STD-PLM: Understanding Both Spatial and Temporal Properties of Spatial-Temporal Data with PLM

Yiheng Huang[1*], Xiaowei Mao[1*], Shengnan Guo[1,2], Yubin Chen[1], Junfeng Shen[1], Tiankuo Li[1], Youfang Lin[1,2], Huaiyu Wan[1,2†]

[1]School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
[2]Beijing Key Laboratory of Traffic Data Mining and Embodied Intelligence, Beijing, China
{huangyiheng,maoxiaowei,guoshn,chenyubin,tanklee,jfshen,yflin,hywan}@bjtu.edu.cn

Abstract

Spatial-temporal forecasting and imputation are important for real-world intelligent systems. Most existing methods are tailored for individual forecasting or imputation tasks but are not designed for both. Additionally, they are less effective for zero-shot and few-shot learning. While pre-trained language model (PLM) have exhibited strong pattern recognition and reasoning abilities across various tasks, including few-shot and zero-shot learning, their applications in spatial-temporal data understanding has been constrained by insufficient modeling of complex correlations such as the temporal correlations, spatial connectivity, non-pairwise and high-order spatial-temporal correlations within data. In this paper, we propose STD-PLM for understanding both spatial and temporal properties of Spatial-Temporal Data with PLM, which is capable of implementing both spatial-temporal forecasting and imputation tasks. STD-PLM understands spatial-temporal correlations via explicitly designed spatial and temporal tokenizers. Topology-aware node embeddings are designed for PLM to comprehend and exploit the topology structure of data in inductive manner. Furthermore, to mitigate the efficiency issues introduced by the PLM, we design a sandglass attention module (SGA) combined with a specific constrained loss function, which significantly improves the model's efficiency while ensuring performance. Extensive experiments demonstrate that STD-PLM exhibits competitive performance and generalization capabilities across the forecasting and imputation tasks on various datasets. Moreover, STD-PLM achieves promising results on both few-shot and zero-shot tasks.

Code — https://github.com/Hyheng/STD-PLM

Introduction

Understanding both spatial and temporal properties of spatial-temporal data is crucial for various real-world dynamic systems such as intelligent transportation (Ding, Zhao, and Jiao 2002) and urban planning (Song, Li, and Li 2017). In practice, spatial-temporal forecasting and imputation are the two most pivotal and common tasks. Specifically, precise spatial-temporal forecasting aids in effective traffic management and travel planning, while spatial-temporal imputation enables precise analysis of spatial-temporal patterns and supporting other dependent tasks. Although extensive studies have achieved satisfactory accuracy in spatial-temporal forecasting and imputation, they rely on extensive historical data for training. However, obtaining comprehensive datasets for all the studied regions is challenging due to the high cost of collecting long-term data. Some studies (Miao et al. 2024) have enhanced data utilization through continuous learning, but they still fall short in handling scenarios with extremely scarce data. Therefore, the capability for zero-shot (Wang et al. 2019) and few-shot (Wang et al. 2020) learning is crucial for the broader applicability of spatial-temporal models. Moreover, existing methods are usually tailored to specific tasks and not designed for both forecasting and imputation. Each method requires domain expertise and task-specific designs, increasing costs and complicating the deployment of forecasting and imputation methods. While some studies (Miao et al. 2022) have applied multi-task learning to integrate various objectives, they mainly focus on forecasting tasks rather than jointly addressing forecasting and imputation. To summarize, we need a method that possesses powerful zero-shot and few-shot learning capabilities and is versatile for both spatial-temporal forecasting and imputation tasks, making it practical and applicable in real-world scenarios.

We have noticed that PLM are widely recognized for strong performance in zero-shot and few-shot learning across a diverse range of tasks (Ge et al. 2024). However, due to the significant differences between spatial-temporal and textual data, it is challenging for PLM to comprehend spatial-temporal data. In addition, training a spatial-temporal PLM from scratch is non-trivial due to the limited availability of spatial-temporal data compared to textual data. To address this issue, researchers have attempted to adapt PLM for understanding spatial-temporal data (Liu et al. 2024b,a; Zhang et al. 2024). However, these methods offer insufficient representation for spatial-temporal data. Thus, they are less effective for understanding both spatial and temporal properties of spatial-temporal data with PLM. First, existing PLM-based methods design tokens along the spatial dimension but ignore the tokens obtained from the temporal dimension. Second, topological connectivity information inherent in spatial-temporal data is less explored.

*These authors contributed equally.
†Corresponding author: hywan@bjtu.edu.cn

Thirdly, existing methods do not take into account that under the large embedding dimensions of the PLM, as the number of tokens increases, the training and inference costs of the model will increase rapidly. On the other hand, existing PLM-based spatial-temporal forecasting methods primarily concentrate on forecasting tasks and overlook the task of imputation. This limitation restricts their versatility and efficiency for practical applications.

To solve the above limitations and challenges, we propose STD-PLM for understanding both spatial-temporal properties of Spatio-Temporal Data with PLM. First, we create spatial-temporal embeddings for fully capturing temporal correlations and exploiting the topology structure of data. Second, we develop a spatial tokenizer and a temporal tokenizer to activate the ability of pre-trained PLM to understand spatial-temporal data from both the spatial and temporal dimensions while incorporating the topology structure of data. Lastly, we design the sandglass attention module with a constrained loss function, which can effectively model more important and sparser non-pairwise and higher-order correlations, significantly reducing computational costs. Based on the explicitly design, STD-PLM is not only versatile for spatial-temporal forecasting and imputation but also exhibits accurate few-shot and zero-shot learning capabilities on spatial-temporal data. In summary, our contributions can be summarized as follows:

- We propose STD-PLM for accurate spatial-temporal forecasting and imputation, as well as zero-shot and few-shot learning, by activating PLM to understand both the spatial and temporal properties of spatial-temporal data.
- We design spatial-temporal tokenizers to generate tokens that embed spatial and temporal features, integrating topology to enable the PLM to capture fine-grained spatial and coarse-grained temporal information.
- We design the sandglass attention module and its constrained loss function, which significantly reduces computational overhead while ensuring performance.

Related Work

Spatial-Temporal Data Forecasting. In recent years, a large number of models that can effectively model spatial-temporal dependencies have emerged. Several studies (Shi et al. 2015) address temporal dynamics using RNNs. To deal with spatial dependency, the data can be divided into grids (Lin et al. 2019; Zonoozi et al. 2018; Zhang, Zheng, and Qi 2017; Yao et al. 2018), and then CNNs are utilized to capture spatial correlations. But not all data can be partitioned into grid form. In order to achieve a more general and effective model, researchers introduced the graph convolution model (Kipf and Welling 2016; Defferrard, Bresson, and Vandergheynst 2016) thereby implementing spatial feature aggregation based on adjacency matrices (Li et al. 2018; Bai et al. 2019; Han et al. 2020; Wu et al. 2019; Song et al. 2020; Bai et al. 2020). Models such as GMAN (Zheng et al. 2020),ASTGCN (Guo et al. 2019) and ASTGNN (Guo et al. 2021) utilize attention mechanisms to further deal with temporal and spatial correlations. PDFormer (Jiang et al. 2023) proposed a delay-aware feature transformation module to explicit model the temporal delay of propagation. The current work has a fairly good forecasting accuracy, but most of the models do not take into account the few-shot learning, zero-shot learning.

Spatial-Temporal Data Imputation. Initially, in order to realize imputation in more complex scenes, successive works based on low-rank matrix completion (Mazumder, Hastie, and Tibshirani 2010; Yu, Rao, and Dhillon 2016) were proposed. LATC (Chen et al. 2021) effectively captures local and global trends in the data by combining low-rank matrix complementation with auto-regressive model. The development of deep learning has also contributed to the research on imputation. Brits (Cao et al. 2018) build an efficient self-supervised imputation model through bidirectional LSTM. E2GAN (Luo et al. 2019), GAIN (Yoon, Jordon, and Schaar 2018)and other GAN (Goodfellow et al. 2020) based models transforms the imputation problem into a generative problem. In addition, some VAE (Kingma and Welling 2014) based works (Shukla and Marlin 2021) recovers missing values by modeling the distribution of the data in the hidden space. Recently the powerful effect of Diffusion Model (Ho, Jain, and Abbeel 2020) in the field of image generation attracts a large number of researchers. CSDI (Tashiro et al. 2021) is based on the Diffusion Model, which recovery of missing values from noise by a well-designed denoising network. Compared to CSDI, another diffusion model PriSTI (Liu et al. 2023a) enhances the utilization of conditional information and the construction of geographic prior knowledge. Existing imputation models face the same problem as forecasting models, both need to reduce the need for training samples and enhance generalization capabilities.

Pre-trained Language Model. PLM provide a powerful and unified framework for different tasks. There has been a significant amount of recent work validating the effectiveness of the PLM in other modalities. One Fits All(OFA) (Zhou et al. 2024) demonstrated the feasibility of using PLM for time series data through the experimental results of eight different temporal tasks, and put forward preliminary explanations. Articles such as Time-LLM (Jin et al. 2023; Cao et al. 2024) make more detailed modifications to PLM, using methods such as reprogramming, prompt pool, and contrast learning to align temporal and textual data. Models such as STLLM (Liu et al. 2024a) and STGLLM (Liu et al. 2024b) enhance PLM's perception of spatial-temporal data by constructing tokens in the dimension of space. Existing PLM-based models have achieved certain results, confirming that PLM has the ability to handle spatial-temporal data. However, they underutilize spatial information and are difficult to accurately capture complex spatial dependencies, and there is much room for improvement.

Sandglass Attention. There are some prior works whose methods are similar to the sandglass attention, such as SSTBAN (Guo et al. 2023) and CrossFormer (Zhang and Yan 2023), which both learn higher-order correlations through learnable reference points combined with an attention mechanism to improve model efficiency. Although these studies have achieved good results, they still have some limitations, such as not fully utilizing the adjacency relationships in the

original graph and being unable to achieve zero-shot learning between different graphs.

Methodology

Problem Definition

Spatial-Temporal data. Considering spatial-temporal data with T time slices and N nodes, we represent it as $\boldsymbol{\mathcal{X}} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_T\} \in \mathbb{R}^{T\times N\times C}$. Here, $\boldsymbol{X}_t = \{\boldsymbol{x}_{t,1}, \boldsymbol{x}_{t,2}, \ldots, \boldsymbol{x}_{t,N}\} \in \mathbb{R}^{N\times C}$, where $\boldsymbol{x}_{t,n}$ represents the feature of node n at time t, and C is the number of features.

We use a binary mask tensor $\boldsymbol{\mathcal{M}} \in \{0,1\}^{T\times N\times C}$ to denote the positions of missing values in $\boldsymbol{\mathcal{X}}$, where $m_{t,n,c} = 0$ indicates that the data is missing, and $m_{t,n,c} = 1$ indicates that the data is observed. To describe the relationships between nodes, we introduce a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$. Here, V represents the set of nodes in the data. $\mathcal{E}$ is the set of edges. $\mathbf{A} \in \mathbb{R}^{N\times N}$ is the adjacency matrix.

Forecasting task. Given the historical spatial-temporal data with T time slices $\boldsymbol{\mathcal{X}}$ and the corresponding graph structure, the objective is to forecast future data $\{\boldsymbol{X}_{t+1}, \ldots, \boldsymbol{X}_{t+T-2}, \boldsymbol{X}_{t+T-1}\}$ for the next T time slices.

Imputation task. Given incomplete spatial-temporal data with T time slices $\boldsymbol{\mathcal{X}}$, mask tensor $\boldsymbol{\mathcal{M}}$ and the corresponding graph structure, the goal is to estimate complete data $\{\hat{\boldsymbol{X}}_{t-T+1}, \ldots, \hat{\boldsymbol{X}}_{t-1}, \hat{\boldsymbol{X}}_t\}$.

Few-shot and Zero-shot task. The term "few-shot" refers to training a model based on a limited amount of data. The term "zero-shot" denotes the direct testing of a model trained on a source dataset on a destination dataset, without any additional training.

Model Structure

As shown in Figure 1, our model is tailored for accurate spatial-temporal forecasting and imputation by fine-tuning PLM to understand the dependencies and evolving patterns in spatial-temporal data. To achieve this, we first develop a inductive spatial-temporal embedding module to exploit the topology structure and periodicity of data. Based on this, we design temporal and spatial tokenizers to convert the spatial-temporal data into sequential tokens, activating the ability of PLM to understand the inherent spatial, temporal and spatial-temporal correlations of the data represented in the sequential token format. Lastly, we further incorporate sandglass attention module consisting of a precoder and a decoder to not only improve the model efficiency, but also further capture the non-pairwise and higher-order spatial-temporal correlations.

Spatial-Temporal Embedding. Spatial-temporal data exhibit heterogeneity at different nodes and time. Thus, we design spatial-temporal embeddings, comprising periodic-aware time embedding and topology-aware node embedding, to represent nodes and time in an inductive manner.

Topology-aware node embedding: In the design of node embeddings, we need them not only to reflect the static characteristics of each node but also to incorporate the topology structure to express the relationships between nodes. At the same time, the node embeddings must also possess inductive learning capabilities across different graph structures.

Noticing that the graph Laplacian matrix has two properties that align well with our needs, first, the graph Laplacian matrix contains important information about the graph's structure, such as degree and connectivity. Second, the eigenvectors of the Laplacian matrix are orthogonal, which can effectively distinguish different nodes. An initial idea is to use the eigenvectors of the Laplacian matrix as embeddings for each node. However, considering that if the dimensionality of the embeddings is related to the graph's order N, it would not be transferable across different graphs. Therefore, we choose to generate node embeddings based on the eigenvectors corresponding to the K largest eigenvalues.

Specifically, we define the normalized Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}$, where $\mathbf{D} = \sum_{j=1}^{N} \mathbf{A}_{i,j}$ is the degree matrix and $\mathbf{I}$ is the identity matrix. The eigendecomposition of $\mathbf{L}$ yields $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$, with $\mathbf{V}$ being the matrix of eigenvectors and $\mathbf{\Lambda}$ the diagonal matrix of eigenvalues. By selecting the eigenvectors corresponding to the top K largest eigenvalues, we obtain $\mathbf{V}^{'} \in \mathbb{R}^{N\times K}$. After passing through a linear layer, we broadcast to obtain the topology-aware node embedding $\mathbf{E}_N \in \mathbb{R}^{T\times N\times d_n}$:

$$\mathbf{w}^* = \text{argtop}_K(\text{diag}(\mathbf{\Lambda})), \tag{1}$$

$$\mathbf{V}^{'} = \mathbf{V}[:, \mathbf{w}^*], \quad \mathbf{E}_N = \mathbf{W}_{ne}\mathbf{V}^{'} + \mathbf{b}_{ne}, \tag{2}$$

where "diag" refers to the operation of extracting the diagonal elements of a matrix. $\mathbf{W}_{ne} \in \mathbb{R}^{K\times d_n}$ and $\mathbf{b}_{ne} \in \mathbb{R}^{d_n}$ represent the trainable parameters of the linear layers.

Periodic-aware time embedding: Spatial-temporal data exhibit periodicity, which is essential for the forecasting and imputation task. To avoid overfitting and underfitting, we select two kinds of periodicity that are moderately coarse including time-of-day and day-of-week to create embedding dictionaries, $\mathbf{D}_t \in \mathbb{R}^{288\times d_t}$ and $\mathbf{D}_w \in \mathbb{R}^{7\times d_t}$, where d_t is the dimension of the time embedding. By looking-up and concatenation along with broadcast operations, we finally obtain the time embedding $\mathbf{E}_T \in \mathbb{R}^{T\times N\times 2d_t}$.

Spatial-Temporal Tokenizer. Utilizing a PLM to process spatial-temporal data, the most crucial step is to transform the spatial-temporal data into tokens that the PLM can handle, as all subsequent processing is based on the PLM's capture of the relationships between these tokens. To this end, we meticulously introduce a spatial-temporal tokenizer to generate tokens from both the spatial and temporal dimensions. Specifically, each node generates a spatial token meanwhile two kinds of temporal tokens are designed to represent the current state and evolving pattern of the spatial-temporal data. Through employing the spatial-temporal tokenizer, the original spatial-temporal graph data is converted into token sequences, enabling the subsequently applied PLM not only to capture the temporal and spatial correlations whithin each kind of tokens but also to capture intertwined spatial-temporal correlations between the spatial and temporal tokens.

Spatial Tokenizer: The spatial tokenizer's objective is to aggregate the temporal information of each node separately,

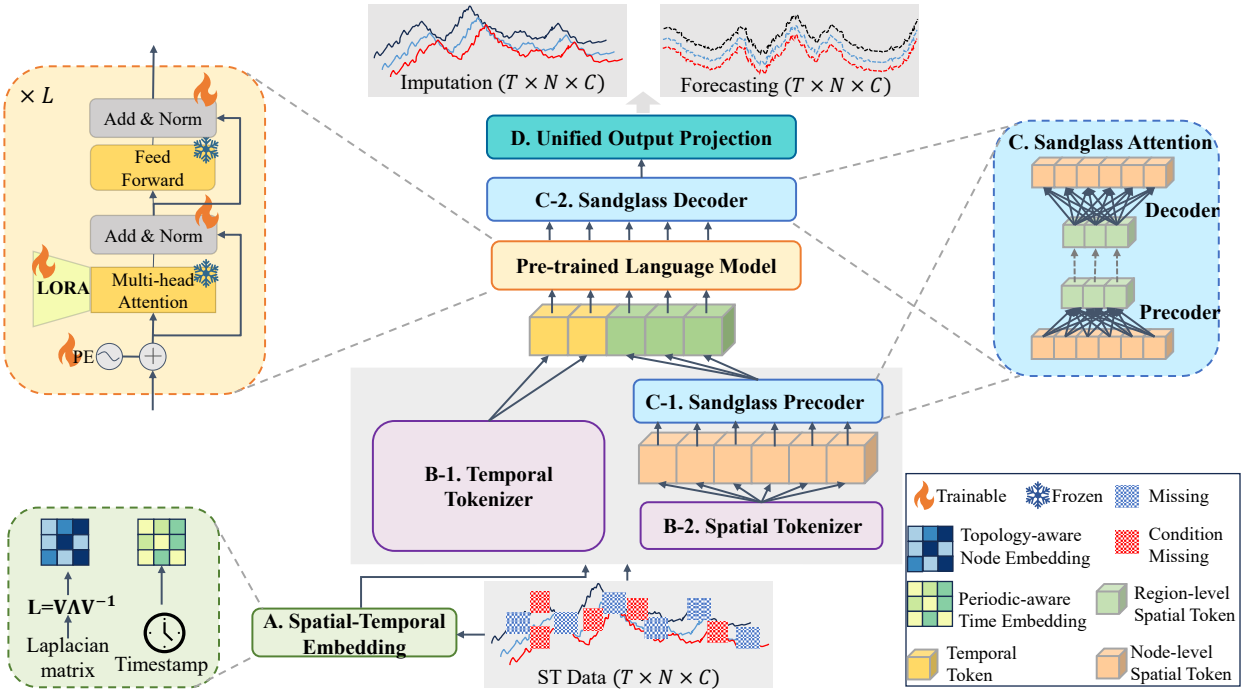

Figure 1: Model architecture. Module **A** provides spatial-temporal embeddings. The spatial-temporal tokenizers **B** construct temporal and spatial tokens from different perspectives. Module **C** builds region-level spatial tokens based on the node-level spatial tokens which is the output of **B-2**. Module **D** projects the hidden representations to target output. For the PLM, We utilize partial freezing and LoRA to fine-tune the multi-head attention, position embedding and layernorm layers.

generating a token that represents the state of each node, thereby enabling the PLM to model the complex spatial correlations between different nodes. For the state of each node, we believe it can be decomposed into a relatively static intrinsic state and a frequently changing dynamic state. The intrinsic state, determined by the node's spatial structure and periodicity, captures the macroscopic characteristics of the node. The dynamic state, derived from historical data, captures the microscopic variations of the node.

Based on the above analysis, we model the intrinsic states $\boldsymbol{Z}_{\text{intrinsic}} \in \mathbb{R}^{N \times d_{PLM}}$ using node embeddings and time embeddings, while using historical data to model the dynamic state $\boldsymbol{Z}_{\text{dynamic}} \in \mathbb{R}^{N \times d_{PLM}}$. To unify the spatial-temporal prediction and imputation tasks, enabling the PLM to detect the patterns of missing is essential. Thus, we further construct mask tokens $\boldsymbol{Z}_{\text{mask}} \in \mathbb{R}^{N \times d_{PLM}}$ based on the mask matrix $\boldsymbol{\mathcal{M}}$. By combining the three elements, we obtain the target node-level spatial tokens $\boldsymbol{Z}_S \in \mathbb{R}^{N \times d_{PLM}}$. Formally,

$$\begin{aligned} &\boldsymbol{Z}_{\text{intrinsic}} = \text{MLP}([\mathbf{E}_T || \mathbf{E}_N]), \boldsymbol{Z}_{\text{dynamic}} = \text{MLP}(\mathcal{X}), \\ &\boldsymbol{Z}_{\text{mask}} = \text{MLP}(\boldsymbol{\mathcal{M}}), \\ &\boldsymbol{Z}_S = \text{LayerNorm}(\boldsymbol{Z}_{\text{dynamic}} + \boldsymbol{Z}_{\text{intrinsic}} + \boldsymbol{Z}_{\text{mask}}). \end{aligned} \tag{3}$$

In this process, we integrate the temporal dimension into the feature dimension. The MLP consists of two layers of linear with a ReLU activation function in between. $||$ denotes concatenation along the feature dimension. d_{PLM} is the token dimension of the PLM.

Temporal Tokenizer: Relying solely on spatial tokens can effectively model the spatial-temporal relationships between nodes, but the model may lack an understanding of the overall state and changing trends of the system. Therefore, we design temporal tokenizer that aggregates information from all the nodes for each time step to encapsulate the overall state and changing trends. Considering that a single time step cannot capture the sequence's locality and trends, which are essential pieces of information, we draw on the concept from PatchTST (Nie et al. 2023), merging all time steps into a single patch. Consequently, only one overall state token $\boldsymbol{Z}_{\text{state}} \in \mathbb{R}^{1 \times d_{PLM}}$ and one overall trend token $\boldsymbol{Z}_{\text{trend}} \in \mathbb{R}^{1 \times d_{PLM}}$ are produced, rather than generating separate tokens for each time step.

Specifically, we first take the mean of all node to obtain the overall average state $\overline{\boldsymbol{\mathcal{X}}} \in \mathbb{R}^{1 \times (T \times C)}$. Then, we use the first difference of $\overline{\boldsymbol{\mathcal{X}}}$, denoted as $\overline{\boldsymbol{\mathcal{X}}}_{\text{trend}} \in \mathbb{R}^{1 \times ((T-1) \times C)}$, to represent the overall trend. Finally, $\overline{\boldsymbol{\mathcal{X}}}$ and $\overline{\boldsymbol{\mathcal{X}}}_{\text{trend}}$ are respectively combined with the time embeddings. A MLP is further utilized to perform feature transformation, resulting in $\boldsymbol{Z}_{\text{state}}$and $\boldsymbol{Z}_{\text{trend}}$. Then we concatenate these two tokens and apply Layer Normalization to normalize them, thereby obtaining the temporal tokens $\boldsymbol{Z}_T \in \mathbb{R}^{2 \times d_{PLM}}$. The above process is defined as follows:

$$\begin{aligned} &\boldsymbol{Z}_{\text{state}} = \text{MLP}(\overline{\boldsymbol{\mathcal{X}}} || \mathbf{E}_T[T-1:, 0, :]), \\ &\boldsymbol{Z}_{\text{trend}} = \text{MLP}(\overline{\boldsymbol{\mathcal{X}}}_{\text{trend}} || \mathbf{E}_T[T-1:, 0, :]), \\ &\boldsymbol{Z}_T = \text{LayerNorm}([\boldsymbol{Z}_{\text{state}} || \boldsymbol{Z}_{\text{trend}}]), \end{aligned} \tag{4}$$

where the $\mathbf{E}_T[T-1:,0,:] \in \mathbb{R}^{1\times d_t}$ represents the time embedding of the last time step in the input.

Sandglass Attention. Taking into account the following two factors, we have developed a sandglass attention module. 1) The number of spatial tokens N is usually large in real-world applications which significantly hampers model efficiency especially the training and inference speed. 2) The spatial tokens primarily encapsulate information at the node level. However, these node-level spatial tokens alone struggle to capture important non-pairwise and higher-order spatial-temporal correlations. Specifically, the proposed sandglass attention (SGA) module first aggregates node-level spatial tokens into fewer region-level spatial tokens thoughr a precoder, enabling the model to capture non-pairwise and higher-order spatial-temporal correlations while enhancing computational efficiency. Then a decoder is employed to restore the original length of spatial tokens. Next, we detail the SGA module.

In the SGA precoder, we aggregate the node-level spatial tokens $\boldsymbol{Z}_S$ into region-level spatial tokens $\boldsymbol{Z}_H \in \mathbb{R}^{M\times d_{LLM}}$ through a learnable query matrix $\mathbf{H}_l \in \mathbb{R}^{M\times d_{PLM}}$:

$$\boldsymbol{Z}_H = \text{LayerNorm}((\text{Attention}(\mathbf{H}_l, \boldsymbol{Z}_S, \boldsymbol{Z}_S)). \tag{5}$$

where M represents the number of region-level spatial tokens, and $M < N$. Besides, the scaled dot-product-attention is employed in our paper. Formally,

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}})\mathbf{V}, \tag{6}$$

We then feed $\boldsymbol{Z}_H$ into PLM to further capture the spatial-temporal correlations between the region-level spatial tokens and temporal tokens. Finally, we get the corresponding $\boldsymbol{Z}_H^{'} \in \mathbb{R}^{M\times d_{PLM}}$. To fit with the output size, it is necessary to map the region-level hidden representation back to the node-level by a SGA decoder:

$$\boldsymbol{Z}_N^{'} = \text{LayerNorm}((\text{Attention}(\boldsymbol{Z}_S, \mathbf{H}_l, \boldsymbol{Z}_H^{'})), \tag{7}$$

where $\boldsymbol{Z}_N^{'} \in \mathbb{R}^{N\times d_{PLM}}$ refers to the recovered node-level hidden representation from the PLM.

Unified Output Projection. Given that both forecasting and imputation tasks fundamentally involve understanding the complex spatial-temporal dependencies of data, and that the PLM has completed a substantial amount of imputation tasks during pre-train phase, we expect our model, which is designed based on PLM, can handle both of the forecasting and imputation tasks. Thus, we need to enable our model to recognize the presence of missing values. In the Eq. 3, we have already enhanced the model's perception of missing values through the mask tokens. Thus, in the output layer, we unify these two tasks through the following projection:

$$\begin{aligned} &[\boldsymbol{Z}_T^{'}, \boldsymbol{Z}_H^{'}] = \text{PLM}([\boldsymbol{Z}_T || \boldsymbol{Z}_H]), \quad \boldsymbol{Z}_N^{'} \leftarrow \boldsymbol{Z}_H^{'}, \\ &\mathbf{Y} = \text{MLP}(\boldsymbol{Z}_N^{'} + \boldsymbol{Z}_T^{'}[1:,:] + \boldsymbol{Z}_S), \end{aligned} \tag{8}$$

where $\mathbf{Y} \in \mathbb{R}^{N\times(T\times C)}$ is the forecasting or imputation result. $\boldsymbol{Z}_T^{'} \in \mathbb{R}^{2\times d_{PLM}}$ is the hidden representations of temporal tokens. After constructing the tokens and inputting them into the PLM, we obtain the hidden representations. Then, we sum the hidden representation of ecah node with the hidden representation of the overall trend temporal token, and make a residual connection with the node-level spatial tokens to obtain the hidden representation of the target state. Finally, a MLP is used to map the target state hidden representations to our target outputs.

Model Training. Given that the PLM has only been exposed to textual data with a significantly different distribution during the pre-training phase and lacks knowledge related to spatial-temporal data, it is necessary to fine-tune the PLM to acquire knowledge of spatial-temporal data and adapt to the distribution patterns of such data. Furthermore, considering that the SGA module needs to learn a query matrix from scratch, this process may lead to overfitting and confusion in spatial correlation if no constraints such as the adjacency matrix are introduced, resulting in a decrease in model performance. Therefore, we designed a constrained loss function to guide and constrain the training of the SGA module. The specific design is detailed in the following text.

Fine-tuning PLM: Based on some existing studies (Liu et al. 2024a; Zhou et al. 2024), we realized that the knowledge of the PLM is primarily stored in the attention modules, while layer normalization and position embeddings are sensitive to the distribution of the data. Therefore, we choose to fine-tune the attention layers to embed spatial-temporal knowledge into the PLM, while also fine-tuning the position embeddings and layer normalization to adapt the PLM to the distribution of spatial-temporal data. To reduce the number of parameters needing updates, we apply Low-Rank Adaptation (LoRA) (Hu et al. 2021) to the attention layers. Meanwhile, the position embeddings and layer normalization, with fewer parameters, are fully updated.

Loss Function: We use the L1 loss between the model's output Y and the ground truth to train our model. Utilizing only this loss function can achieve satisfactory results on the training set. However, to enable the model to make more effective use of the graph structure and to mitigate the phenomenon of overfitting, further design is required. Specifically, to ensure that in the SGA module, the learned region-level spatial nodes can reflect the original graph structure G, we design *a structure-aware loss function* $\mathcal{L}_G$ when aggregating node-level spatial tokens. Furthermore, to prevent overfitting that might cause some node information to be ignored (attention weights are close to zero), we introduce *a regularization term* $\mathcal{L}_R$. The sum of these two components forms constraint loss function $\mathcal{L}_C = \mathcal{L}_G + \mathcal{L}_R$. Assuming $S \in \mathbb{R}^{M\times N}$ represents the attention weights between each region-level spatial token and node-level spatial token, $\mathcal{L}_G$ and $\mathcal{L}_R$ can be expressed as:

$$\begin{aligned} \mathcal{L}_G &= -\sum_{m=1}^{M}\sum_{i,j(i\neq j)} S_{m,i}S_{m,j}A_{i,j}, \\ \mathcal{L}_R &= -\log\pi(\text{Softmax}(\sum_{m=1}^{M} S_{m,:})|\boldsymbol{\alpha}), \end{aligned} \tag{9}$$

where π is the Dirichlet distribution. $\boldsymbol{\alpha} \in \mathbb{R}^N$ is the parameter of the Dirichlet distribution. $S_{m,:} \in \mathbb{R}^N$ denotes the vec-

tor of attention weights for the m-th region token across all node tokens, which satisfies the properties: $\sum_{i=1}^{N} S_{m,i} = 1$.

It is clear that the more edges there are, the smaller the minimum value of L_G can be. Consider a complete graph of size P, where:

$$\begin{aligned}\mathcal{L}_G &= -\sum_{m=1}^{M} \sum_{i,j(i\neq j)} S_{m,i} S_{m,j} \cdot 1, \\ &= -\sum_{m=1}^{M} \sum_{i} S_{m,i}(1 - S_{m,i}) = -M + \|S \odot S\|_1.\end{aligned} \tag{10}$$

The $\odot$ denotes the Hadamard Product. It is evident that minimizing the objective function $\mathcal{L}_G$ is achieved by evenly distributing the values of S, such that $S_{m,i} = \frac{1}{P}$. Under this condition, $\mathcal{L}_G$ simplifies to $-\frac{M(P-1)}{P}$. Shifting the discussion to a regular graph of order N, a lower $\mathcal{L}_G$ facilitates the SGA module's ability to concentrate on larger complete subgraphs or those with robust connectivity, effectively integrating the graph's structural properties into the model.

The essence of $\mathcal{L}_R$ is simple: the expected value of π is $\boldsymbol{\alpha}/\sum_i \boldsymbol{\alpha}_i$, indicating that a larger $-\mathcal{L}_R$ drives $\text{Softmax}(\sum_{m=1}^{M} S_{m,:})$ closer to this expected value. By tuning $\boldsymbol{\alpha}$, we indirectly control the attention weights distribution. Setting $\boldsymbol{\alpha} = \{1.05\}^N$ (a small value greater than 1) ensures equal aggregation of nodes. Yet, to prioritize significant nodes, we refine $\boldsymbol{\alpha}$ by adding the node degrees: $\boldsymbol{\alpha} = \{1.05\}^N + \text{Softmax}(\text{diag}(D))$.

Experiments

Datasets

The experiments conducted on four traffic datasets (PEMS03, PEMS04, PEMS07, PEMS08) (Chen et al. 2001).

To assess the model's imputation capability, we generated two types of missing data patterns, RM (random missing) and CM (spatial-temporal continuity missing), on the PEMS08 dataset, each with a 70% missing rate. Additionally, following the self-supervised training approach of CSDI (Tashiro et al. 2021), we generated condition missing for training. Condition missing refers to the artificially created missing data, used to generate training samples.

Experimental Settings

We follow the approach of ASTGCN (Guo et al. 2019) by partitioning the dataset into training set, validation set, and testing set with a ratio of 6: 2: 2, and employ a sliding window of size 12 to construct samples that forecast the next 12 steps based on the historical 12 steps.

To comprehensively compare the performance of models, we selected nine representative baselines for the forecasting task, including LSTM (Hochreiter and Schmidhuber 1997), ASTGCN (Guo et al. 2019), AGCRN (Bai et al. 2020), SSTBAN (Guo et al. 2023), PDFormer (Jiang et al. 2023), iTransformer (Liu et al. 2023b), OFA (Zhou et al. 2024), STLLM (Liu et al. 2024a) and STGLLM (Liu et al. 2024b). For the imputation task, we also chose four well-known baselines, which are Brits (Cao et al. 2018), E2GAN (Luo et al. 2019), mTAN (Shukla and Marlin 2021), and PriSTI (Liu et al. 2023a).

The experiments were conducted on NVIDIA A40 GPUs with a PyTorch version of 2.1.2. Using an AdaW optimizer with a learning rate of 1×10^{-3}. The training epoch is 500 with a 50 epochs early stopping mechanism. We use GPT-2 (Radford et al. 2019) as the PLM and utilized only its initial three layers. STD-PLM employs the hyperparameters identified through tuning on the PEMS08 validation set for all datasets. The detail of experiments settings are depicted in supplementary material.

Overall Performance

We measure the model's performance using three common regression metrics: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). Table 1 compares the forecasting performance of STD-PLM with baselines, where bold indicates the best results and underline denotes the second-best. From Table 1, we can observe that: 1) Time series models like LSTM, iTransformer, and OFA underperform compared to methods that capture spatial-temporal correlations, indicating that temporal correlation alone is insufficient for high-precision forecasting; 2) The results of PLM-based models (OFA, STGLLM, STLLM) highlight the necessity of our design, as simply using the PLM to process spatial-temporal data does not lead to optimal performance. 3) STD-PLM achieve the best or second-best results on the all datasets.

Table 2 compares imputation performance, showing : 1) Our method achieves state-of-the-art performance. 2) STD-PLM exhibits a substantial improvement over other baseline models in the RM 70% case. This implies that STD-PLM can fully harness the imputation capabilities of the PLM for spatial-temporal data.

In summary, the experimental results demonstrate that STD-PLM has strong performance and can achieve high accuracy in both forecasting and imputation tasks.

Few-shot and Zero-shot Performance

To extensively evaluate the performance of STD-PLM, we conduct few-shot and zero-shot experiments. The few-shot experiments evaluate the forecasting performance of STD-PLM using only the first 5%, 10%, and 20% of the training samples, with results summarized in Table 5. The zero-shot experiments evaluate the forecasting performance of the trained model when applied to other datasets, with the results compiled in Table 4.

Table 5 shows that using just 5% of training data, STD-PLM matches LSTM trained on the full set, and with 20%, it outperforms ASTGCN. This highlights its strong few-shot learning capability for data-scarce scenarios. Table 4 further demonstrates that STD-PLM maintains acceptable performance when directly transferred to datasets with different temporal scopes and graph structures from the training set, without undergoing any training.

Ablation Study

We conduct an ablation study to assess the effectiveness of each component. The experiments compare our model with

Model	PEMS03			PEMS04			PEMS07			PEMS08		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
LSTM	20.62	33.54	28.94%	26.81	40.74	22.33%	29.71	45.32	14.14%	22.19	33.59	18.74%
ASTGCN	17.85	29.88	17.65%	22.42	34.75	15.87%	25.98	39.65	11.84%	18.86	28.55	12.50%
AGCRN	15.98	28.25	15.23%	19.83	32.26	12.97%	22.37	36.55	9.12%	15.95	25.22	10.09%
SSTBAN	15.90	26.11	17.27%	18.88	31.10	12.56%	20.17	33.45	8.93%	14.39	24.19	10.06%
PDFormer	14.74	25.59	15.35%	18.31	**29.97**	12.10%	19.83	32.87	8.53%	13.58	23.51	9.05%
iTransformer	19.48	31.20	17.84%	22.56	35.21	16.29%	24.70	37.91	11.40%	20.05	31.90	11.99%
OFA	20.96	33.43	19.11%	27.37	42.99	17.97%	30.53	47.51	12.98%	21.89	34.63	13.30%
STGLLM	15.26	**24.11**	15.73%	20.00	32.11	13.69%	21.98	35.02	9.72%	15.53	24.74	10.15%
STLLM	17.25	27.25	22.96%	19.00	30.35	13.55%	21.48	34.07	10.20%	14.67	23.50	10.63%
STD-PLM	**14.59**	25.36	**14.92%**	**18.16**	30.21	**11.89%**	**19.25**	**32.84**	**8.06%**	**13.31**	**23.19**	**8.84%**

Table 1: Forecasting performance.

Model	RM 70%			CM 70%		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
BRITS	29.56	41.52	10.79%	38.33	53.60	13.90%
E2GAN	27.55	41.99	17.52%	31.58	50.24	19.17%
mTAN	21.23	33.98	12.89%	31.42	49.10	18.96%
PriSTI	25.54	36.00	16.73%	22.93	40.90	14.43%
STD-PLM	**14.36**	**23.20**	**9.58%**	**22.69**	**39.66**	**13.82%**

Table 2: Imputation performance on PEMS08.

Method	PEMS08			PEMS08 CM 70%		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
w/o TT	13.81	23.52	9.03%	22.82	**39.50**	14.35%
w/o $\mathcal{L}_C$	13.53	23.69	8.97%	23.62	40.05	14.65%
w/o PLM	13.63	23.39	9.15%	24.05	41.15	15.34%
STD-PLM	**13.31**	**23.19**	**8.84%**	**22.69**	39.66	**13.82%**

Table 3: Ablation study.

	MAE	**RMSE**	**MAPE**
PEMS04→PEMS08	29.52	45.63	22.92%
PEMS08→PEMS04	25.32	37.80	25.32%
PEMS07→PEMS03	23.72	36.94	41.59%
PEMS03→PEMS07	34.72	52.66	20.31%

Table 4: Zero-shot performance.

Ratio	PEMS04			PEMS08		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
5%	27.73	42.12	20.62%	22.56	34.35	16.91%
10%	25.11	38.68	17.65%	19.72	31.35	13.27%
20%	21.16	34.03	13.92%	16.58	27.03	10.78%

Table 5: Few-shot performance.

Method	PEMS03		PEMS07	
	Time	Memory	Time	Memory
w SGA	7.40	8554	9.15	15020
w/o SGA	17.96	15366	52.82	29718

Table 6: inference consumption (batch size=32).

the following variants: 1) w/o TT: removing temporal tokens; 2) w/o $\mathcal{L}_C$: removing constraint loss $\mathcal{L}_C$; 3) w/o PLM: replacing the PLM with a transformer encoder of the same layer and hidden dimension.

Table 3 summarizes the results, revealing that: 1) Temporal Tokens and $\mathcal{L}_C$ improve forecasting and imputation with small computational cost. 2) Fine-tuning the PLM outperforms a similarly scaled transformer encoder trained from scratch, indicating the PLM's embedded knowledge benefits spatial-temporal tasks.

We further assess SGA's efficiency impact in Table 6, recording inference time (s) and GPU memory (MiB) usage on the test set, showing SGA substantially improves efficiency.

Conclusion

In this paper, we propose STD-PLM, a unified framework for spatial-temporal forecasting and imputation using PLM. By employing explicitly designed spatial-temporal tokenizers and embeddings, STD-PLM effectively captures spatial and temporal patterns. Additionally, the SGA module reduces computational costs by generating region-level spatial tokens. Extensive experiments show competitive performance, highlighting the potential of a unified pre-trained spatial-temporal model based on PLM.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62202043) and the Beijing Natural Science Foundation (Grant No. 4242029), and Sponsored by CCF-Zhipu AI Large Model Fund.

References

Bai, L.; Yao, L.; Kanhere, S. S.; Wang, X.; and Sheng, Q. Z. 2019. STG2seq: spatial-temporal graph to sequence model for multi-step passenger demand forecasting. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 1981–1987.

Bai, L.; Yao, L.; Li, C.; Wang, X.; and Wang, C. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems*, 33: 17804–17815.

Cao, D.; Jia, F.; Arik, S. O.; Pfister, T.; Zheng, Y.; Ye, W.; and Liu, Y. 2024. TEMPO: Prompt-based Generative Pre-trained Transformer for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations*.

Cao, W.; Wang, D.; Li, J.; Zhou, H.; Li, L.; and Li, Y. 2018. Brits: Bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31.

Chen, C.; Petty, K.; Skabardonis, A.; Varaiya, P.; and Jia, Z. 2001. Freeway performance measurement system: mining loop detector data. *Transportation research record*, 1748: 96–102.

Chen, X.; Lei, M.; Saunier, N.; and Sun, L. 2021. Low-rank autoregressive tensor completion for spatiotemporal traffic data imputation. *IEEE Transactions on Intelligent Transportation Systems*, 23(8): 12301–12310.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Ding, A.; Zhao, X.; and Jiao, L. 2002. Traffic flow time series prediction based on statistics learning theory. In *Proceedings. The IEEE 5th International Conference on Intelligent Transportation Systems*, 727–730. IEEE.

Ge, Y.; Hua, W.; Mei, K.; Tan, J.; Xu, S.; Li, Z.; Zhang, Y.; et al. 2024. Openagi: When llm meets domain experts. *Advances in Neural Information Processing Systems*, 36.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2020. Generative adversarial networks. *Communications of the ACM*, 63(11): 139–144.

Guo, S.; Lin, Y.; Feng, N.; Song, C.; and Wan, H. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 922–929.

Guo, S.; Lin, Y.; Gong, L.; Wang, C.; Zhou, Z.; Shen, Z.; Huang, Y.; and Wan, H. 2023. Self-supervised spatial-temporal bottleneck attentive network for efficient long-term traffic forecasting. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, 1585–1596. IEEE.

Guo, S.; Lin, Y.; Wan, H.; Li, X.; and Cong, G. 2021. Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 34(11): 5415–5428.

Han, H.; Zhang, M.; Hou, M.; Zhang, F.; Wang, Z.; Chen, E.; Wang, H.; Ma, J.; and Liu, Q. 2020. STGCN: a spatial-temporal aware graph learning method for POI recommendation. In *2020 IEEE International Conference on Data Mining (ICDM)*, 1052–1057. IEEE.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33: 6840–6851.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Hu, E. J.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2021. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Jiang, J.; Han, C.; Zhao, W. X.; and Wang, J. 2023. Pdformer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 4365–4373.

Jin, M.; Wang, S.; Ma, L.; Chu, Z.; Zhang, J. Y.; Shi, X.; Chen, P.-Y.; Liang, Y.; Li, Y.-F.; Pan, S.; et al. 2023. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*.

Kingma, D. P.; and Welling, M. 2014. Auto-Encoding Variational Bayes. *stat*, 1050: 1.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*.

Lin, Z.; Feng, J.; Lu, Z.; Li, Y.; and Jin, D. 2019. Deepstn+: Context-aware spatial-temporal neural network for crowd flow prediction in metropolis. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 1020–1027.

Liu, C.; Yang, S.; Xu, Q.; Li, Z.; Long, C.; Li, Z.; and Zhao, R. 2024a. Spatial-temporal large language model for traffic prediction. *arXiv preprint arXiv:2401.10134*.

Liu, L.; Yu, S.; Wang, R.; Ma, Z.; and Shen, Y. 2024b. How can large language models understand spatial-temporal data? *arXiv preprint arXiv:2401.14192*.

Liu, M.; Huang, H.; Feng, H.; Sun, L.; Du, B.; and Fu, Y. 2023a. PriSTI: A Conditional Diffusion Framework for Spatiotemporal Imputation. *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, 1927–1939.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2023b. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations*.

Luo, Y.; Zhang, Y.; Cai, X.; and Yuan, X. 2019. E2gan: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of the 28th international joint conference on artificial intelligence*, 3094–3100. AAAI Press Palo Alto, CA, USA.

Mazumder, R.; Hastie, T.; and Tibshirani, R. 2010. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11: 2287–2322.

Miao, H.; Shen, J.; Cao, J.; Xia, J.; and Wang, S. 2022. MBA-STNet: Bayes-enhanced discriminative multi-task learning for flow prediction. *IEEE Transactions on Knowledge and Data Engineering*, 35(7): 7164–7177.

Miao, H.; Zhao, Y.; Guo, C.; Yang, B.; Zheng, K.; Huang, F.; Xie, J.; and Jensen, C. S. 2024. A unified replay-based continuous learning framework for spatio-temporal prediction on streaming data. *arXiv preprint arXiv:2404.14999*.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *The Eleventh International Conference on Learning Representations*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Shi, X.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.-K.; and Woo, W.-c. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28.

Shukla, S. N.; and Marlin, B. M. 2021. Multi-time attention networks for irregularly sampled time series. *arXiv preprint arXiv:2101.10318*.

Song, C.; Lin, Y.; Guo, S.; and Wan, H. 2020. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 914–921.

Song, Q.; Li, D.; and Li, X. 2017. Traffic prediction based route planning in urban road networks. In *2017 Chinese Automation Congress (CAC)*, 5854–5858. IEEE.

Tashiro, Y.; Song, J.; Song, Y.; and Ermon, S. 2021. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34: 24804–24816.

Wang, W.; Zheng, V. W.; Yu, H.; and Miao, C. 2019. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10: 1–37.

Wang, Y.; Yao, Q.; Kwok, J. T.; and Ni, L. M. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53: 1–34.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2019. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*.

Yao, H.; Wu, F.; Ke, J.; Tang, X.; Jia, Y.; Lu, S.; Gong, P.; Ye, J.; and Li, Z. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Yoon, J.; Jordon, J.; and Schaar, M. 2018. Gain: Missing data imputation using generative adversarial nets. In *International conference on machine learning*, 5689–5698. PMLR.

Yu, H.-F.; Rao, N.; and Dhillon, I. S. 2016. Temporal regularized matrix factorization for high-dimensional time series prediction. *Advances in neural information processing systems*, 29.

Zhang, J.; Zheng, Y.; and Qi, D. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Zhang, Q.; Wang, H.; Long, C.; Su, L.; He, X.; Chang, J.; Wu, T.; Yin, H.; Yiu, S.-M.; Tian, Q.; et al. 2024. A Survey of Generative Techniques for Spatial-Temporal Data Mining. *arXiv preprint arXiv:2405.09592*.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*.

Zheng, C.; Fan, X.; Wang, C.; and Qi, J. 2020. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 1234–1241.

Zhou, T.; Niu, P.; Sun, L.; Jin, R.; et al. 2024. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36.

Zonoozi, A.; Kim, J.-j.; Li, X.-L.; and Cong, G. 2018. Periodic-CRN: A convolutional recurrent model for crowd density prediction with recurring periodic patterns. In *IJCAI*, volume 18, 3732–3738.