hu, Yuxuan Liang, Zhencheng Fan, Li Liu, Yifang Yin, and Roger Zimmermann. 2023. Decoupling long- and short-term patterns in spatiotemporal inference. *IEEE Transactions on Neural Networks and Learning Systems* (2023).
- [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] Guangyin Jin, Yuxuan Liang, Yuchen Fang, Zezhi Shao, Jincai Huang, Junbo Zhang, and Yu Zheng. 2023. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE Transactions on Knowledge and Data Engineering* (2023).
- [13] Ming Jin, Qingsong Wen, Yuxuan Liang, Chaoli Zhang, Siqiao Xue, Xue Wang, James Zhang, Yi Wang, Haifeng Chen, Xiaoli Li, et al. 2023. Large models for time series and spatio-temporal data: A survey and outlook. *arXiv preprint arXiv:2310.10196* (2023).
- [14] Yilun Jin, Kai Chen, and Qiang Yang. 2022. Selective cross-city transfer learning for traffic prediction via source city region re-weighting. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 731–741.
- [15] Yilun Jin, Kai Chen, and Qiang Yang. 2023. Transferable Graph Structure Learning for Graph-based Traffic Forecasting Across Cities. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1032–1043.
- [16] Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. Ammus: A survey of transformer-based pretrained models in natural language processing. *arXiv preprint arXiv:2108.05542* (2021).
- [17] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. 2022. Transformers in vision: A survey. *ACM computing surveys (CSUR)* 54, 10s (2022), 1–41.
- [18] 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*. PMLR, 11906–11917.
- [19] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*.
- [20] Zhonghang Li, Lianghao Xia, Yong Xu, and Chao Huang. 2024. FlashST: A Simple and Universal Prompt-Tuning Framework for Traffic Prediction. *arXiv:2405.17898* [cs.LG]
- [21] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *Comput. Surveys* 55, 9 (2023).
- [22] Xu Liu, Junfeng Hu, Yuan Li, Shizhe Diao, Yuxuan Liang, Bryan Hooi, and Roger Zimmermann. 2023. UniTime: A Language-Empowered Unified Model for Cross-Domain Time Series Forecasting. *arXiv preprint arXiv:2310.09751* (2023).
- [23] Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, Lei Bai, Chao Huang, Zhengguang Liu, Bryan Hooi, and Roger Zimmermann. 2023. LargeST: A Benchmark Dataset for Large-Scale Traffic Forecasting. In *Advances in Neural Information Processing Systems*.
- [24] Zemin Liu, Xingtong Yu, Yuan Fang, and Xinming Zhang. 2023. Graphprompt: Unifying pre-training and downstream tasks for graph neural networks. In *Proceedings of the ACM Web Conference 2023*. 417–428.
- [25] Zhanyu Liu, Guanjie Zheng, and Yanwei Yu. 2023. Cross-city Few-Shot Traffic Forecasting via Traffic Pattern Bank. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 1451–1460.
- [26] Bin Lu, Xiaoying Gan, Weinan Zhang, Huaxiu Yao, Luoyi Fu, and Xinbing Wang. 2022. Spatio-Temporal Graph Few-Shot Learning with Cross-City Knowledge Transfer. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1162–1172.
- [27] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *Comput. Surveys* 56, 2 (2023), 1–40.
- [28] Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999* (2018).
- [29] 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 *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1720–1730.
- [30] Yujia Qin, Xiaozhi Wang, Yusheng Su, Yankai Lin, Ning Ding, Zhiyuan Liu, Juanzi Li, Lei Hou, Peng Li, Maosong Sun, et al. 2021. Exploring lowdimensional intrinsic task subspace via prompt tuning. *arXiv preprint arXiv:2110.07867* (2021).
- [31] Xuan Rao, Hao Wang, Liang Zhang, Jing Li, Shuo Shang, and Peng Han. 2022. FOGS: First-order gradient supervision with learning-based graph for traffic flow forecasting. In *Proceedings of International Joint Conference on Artificial Intelligence, IJCAI*.
- [32] Zezhi Shao, Zhao Zhang, Fei Wang, and Yongjun Xu. 2022. Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1567–1577.
- [33] Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. 2023. All in One: Multi-Task Prompting for Graph Neural Networks. (2023).
- [34] Yihong Tang, Ao Qu, Andy HF Chow, William HK Lam, SC Wong, and Wei Ma. 2022. Domain adversarial spatial-temporal network: a transferable framework for short-term traffic forecasting across cities. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 1905–1915.
- [35] Leye Wang, Xu Geng, Xiaojuan Ma, Feng Liu, and Qiang Yang. 2019. Cross-city Transfer Learning for Deep Spatio-temporal Prediction. In *IJCAI International Joint Conference on Artificial Intelligence*. 1893.
- [36] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. 2022. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125* (2022).
- [37] Billy M Williams and Lester A Hoel. 2003. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of transportation engineering* 129, 6 (2003), 664–672.
- [38] Yuankai Wu, Dingyi Zhuang, Aurelie Labbe, and Lijun Sun. 2021. Inductive graph neural networks for spatiotemporal kriging. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4478–4485.
- [39] Yuankai Wu, Dingyi Zhuang, Mengying Lei, Aurelie Labbe, and Lijun Sun. 2021. Spatial aggregation and temporal convolution networks for real-time kriging. *arXiv preprint arXiv:2109.12144* (2021).
- [40] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*.
- [41] Hao Xue and Flora D Salim. 2023. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering* (2023).
- [42] Huaxiu Yao, Yiding Liu, Ying Wei, Xianfeng Tang, and Zhenhui Li. 2019. Learning from multiple cities: A meta-learning approach for spatial-temporal prediction. In *The world wide web conference*. 2181–2191.
- [43] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do Transformers Really Perform Badly for Graph Representation?. In *Thirty-Fifth Conference on Neural Information Processing Systems*.
- [44] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 3634–3640.
- [45] Yuan Yuan, Jingtao Ding, Jie Feng, Depeng Jin, and Yong Li. 2024. UniST: A Prompt-Empowered Universal Model for Urban Spatio-Temporal Prediction. *arXiv preprint arXiv:2402.11838* (2024).
- [46] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, Jianzhong Qi, Chaochao Chen, and Longbiao Chen. 2023. INCREASE: Inductive Graph Representation Learning for Spatio-Temporal Kriging. In *Proceedings of the ACM Web Conference 2023*. 673–683.
- [47] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proc. IEEE* 109, 1 (2020), 43–76.