Scaling up Multivariate Time Series Pre-Training with Decoupled Spatial-Temporal Representations

Rui Zha*, Le Zhang†, Shuangli Li*, Jingbo Zhou†, Tong Xu*, Hui Xiong‡, Enhong Chen*
*School of Computer Science, University of Science and Technology of China
†Business Intelligence Lab, Baidu Research
‡The Hong Kong University of Science and Technology (Guangzhou)
{crui0210,zhangle0202,lishuangli0209}@gmail.com, zhoujingbo@baidu.com,
{tongxu,cheneh}@ustc.edu.cn, xionghui@ust.hk

Abstract—**Data scale has been acknowledged as a crucial factor for enhancing the generalization and effectiveness of pre-training models. While existing methods of multivariate time series pre-training are primarily limited to a single specific dataset, scaling to a larger scenario that includes multiple diverse datasets (e.g., multi-region data) remains a substantial challenge. In this paper, we present a novel Decoupled Spatial-Temporal Representation Learning (DeSTR) framework to serve as the backbone network for investigating the data scaling capability of multivariate time series pre-training architectures. Specifically, DeSTR utilizes two separate encoders to capture both the temporal dynamics within each time series and the spatial correlations among multiple variables. The obtained representations of distinct modalities are then fed into a Spatial-Guided Temporal Transformer to equip the temporal features with spatial discriminative information. Moreover, we employ masked autoencoding as the foundational pre-training framework and introduce spacetime-agnostic augmentation to improve robustness and facilitate implicit spatiotemporal modeling. Finally, we successfully pre-train a unified time series representation learning framework on real-world datasets from three different cities. Extensive experiments are carried out on various downstream tasks to validate the performance of DeSTR, compared with three categories of state-of-the-art baselines: deep sequential models, spatial-temporal graph neural networks, and time series representation learning methods. The results clearly demonstrate the advantages of scaling multivariate time series pre-training to multiple datasets, highlighting the effectiveness of DeSTR as a general spatiotemporal learner.**

Index Terms—**multivariate time series pre-training, spatiotemporal data, data scalability**

I. Introduction

Self-supervised representation learning has emerged as a highly effective paradigm in learning generic representations for multiple data modalities, such as text [1], [2], vision [3]–[5], and audio [6], [7]. These foundational models, once pre-trained, can be easily adapted to a wide range of downstream tasks through fine-tuning, resulting in a substantial performance boost compared to those task-specific crafted architectures [1], [3].

For multivariate time series, numerous efforts have been devoted to developing effective pre-training models. Among them, the mainstream frameworks include methods based on contrastive learning [8]–[10] and masked autoencoding [11], [12]. Despite their encouraging results, a significant limitation of these methods is the necessity to pre-train individual models tailored for each dataset. In multi-region spatiotemporal scenarios, this will require separate pre-training models for each region even for the same application, leading to significant training overhead. Meanwhile, the compelling evidence from the emerging Large Language Models (LLM) [1], [2] has demonstrated the crucial role of scaling data size for achieving remarkable performance improvements. In this context, pre-training solely on data from a single geographic region fails to fully exploit the benefit of data scale, leading to suboptimal generalization capability.

Building upon the promising findings in recent LLM, we aim to *investigate the data scaling capability of multivariate time series pre-training models*. The goal is to pre-train a unified representation learning framework by harnessing multi-region spatiotemporal data. This framework can subsequently be adapted to various downstream tasks in each specific region through fine-tuning. In particular, we attempt to address this problem by discussing the following two most-adopted backbone networks for multivariate time series analysis over spatiotemporal data:

(i) Using *Deep Sequential Models* to extract high-dimensional representations has been widely adopted in time series analysis. However, their application to real-world situations still presents two main challenges. First, practical data often encounter noise from irregular missing values, potentially arising from non-synchronous measurements or errors during data collection [13]. This missing pattern substantially undermines the robustness of sequential models, preventing them from deriving comprehensive time series representations. Second, despite their effectiveness in capturing temporal dynamics, these sequential models often overlook the crucial aspect of spatial correlations. As depicted in Figure 1a, conventional sequential models process each univariate time series independently, lacking the capability to differentiate diverse variables. In scenarios involving large-scale datasets with extensive variables, these models tend to learn generic knowledge, resulting in a decline in prediction accuracy.

(ii) As an alternative, *Spatial-Temporal Graph Neural Networks* (STGNNs) have attracted significant attention for integrating spatial information into time series analysis. By utilizing Graph Neural Networks (GNNs), these models adeptly

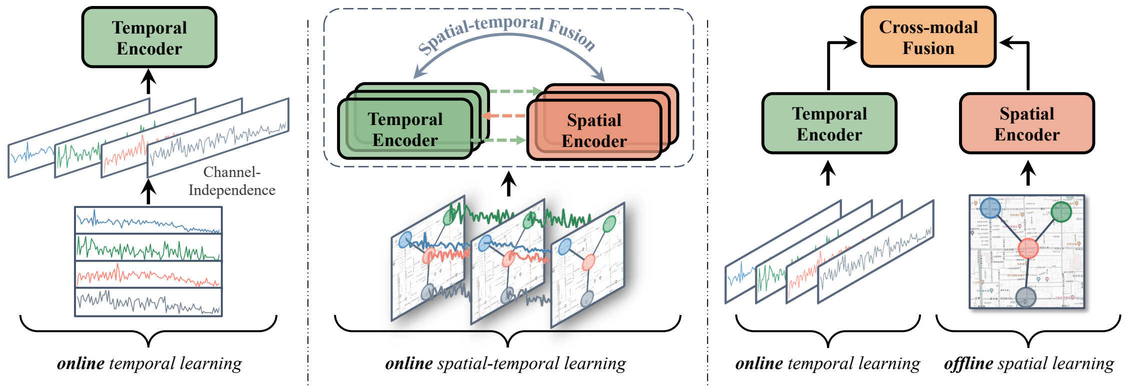


Fig. 1: Concept comparison of Deep Sequential Model, STGNNs, and our DeSTR paradigms.

capture spatial correlations, offering enhanced discriminative ability across different variables and harnessing valid neighboring information to mitigate noise induced by missing data. Nevertheless, these models heavily depend on explicit online message passing. As depicted in Figure 1b, for any batch of input time series at any timestep, each node has to exchange signal information with adjacent nodes. This process entails two critical limitations: First, deep-layer STGNNs bear a high computational cost due to the inherent *neighborhood explosion* problem associated with GNNs [14]. Even shallow STGNNs, which must repeatedly propagate information at each timestep in an online setting, remain time-intensive. This complexity poses challenges in meeting low-latency inference requirements for time-sensitive applications, such as traffic flow forecasting. Second, achieving real-time communications between different nodes is infeasible due to hardware constraints, especially when handling geographically dispersed data. This constraint hinders the efficient implementation of STGNNs in time-critical situations.

Driven by the above discussions, we introduce a novel backbone network based on decoupled spatial-temporal representations, eliminating the need for online message passing across nodes. As depicted in Figure 1c, we first employ a dual encoder composed of separate convolution networks to distill both temporal dynamics and spatial correlations into two distinct dense vector embeddings. These representations of different modalities are then fed into a transformer-like module, namely the Spatial-Guided Temporal Transformer, equipping each independent time series with its corresponding spatial discriminative information.

Contrasting with traditional decoupled spatial-temporal approaches, our backbone architecture is distinguished by two novel features. Firstly, since spatial learning relies solely on the static graph structure and is unrelated to online time series, our proposed backbone can be effectively scaled to handle larger datasets covering multiple regions. Secondly, during the online deployment, we can directly use the pre-trained static node embeddings to serve as side information for online time series processing. This procedure obviates the need for inter-node message passing in existing methods, thereby resulting in enhanced computational efficiency.

To further enhance the robustness of the pre-trained encoder against missing data, we employ masked autoencoding as the foundational framework for learning generic time series representations. Specifically, we introduce a spacetime-agnostic augmentation for multivariate time series, randomly masking a subset of valid values across both spatial and temporal dimensions. These corrupted inputs are processed through the aforementioned backbone network, followed by a lightweight decoder to reconstruct the original time series. The objective of this pre-training phase is to teach the model to undo these degradations. In summary, the main contributions of this paper are outlined as follows:

- We are among the first to explore the data scalability in multivariate time series pre-training, focusing on multi-regional spatiotemporal datasets.
- We introduce a novel backbone network for streamlined spatiotemporal data processing, facilitating operations on large-scale datasets with remarkable computational efficiency.
- We present a unified pre-training architecture based on masked autoencoding, supporting robust pre-training on practical multi-region spatiotemporal datasets.
- We perform extensive experiments on large-scale real-world datasets spanning multiple cities, and the results strongly demonstrate the benefits of scaling data size for multivariate time series pre-training.

II. Related Work

Multivariate Time Series Forecasting. Multivariate Time Series (MTS) forecasting has been a critical task across various domains, ranging from transportation to environmental science [15], [16]. Conventional deep learning methods mainly utilize Convolution Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) to enhance MTS prediction efficacy. Among these, methods such as TCN [17] and its variants treat time series as 1D structures and employ convolution kernels to distill high-level features. Conversely, architectures like LSTM [18] and GRU [19] embrace an autoregressive framework for forecasting. With the rapid growth of Transformer [20], recent methods like Informer [21] and Autoformer [22] exploit its capability for handling long-term dependencies in MTS forecasting. Nonetheless, our study focuses on the spatial-temporal data which prioritizes short-term time series for real-time forecasting needs.

Building upon this, Spatial-Temporal Graph Neural Networks (STGNNs) have recently gained substantial attention [23]–[30] to improve the prediction by incorporating inter-variant correlations. These methods identify temporal patterns within each node using sequential models and propagate them across neighboring nodes through GNNs, effectively capturing spatial-temporal dependencies. Based on their spatial learning strategies, these methods can be categorized into two distinct branches: The first branch utilizes static graphs based on various distance functions, such as topological distance [31] and Dynamic Time Warping (DTW) distance [26]. In contrast, the second branch posits that static graphs may infuse domain-specific biases, potentially leading to inadequate modeling of spatial correlations. Hence, they opt to dynamically and adaptively construct a latent graph, typically fashioned using pair-wise similarities from current temporal embeddings [23], [24], [27]. However, a notable limitation of these approaches is their significant reliance on the quality of temporal representations. Real-world noise, such as missing data, can compromise these temporal representations, subsequently undermining the graph structure learning process and culminating in suboptimal forecasting performance.

Multivariate Time Series Imputation. Multivariate time series imputation is another important task in time series analysis, aiming to fill in missing values from observed records. Traditional studies primarily adopt statistical methods like K-Nearest Neighbors [32], Expectation-Maximization algorithms [33], and matrix factorization [34]. Recently, deep learning architectures have surged to the forefront [35]–[38], especially the recurrent networks. For instance, BRITS [35] employs a bidirectional recurrent dynamical system for handling missing data. GAIN [36] integrates RNN with Generative Adversarial Networks (GAN) to achieve imputation in a generative manner. Additionally, the rising Transformer has also been adapted for sequence imputation, with SAITS [37] employing diagonally-masked self-attention blocks to capture both fine-grained feature correlations between timesteps.

While these methods prioritize inner-channel temporal dynamics, they tend to overlook inter-variate relationships. To bridge this gap, a subset of research employs STGNNs as the backbone network for higher-accuracy time series imputation [39]–[42]. For example, GRIN [41] extends the sequential module of BRITS with STGNN to improve the imputation performance. SPIN [42] replaces both temporal and spatial modeling in typical STGNN with attention-based architectures to mitigate the unstable learning dynamics. However, despite the effectiveness of STGNNs in boosting the performance of both downstream time series forecasting and imputation, they still suffer from high computational costs.

Time Series Representation Learning. The representation learning for multivariate time series has also gained significant attention in recent years. Existing techniques in this domain typically follow two basic paradigms: *contrastive learning* and *masked autoencoders*. For time series contrastive learning,

TABLE I: Notations Used in This Paper

Notion	Description
G	The static graph structure
h_g^i	The spatial embedding of node i
e_{ji}	The weighted geospatial distance from j-th node to the i-th node
x_s^ℓ	The signal features within a specific convolutional window at ℓ-th layer
m^ℓ	The binary mask of signal features within a specific convolutional window at ℓ-th layer
H_s^i	The tokenized signal representations of i-th node
x_t	The calendar features at timestamp t
h_t	The time embedding at timestamp t
H_t	The tokenized time embeddings
$\tilde{H}_s^i$	The tokenized signal representations which equipped with the temporal position encoding
z^i	The spatial-guided temporal representation of i-th node
$\mathcal{M}$	The spacetime-agnostic mask over input signals
$\mathcal{S}_{in}$	The ground truth of input signals
$\mathcal{S}_{out}$	The outputs of the reconstructed signals
$\mathcal{L}_{mask}$	The reconstruction loss over mask signals
$\mathcal{L}_{valid}$	The reconstruction loss over observed signals

the fundamental principle involves defining positive pairs that are encouraged to be semantically similar, and negative pairs that should be dissimilar [9], [10], [43], [44]. Early studies like SRL [43] treat random subseries within an anchor series as positive samples and subseries from other variables as negative, using a Triplet loss function for optimization. Subsequent approaches introduce various data augmentation techniques and contrastive loss designs. For example, TNC [9] utilizes a debiased contrastive objective to ensure distinction between the distributions of signals within neighborhoods and those outside. However, contrastive-based methods inevitably introduce sampling biases, which can potentially undermine the effectiveness of the learned representations. To mitigate this impact, *masked autoencoders* introduce denoising as a data augmentation strategy, allowing the backbone encoder to fully capture the data distribution through the reconstruction of masked inputs [11], [12]. For instance, TST [11] first leverages Transformer as the backbone to distill dense vectors from corrupted time series, guided by a denoising objective. PatchTST [12], another state-of-the-art model in MTS pre-training, further delves into the capabilities of Transformer for time series representation learning. Unlike these pre-training studies that customize models to individual single-region datasets, we explore a more challenging yet crucial problem: scaling time series pre-training across extensive multi-region datasets, and enabling adaptable fine-tuning for various tasks in diverse urban contexts.

III. Methodology

A. Problem Definition

Given a collection of multivariate time series instances with a look-back window $L : \{x_1, \ldots, x_N\}$, our goal is to devise two nonlinear embedding functions, f_θ and f_ϕ, to transform each variate i into a spatial representation h_g^i, and

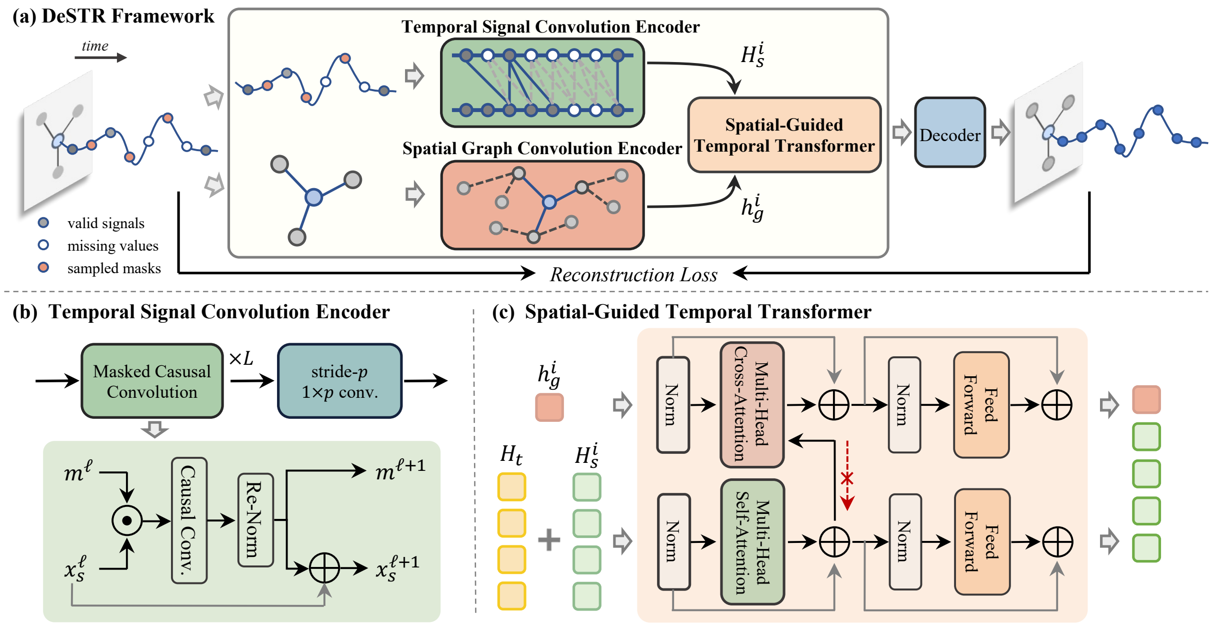

(a) DeSTR Framework
time
valid signals
missing values
sampled masks
Temporal Signal Convolution Encoder
Spatial Graph Convolution Encoder
H_s^i
h_g^i
Spatial-Guided Temporal Transformer
Decoder
Reconstruction Loss
(b) Temporal Signal Convolution Encoder
Masked Causal Convolution
×L
stride-p 1×p conv.
m^ℓ
x_s^ℓ
Causal Conv.
Re-Norm
$m^{\ell+1}$
$x_s^{\ell+1}$
(c) Spatial-Guided Temporal Transformer
h_g^i
H_t
H_s^i
Norm
Multi-Head Cross-Attention
Feed Forward
Multi-Head Self-Attention

Fig. 2: **The overview of Decoupled Spatial-Temporal Representation Learning (DeSTR) framework**.

its corresponding univariate time series x_i into a temporal representation h_s^i, respectively. These embeddings are subsequently integrated through a learnable fusion function g_ψ to generate the final d-dimensional embedding, which is expected to accurately reconstruct the input time series.

B. Model Overview

As shown in Figure 2, we propose a representation learning framework, DeSTR, for multivariate time series pre-training. The core idea is to utilize the decoupled spatial-temporal encoder as the backbone network to learn generic time series representations within a masked autoencoding paradigm. Specifically, the proposed backbone is structured based on a hybrid convolution-transformer architecture. Given the input multivariate time series, two distinct convolution encoders are employed independently: one for extracting temporal dynamics from each time series, and the other for capturing spatial correlations from the graph structure characterized by inter-variant relationships. The resulting representations in diverse modalities are then fused by the Spatial-Guided Temporal Transformer, which serves to enrich temporal features with corresponding spatial discriminative information. This backbone is further integrated into a masked autoencoding paradigm for pre-training, wherein a *spacetime-agnostic* augmentation is introduced for effective masked signal modeling. The notations used in this paper are listed in Table I.

C. Dual-Encoder for Spatiotemporal Signals

In this subsection, we provide an in-depth discussion of *how we decompose input multivariate time series into decoupled spatial-temporal representations* utilizing convolutional stems that are specialized for both spatial and temporal dimensions.
Spatial Graph Convolution Encoder. In the spatial dimension, we represent the graph structure by $G = (V, E, W)$, where edges E characterize the interconnectivity of nodes V, and edge weights W quantify the distance along with each edge. Besides, we initialize the node embedding with one-hot encoding, packed together into a feature matrix $X_g \in \mathbb{R}^{N\times N}$, where N denotes the number of nodes.

In practice, we employ the Graph Convolution Network (GCN), a typical GNN, to transform the input node features into high-level static spatial embeddings, capturing the spatial correlations effectively. Considering the hidden state of node i at layer ℓ, the message passing from its neighbors $\mathcal{N}(i)$ towards it takes the form:

$$(h_g^i)^{\ell+1} = \sigma\left(b^\ell + \sum_{j\in\mathcal{N}(i)} \frac{e_{ji}}{c_{ji}} (h_g^j)^\ell W^\ell\right) \tag{1}$$

where e_{ji} is the weighted geospatial distance from the j-th node to the i-th node, $c_{ji} = \sqrt{|\mathcal{N}(j)|}\sqrt{|\mathcal{N}(i)|}$ corresponds to the ji-th value of the normalized adjacency matrix. Moreover, $W^\ell \in \mathbb{R}^{d\times d}, b^\ell \in \mathbb{R}^d$ are learnable parameters and σ denotes the activation function.

We declare that this model design is agnostic to particular choices of graph structures and GNNs. Various alternative graphs can be utilized to delineate the inter-variant relationships, such as applying Gaussian kernels to the Euclidean distance [45] or adapting the DTW distance based on observed time series [26]. Additionally, a wide range of GNNs can also be adopted, such as GAT [46] and Graph Transformer [47]. We opt for the conventional GCN since it has fewer parameters compared to attention-based models, offering superior computational efficiency. Considering the relative stability of spatial distribution in independent graph structures, employing a simple GCN provides an optimal trade-off between accuracy and efficiency.

Temporal Signal Convolution Encoder. In the temporal dimension, we carefully design our temporal signal convolution encoder to capture temporal dynamics within each independent time series as shown in Figure 2b. Before diving into this module, we first highlight two key characteristics of the investigated signals:

(i) In real-world situations, signals frequently display *irregular missing patterns*, potentially caused by sensor malfunctions. These missing values can take the form of contiguous blocks or discrete points. In more extreme cases, an entire historical window might be devoid of data, posing significant challenges to stable and accurate feature extraction.

(ii) Natural signals typically exhibit *low information-dense with heavy temporal redundancy*. In other words, the alterations in signals between adjacent timesteps are subtle and continuous. Such continuity will be conveyed into the temporal representations extracted by the sliding convolutional kernels, making it difficult to distinguish discrete global dynamics within them.

To address the first point, we introduce a *masked causal convolution* to adjust for the irregular missing patterns. This scheme, derived from the standard causal convolution, ensures the prevention of information leakage from future to past. Considering a sliding convolution window with the kernel size of $1 \times S$ that operates on the input time series of length L. Let W and b represent the convolution weights and the corresponding bias term, respectively. For any particular timestep, let $x_s^\ell \in \mathbb{R}^{1\times S}$ denote the localized signal within the current window, while $m^\ell \in \mathbb{R}^{1\times S}$ serves as the corresponding binary mask that indicates the absence of signal values at each location. The convolution operation can be formulated as:

$$x_s^{\ell+1} = \begin{cases} \frac{1}{\|m^\ell\|_1} W(x_s^\ell \odot m^\ell) + b, & \text{if } \|m^\ell\|_1 > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where $\odot$ symbolizes element-wise multiplication and $\|m^\ell\|_1$ computes the sum of the valid signal values. We remark that the refined representations are influenced solely by the valid inputs. Besides, the scaling factor $\frac{1}{\|m^\ell\|_1}$ provides adaptive adjustments for the varying number of these valid values. We also update the mask m^ℓ according to a simple principle: if all signal values covered by the current kernel are missing, the result output mask is assigned to zero. Mathematically, this can be represented as:

$$m_t^{\ell+1} = \begin{cases} 1, & \text{if } \|m^\ell\|_1 > 0 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where the subscript t represents the output time point corresponding to the current localized window.

We implement the convolutional stem by stacking multiple layers of the masked causal convolution to distill the localized temporal dynamics into dense vector embeddings. However, since the output produced by casual convolution has the same length as the input, the second issue about *low information-dense* still exists. To tokenize the time series effectively, we introduce a large-kernel and large-stride convolution scheme at the end of the convolutional stem for rapid downsampling. By default, both the kernel size and stride are set to p. We use $H_s^i = [h_s^{i,1}, \cdots, h_s^{i,k}]$ to denote the tokenized representations, where $k = L/p$ indicates the number of time series patches. This model design comes with two advantages: it meets the d-dimension input requirements of the subsequent transformer through the terminal downsampling operation. Besides, given the $O(L^2)$ complexity of the transformer, this form of time series tokenization diminishes it to $O(L^2/k^2)$, resulting in a significant efficiency boost.

D. Spatial-Guided Temporal Transformer

After extracting both the temporal signal features and their corresponding static topological attributes, we then discuss *how to combine these two decoupled components*. To this end, we introduce the Spatial-Guided Temporal Transformer (SGTT) as illustrated in Figure 2c, which serves two roles: performing sequential modeling over the tokenized time series and equipping the temporal features with static discriminative information.

Temporal Position Encoding. Like most transformer-based architectures, we incorporate positional encodings to help each time series token be aware of its positional information. While vanilla or learnable positional encodings are commonly used, they often overlook important timestamp information that reveals natural temporal dynamics, such as the month, week, and hour. Recent advancements in multivariate time series analysis have attempted to leverage such golden information [21], [22], but they still possess a significant inductive bias. For instance, the assumption that durations such as a week or a day intrinsically display cyclical signal patterns is often taken as given.

Different from existing methods, we introduce a novel *temporal position encoding* to harness the timestamp information in an adaptive fashion. Drawing inspiration from Time2Vec [48], we assume that the representation of each timestamp is composed of a non-periodic component and several periodic dimensions. Consider the input calendar feature $x_t \in \mathbb{R}^C$ at timestamp t, the corresponding time embedding can be derived as:

$$h_t = \underbrace{[W_{np}x_t + b_{np}]}_{\text{non-periodic}} \oplus \underbrace{[\mathcal{F}(W_p x_t + b_p)]}_{\text{periodic}} \tag{4}$$

where $\oplus$ denotes the concatenation operation, $\mathcal{F}(\cdot)$ represents the periodic activation function, such as sine and cosine function. The terms $W_{np} \in \mathbb{R}^{1\times C}$ and $W_p \in \mathbb{R}^{(d-1)\times C}$ are learnable parameters, while b_{np} and b_p are the corresponding bias associated with these parameters.

In this way, we can construct cyclical factors in a learnable manner, dispensing with empirical configurations. Moreover, the quantity of periodic patterns in the time features can be controlled flexibly by adjusting the dimension of the time embedding. To ensure compatibility with the shape of the tokenized signal patch, we also apply a stride-p $1\times p$ convolution on the derived time embeddings, yielding the final temporal position encodings $H_t = [h_t^1, \cdots, h_t^k]$. We further add this specific positional information into signal token embeddings H_s^i as the input for the vanilla Transformer encoder to capture inter-patch correlations, which takes the form:

$$\tilde{H}_s^i = \text{TransformerEncoder}\left(H_s^i + H_t\right) \tag{5}$$

Spatial-Temporal Cross Attention. Following the idea of decoupled spatial-temporal representations, we develop a cross-attention mechanism to serve the second function of SGTT: equipping the temporal features with static spatial attributes to provide discriminative information. To this end, we treat the spatial representation h_g^i of the i-th node as the query to compute the compatibility function in conjunction with each term of the signal token embeddings $\tilde{H}_s^i$, which takes the form:

$$\tilde{h}_g^i \leftarrow \underset{\forall l \in \{1,\cdots,k\}}{\text{Aggregate}}\left(\text{Attention}(h_g^i, \tilde{h}_s^{i,l}) \cdot \text{Value}(\tilde{h}_s^{i,l})\right) \tag{6}$$

We remark that the static node representation remains independent of the current input time sequence, the latter of which keeps changing during the training phase. This could introduce misleading gradient flows from static spatial to dynamic temporal modeling. To tackle this issue, we employ a *gradient-stop* mechanism for each cross-attention connection, isolating the static spatial influence from the dynamic temporal tokens. Finally, the dual-channel of SGTT produces both updated temporal and spatial embeddings, which are fused together after the stacked SGTT blocks to obtain the final spatial-guided temporal representation as follows:

$$z^i = f_\Phi\left(\left[\tilde{h}_s^{i,1}; \cdots; \tilde{h}_s^{i,k}; \tilde{h}_g^i\right]\right) \tag{7}$$

where f_Φ is a fully connected feed-forward network parameterized with Φ and $[\cdot;\cdot]$ denotes concatenation.

E. Spatial-Temporal Masked AutoEncoder

Motivated by the recent progress of using denoising autoencoders for representation learning in vision [3] and text [49], we develop a Spatial-Temporal Masked Autoencoder to achieve effective multivariate time series pre-training. The goal is to approximate the distribution of MTS and generate comprehensive representations within a unified framework.

Spacetime Masking. We employ a *spacetime-agnostic* augmentation tailored for the spatiotemporal data structure in multivariate time series. Specifically, we randomly sample valid observations across both spatial and temporal dimensions in accordance with a Bernoulli distribution, resulting in a mask $\mathcal{M}(\rho)$ that is characterized by probabilities ρ. In contrast to the commonly used *time-only* or *space-only* strategies which perform sampling solely within a single dimension [12], this augmentation aligns more closely with the principle of decoupled representations. Moreover, unlike the prevalent approach in Masked Autoenoders [3], [4] where sampling is done without replacement, we replace the sampled observations with a default value that signifies the missing case. Since the practical signals that we investigate contain inherent irregular missing data, this adaptation is expected to enhance the robustness of the pre-trained encoder.

Time Series Reconstruction. In terms of information density, the time series shares similarities with *vision*, in that both exhibit heavy information redundancy. For 2D images and 3D videos, using Transformers as Decoders to discern the semantic content of the learned representations has become mainstream, as evidenced by approaches like MAE [3] and VideoMAE [4]. Conversely, we remark that a 1D time series typically contains substantially less semantic information, as exemplified by a 24-step signal value sequence. In this context, the model choice of the decoder becomes trivial and a simple MLP proves to be adequately efficient.

Once the lightweight Decoder outputs the reconstructed multivariate time series $\mathcal{S}_{out}$, our target is to enforce both masked and unmasked signals aligning with the input ground truth $\mathcal{S}_{in}$. Specifically, we optimize the following objective:

$$\begin{aligned}
\mathcal{L} &= \mathcal{L}_{valid} + \lambda * \mathcal{L}_{mask} \\
\mathcal{L}_{mask} &= \frac{1}{\|\mathcal{M}\|_1}\|\mathcal{M} \odot (\mathcal{S}_{out} - \mathcal{S}_{in})\|_1 \\
\mathcal{L}_{valid} &= \frac{1}{\|\mathbf{1} - \mathcal{M}\|_1}\|(\mathbf{1} - \mathcal{M}) \odot (\mathcal{S}_{out} - \mathcal{S}_{in})\|_1
\end{aligned} \tag{8}$$

where λ is a hyper-parameter. Terms $\mathcal{L}_{mask}$ and $\mathcal{L}_{valid}$ denote the reconstruction loss for the sampled masks and the unsampled valid values, respectively.

IV. Experiments

A. Experimental Setup

Datasets. To evaluate the efficacy of DeSTR, we collected large-scale traffic flow data across three distinct cities in China from 1/Jan/2023 to 31/Mar/2023, namely *Zhuzhou*, *Guangzhou* and *Yizhuang*. Each dataset records the vehicle volume in different directions at each intersection. Unlike previously used traffic datasets, the collected practical data covers a larger scale of nodes, well supporting the exploration into the data scaling property for multivariate time series pre-training. We aggregated the original data into 5-minute intervals and preprocessed the flow values using z-score normalization. Besides, the road network is an asymmetric graph characterized by the Euclidean distance between sensor nodes. In addition to traffic data, we also conducted experiments on an open-source geo-tagged time series dataset [50], which records the Air Quality Index (AQI) across three major Chinese cities: Bejing, Tianjin, and Guangzhou. Each dataset encapsulates hourly air quality statistics recorded by various stations from 1/May/2014 to 30/April/2015. Following previous works [41], we apply a threhold Gaussian kernel to the geographic distances between stations to obtain the undirected adjacency matrix. Table II presents the details of these pre-training datasets.

Tasks. We fine-tune the pre-trained model on the following downstream tasks to validate its generalization capability:

Time Series Forecasting. Given a series of observations $\{x_1, \ldots, x_T\}$ spanning the past T timesteps, the goal of time series forecasting is to predict the values $\{x_{T+1}, \ldots, x_{T+H}\}$ for the next H timesteps. Consistent with previous studies [24], [28], we divided the datasets chronologically into

TABLE II: Statistics of datasets

	Datasets	Nodes	Edges	TimeSteps	MissingRatio
Traffic	Zhuzhou	329	372	25546	15.30%
	Guangzhou	357	251	25541	16.16%
	Yizhuang	1402	1492	25827	37.96%
AQI	Beijing	36	38	8759	13.25%
	Tianjin	26	28	8759	18.67%
	Guangzhou	37	64	8759	17.09%

training, validation, and testing sets following a 6:2:2 partitioning scheme. In experiments, we use the past 24 timesteps for prediction and report results for varying forecasting horizons, specifically at 3/6/12 timesteps. For a comprehensive comparative assessment, we choose three distinct categories of baseline models, including Deep Sequential Models (FC-LSTM, TCN, DLinear), STGNN-based models (ASTGCN, AGCRN, STGODE, DSTAGNN), and Time Series Pre-training models (TNC, TS2Vec, Patch2TST):

- FC-LSTM: which employs a fully connected layer on input features, processed with LSTM.
- TCN [17]: which combines dilated convolution and causal convolution to extract the dense signal embeddings.
- DLinear [51]: which decomposes the input time series into trend and seasonality patterns and handle these two patterns separately with simple linear layers.
- ASTGCN [23]: which introduces spatial-temporal attention mechanism to capture the dynamic spatial-temporal correlations.
- AGCRN [24]: which attempts to capture node-specific patterns with node adaptive parameter learning and generate the graph in an adaptive manner.
- STGODE [26]: which captures spatial-temporal dynamics with a tensor-based ordinary differential equation.
- DSTAGNN [28]: which introduces Wasserstein distance to construct the spatial-temporal aware graph and process the time series with multiple spatial-tempotal attention blocks.
- TNC [9]: which is multivariate time series representation framework that employs contrastive learning on neighboring time points.
- TS2Vec [10]: which introduces contrastive learning on both temporal and spatial dimensions to obtain more discriminative representations.
- PatchTST [12]: which is a fashion framework for time series representation learning based on the masked autoencoding strategy.

Time Series Imputation. Given a sequence of observations $\boldsymbol{x} = (x_1, \ldots, x_T)$ spanning T timesteps, along with a binary masking vector $\boldsymbol{m} = (m_1, \ldots, m_T)$ indicating the absence of x_i. The objective of time series imputation is to estimate the missing values by leveraging the observed features. In practice, we concentrate on the *out-of-sample* scenario, a more challenging problem where the model is trained and evaluated on disjoint sequences [41]. Specifically, we partitioned the dataset temporally in the same fashion as in the forecasting task. Building upon previous studies [42], we simulated the presence of missing data by randomly masking 30%/50%/70% of valid time series and utilizing a window size of 24 steps. In addition to adopting time series pre-trained models as comparative baselines (TNC, TS2Vec, Patch2TST), similar to those employed in forecasting tasks, we also compare DeSTR against three other categories of imputation-oriented models, including Statistical Methods (MEAN, KNN, MICE), Deep Sequential Models (BRITS, SAITS), and STGNN-based models (GRTN, SPIN):

- MEAN: which impute the missing data naively with the mean value of observed data at the same time point.
- KNN [32]: which considers the relationships of different variables to impute the missing values.
- MICE [52]: which iteratively impute the missing data using multiple regression models.
- BRITS [35]: which treats the inputs as a RNN graph and impute the missing values in a bidirectional manner.
- SAITS [37]: which introduces self-attention mechanism into time series imputation.
- GRIN [41]: which uses a bidirectional RNN graph and process the hidden representations with a spatial decoder.
- SPIN [42]: which is a fully attention-based architecture that impute the missing values on sparse discrete observations.

Implementations. We employ a temporal convolution stem comprising three layers of masked causal convolutions and set parameter p to 4, tokenizing the 24-step time series into 6 patches. For the spatial convolution, two layers of GNN are used to extract the static node features. We use two layers of SGTT blocks to fuse the temporal and spatial representations, maintaining a fixed number of four multi-heads in the attention mechanism. During pre-training, the ratio of *spacetime-agnostic* masking ρ is set to 0.4 and the window size is maintained at 24 steps. The reconstruction decoder is structured as a two-layer MLP and the hyper-parameter λ is set to 6. The comprehensive pre-training is performed solely on the training data for 200 epochs, and we use a cosine learning rate [53] of $5e{-}4$ warming up for 20 epochs. In addition, we use the AdamW optimizer and set the batch size to 28. For each downstream task, two training alternatives are provided: (1) an end-to-end supervised learning strategy that involves pre-training the DeSTR from scratch, and (2) replacing the reconstruction decoder with an MLP head and fine-tuning the entire network. For all baseline models, we follow the parameter configurations specified in their original works to reproduce the results. Additionally, to ensure fairness in evaluation, the dimension of the hidden embedding across all models is standardized to 128. We use 4 Nvidia Tesla V100 32G GPUs to perform the pre-training and conduct all end-to-end supervised training on a single GPU. It takes about 8 hours to pre-train DeSTR. In the fine-tuning stage, we use a constant learning rate schedule with a learning rate of 1×10^{-3}. Both the data and source codes have been made publicly accessible at: https://github.com/zruiii/DeSTR.

B. Overall Performance

Results on forecasting. Table III details comparative time series forecasting results of DeSTR (both supervised from

TABLE III: The comparison results for time series forecasting with DeSTR. We use prediction horizons of 3/6/12 steps for all datasets. The best results are highlighted in bold and the second-best are underlined.

Traffic Datasets	Zhuzhou		Guangzhou		Yizhuang	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
FC-LSTM	5.740 / 6.102 / 6.777	3.337 / 3.498 / 3.794	5.652 / 5.981 / 6.762	2.904 / 3.029 / 3.317	5.325 / 6.007 / 7.050	2.422 / 2.628 / 2.998
TCN	5.886 / 6.359 / 7.090	3.413 / 3.619 / 3.941	6.031 / 6.476 / 7.395	3.059 / 3.217 / 3.511	5.578 / 6.253 / 7.267	2.540 / 2.777 / 3.127
DLinear	6.168 / 6.652 / 7.581	3.527 / 3.731 / 4.097	5.883 / 6.345 / 7.467	2.993 / 3.151 / 3.510	6.022 / 6.596 / 7.494	2.791 / 3.029 / 3.395
ASTGCN	6.071 / 6.457 / 7.212	3.451 / 3.615 / 3.941	6.167 / 6.731 / 7.855	3.083 / 3.298 / 3.655	6.055 / 6.581 / 7.371	2.839 / 3.038 / 3.348
AGCRN	5.865 / 6.192 / 6.851	3.398 / 3.545 / 3.840	5.674 / 5.914 / 6.504	2.919 / 3.012 / 3.238	6.077 / 6.745 / 7.872	2.871 / 3.112 / 3.513
STGODE	5.579 / 5.852 / 6.352	3.227 / 3.332 / 3.558	5.521 / 5.778 / 6.508	2.864 / 2.966 / 3.220	4.851 / 5.265 / 6.025	2.374 / 2.519 / 2.785
DSTAGNN	5.751 / 6.021 / 6.599	3.299 / 3.417 / 3.667	5.751 / 6.006 / 6.638	2.933 / 3.030 / 3.290	5.849 / 6.397 / 7.229	2.820 / 3.070 / 3.425
TNC	5.838 / 6.151 / 6.493	3.281 / 3.514 / 3.773	5.691 / 5.788 / 6.098	2.855 / 2.931 / 3.224	5.255 / 5.422 / 5.650	2.578 / 2.647 / 2.876
TS2Vec	5.824 / 6.132 / 6.678	3.257 / 3.474 / 3.681	5.820 / 6.044 / 6.943	2.953 / 3.027 / 3.274	OOM	OOM
PatchTST	5.733 / 6.122 / 6.762	3.332 / 3.500 / 3.787	5.565 / 5.981 / 6.836	2.875 / 3.037 / 3.331	5.357 / 5.965 / 6.998	2.445 / 2.633 / 2.996
Supervised	5.462 / 5.678 / 5.959	3.161 / 3.237 / 3.323	**5.146 / 5.293 / 5.589**	**2.671 / 2.717 / 2.804**	4.420 / 4.638 / 4.947	2.113 / 2.171 / 2.256
Fine-tuning	**5.424 / 5.618 / 5.843**	**3.124 / 3.192 / 3.278**	5.224 / 5.406 / 5.663	2.686 / 2.740 / 2.826	**4.113 / 4.287 / 4.552**	**2.045 / 2.098 / 2.177**

AQI Datasets	Beijing		Tianjin		Guangzhou	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
FC-LSTM	35.440 / 43.363 / 53.068	20.539 / 26.033 / 34.166	25.924 / 32.921 / 38.331	15.431 / 21.074 / 25.219	11.233 / 13.657 / 16.256	6.807 / 8.515 / 10.467
TCN	35.630 / 43.906 / 52.041	20.281 / 26.523 / 34.566	26.215 / 32.773 / 37.886	16.544 / 20.117 / 25.175	11.065 / 13.526 / 16.185	6.819 / 8.505 / 10.498
DLinear	35.353 / 43.696 / 53.027	19.094 / 25.368 / 33.393	26.070 / 31.763 / 37.171	15.520 / 21.153 / 25.208	11.072 / 13.513 / 16.180	6.770 / 8.464 / 10.423
ASTGCN	36.180 / 44.655 / 53.702	20.104 / 25.248 / 32.797	27.089 / 32.658 / 37.319	16.555 / 21.058 / 25.992	11.293 / 13.775 / 16.326	7.200 / 8.911 / 10.765
AGCRN	36.389 / 44.512 / 53.588	20.186 / 26.309 / 34.178	27.490 / 32.182 / 37.846	16.246 / 21.801 / 25.911	11.702 / 14.043 / 16.539	7.193 / 8.789 / 10.711
STGODE	33.891 / 42.185 / 51.815	18.795 / 24.864 / 32.704	25.325 / 32.712 / 37.074	15.309 / 20.607 / 24.844	10.518 / 13.001 / 15.951	6.691 / 8.835 / 10.463
DSTAGNN	34.262 / 42.754 / 52.195	19.238 / 25.714 / 33.888	27.235 / 33.071 / 38.826	16.391 / 21.004 / 25.138	11.573 / 14.066 / 16.625	7.403 / 9.341 / 11.106
TNC	35.785 / 43.886 / 53.067	19.599 / 25.136 / 33.675	26.122 / 32.505 / 38.513	15.721 / 21.034 / 25.581	11.165 / 13.663 / 16.315	6.908 / 8.674 / 10.878
TS2Vec	35.420 / 43.428 / 52.392	19.121 / 25.235 / 33.530	25.929 / 31.361 / 37.943	15.452 / 20.459 / 25.136	10.973 / 13.529 / 15.954	6.727 / 8.481 / 10.358
PatchTST	35.701 / 44.115 / 53.581	20.013 / 26.150 / 34.525	26.268 / 32.568 / 38.325	15.934 / 21.375 / 25.194	11.115 / 13.884 / 16.260	7.185 / 8.726 / 10.282
Supervised	33.613 / 41.851 / 51.172	17.941 / 23.189 / 31.125	24.920 / 31.163 / 36.519	15.077 / 19.872 / 24.364	10.931 / 13.278 / 16.010	6.508 / 8.355 / 10.108
Fine-tuning	**33.242 / 40.304 / 50.531**	**17.445 / 22.298 / 30.760**	**24.268 / 30.867 / 36.120**	**14.827 / 19.303 / 23.888**	**10.137 / 12.514 / 15.137**	**6.299 / 8.005 / 9.876**

scratch and fine-tuned) against other state-of-the-art approaches. Across the board for both Traffic and AQI datasets, our method consistently outperforms the baseline models under these two training procedures. Specifically, for the traffic scenario, DeSTR achieves up to 7.9% improvement on *Zhuzhou*, 12.9% on *Guangzhou*, and a substantial improvement of 21.8% on *Yizhuang*, as measured by the MAE metric. For the air quality data, our approach registers enhancements of up to 7.2% on *Beijing*, 6.4% on *Tianjin*, and 5.8% on *Guangzhou*. This demonstrates the superiority of DeSTR in capturing the temporal dynamics of signals under real-world noises. Besides, across a majority of the evaluated scenarios (including traffic data from *Zhuzhou* and *Yizhuang*, along with all three air quality datasets), we can observe that models fine-tuned from pre-trained DeSTR exhibit enhanced performance compared to those subject to end-to-end supervised training. In particular, the enhancements manifest as a remarkable increase in accuracy: a 3.5% reduction in MAE and an 8.0% reduction in RMSE are observed for traffic forecasting in *Yizhuang*. Similarly, air quality forecasting in *Guangzhou* experienced improvements of 4.2% in MAE and 7.3% in RMSE. Regarding the baseline models, a notable performance decline is observed in adaptive graph-based STGNNs like ASTGCN and AGCRN, falling even below TCN which lacks spatial correlation capabilities. We conjecture that the inherent missing data precludes these models from deriving comprehensive temporal representations, adversely affecting latent graph learning. This generated graph in turn misguides the information propagation between nodes, leading to suboptimal performance. We also observe that STGODE performs the best among all baselines, which suggests the promising efficacy of integrating ordinary differential equations to capture spatiotemporal dynamics and employing multiple pre-defined graphs concurrently to minimize learning biases. Moreover, among pre-training methods, the contrastive-based models slightly outperform those based on masked autoencoding. We attribute this to the ability of temporal contrastive learning to implicitly capture temporal dynamics, which benefits time series forecasting. However, these methods typically require a large number of negative samples [54], leading to substantial memory overhead and failure to adapt to high-dimensional multivariate time series, such as *Yizhuang* which encompasses thounds of nodes.

Results on imputation. We present a comprehensive comparison with time series imputation models in Table IV. Compared to baseline models, DeSTR improves performance in ten of the eighteen traffic flow imputation scenarios with fine-tuning, achieving notable enhancements in the MAE metric by up to 0.8% on *Zhuzhou* and a remarkable 18.6% on *Yizhuang*. Furthermore, these advancements span across all scenarios within the air quality imputation domain, marking significant enhancements in the MAE metric by up to 7.5% in Beijing, 6.0% in Tianjin, and 8.1% in Guangzhou. The results demonstrate the ability of DeSTR to better fill in missing data based on valid observations. Besides, we note that pre-training consistently enhances performance in both types of data compared to supervised training from scratch, with maximum gains of 6.4% on traffic flow imputation in *Guangzhou*, and 7.3% on air quality imputation in *Tianjin*. This empirical evidence strongly validates the generalization capabilities of our pre-training model. Among baseline models, the STGNN-based SPIN model also achieves competitive results. However, it suffers from high computational costs due to explicit real-time message passing and fails to operate on large-scale datasets like *Yizhuang*. In addition, we observe comparable

TABLE IV: The comparison results for time series imputation with DeSTR. We simulate missing scenario with 30%/50%/70% masking ratio for all datasets. The best results are highlighted in bold and the second-best are underlined.

Traffic Datasets	Zhuzhou		Guangzhou		Yizhuang	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
MEAN	6.774 / 6.779 / 6.820	3.960 / 3.964 / 3.984	8.409 / 8.432 / 8.474	4.029 / 4.033 / 4.054	5.984 / 5.985 / 5.992	2.796 / 2.794 / 2.801
KNN	5.804 / 6.124 / 6.628	3.446 / 3.589 / 3.809	5.900 / 6.089 / 6.339	3.092 / 3.179 / 3.301	4.513 / 5.164 / 5.993	2.280 / 2.510 / 2.813
MICE	6.774 / 6.779 / 6.820	3.960 / 3.964 / 3.984	8.409 / 8.433 / 8.474	4.029 / 4.033 / 4.054	9.260 / 9.253 / 9.250	4.460 / 4.460 / 4.458
BRITS	5.342 / 5.569 / 6.035	3.066 / 3.199 / 3.488	5.017 / 5.442 / 7.046	2.622 / 2.859 / 3.724	4.876 / 5.474 / 6.299	2.438 / 2.757 / 3.201
SAITS	6.168 / 7.554 / 9.728	3.544 / 4.690 / 6.234	7.287 / 9.151 / 11.912	3.639 / 4.711 / 6.116	9.156 / 9.389 / 9.481	4.258 / 4.372 / 4.410
SPIN	5.182 / 5.384 / **5.693**	3.038 / 3.141 / **3.268**	**4.795 / 4.950 / 5.284**	**2.493 / 2.571 / 2.704**	OOM	OOM
GRIN	5.631 / 6.098 / 7.440	3.295 / 3.523 / 4.214	5.097 / 5.591 / 6.619	2.641 / 2.876 / 3.375	4.128 / 4.537 / 5.353	2.079 / 2.256 / 2.605
TNC	5.688 / 5.851 / 6.076	3.568 / 3.720 / 3.892	5.982 / 6.183 / 6.427	3.097 / 3.224 / 3.368	4.535 / 4.711 / 4.937	2.285 / 2.400 / 2.539
TS2Vec	5.680 / 5.842 / 6.066	3.549 / 3.701 / 3.871	5.780 / 6.174 / 6.417	2.962 / 3.078 / 3.210	OOM	OOM
PatchTST	5.598 / 5.999 / 6.753	3.265 / 3.445 / 3.791	5.551 / 6.031 / 7.151	2.867 / 3.051 / 3.526	4.300 / 4.809 / 5.549	2.136 / 2.345 / 2.622
Supervised	5.210 / 5.550 / 6.283	3.072 / 3.224 / 3.546	5.251 / 5.487 / 6.066	2.727 / 2.823 / 3.041	3.777 / 4.026 / 4.365	1.938 / 2.031 / 2.171
Fine-tuning	**5.140 / 5.356** / 5.898	**3.033 / 3.114** / 3.368	4.943 / 5.138 / 5.560	2.602 / 2.686 / 2.847	**3.747 / 3.979 / 4.344**	**1.918 / 1.994 / 2.116**

AQI Datasets	Beijing		Tianjin		Guangzhou	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
MEAN	31.729 / 31.557 / 32.621	18.705 / 18.863 / 19.304	24.853 / 24.972 / 26.211	16.271 / 16.444 / 17.165	14.452 / 14.650 / 14.999	9.686 / 9.790 / 10.081
KNN	21.511 / 23.481 / 26.226	12.888 / 13.986 / 15.852	20.847 / 21.801 / 24.112	12.796 / 13.716 / 15.298	12.443 / 13.468 / 15.195	8.347 / 8.975 / 10.172
MICE	32.014 / 31.926 / 32.943	18.800 / 18.992 / 19.381	24.853 / 24.970 / 25.182	16.271 / 16.444 / 17.078	14.452 / 14.650 / 15.005	9.686 / 9.790 / 10.084
BRITS	23.210 / 24.514 / 29.216	12.034 / 13.454 / 16.659	19.306 / 20.765 / 23.818	12.100 / 13.342 / 15.924	9.486 / 10.505 / 11.725	6.101 / 6.724 / 7.787
SAITS	26.686 / 34.643 / 51.352	15.007 / 22.182 / 36.333	19.788 / 24.999 / 32.145	13.070 / 17.614 / 23.908	10.519 / 12.930 / 16.225	6.783 / 8.510 / 11.393
SPIN	19.279 / 20.232 / 24.211	8.544 / 9.513 / 11.117	14.113 / 16.026 / 19.296	7.743 / 8.907 / 11.152	6.969 / 7.918 / 9.312	4.067 / 4.970 / 5.554
GRIN	19.099 / 20.744 / 25.252	9.801 / 11.337 / 13.804	14.258 / 16.635 / 19.797	8.069 / 9.463 / 11.936	7.005 / 7.920 / 9.873	4.075 / 4.714 / 5.796
TNC	22.486 / 23.542 / 23.613	13.248 / 13.771 / 13.951	15.618 / 15.728 / 18.333	10.171 / 11.852 / 12.562	8.832 / 9.577 / 11.363	5.452 / 7.051 / 7.806
TS2Vec	21.968 / 23.042 / 23.163	12.776 / 13.182 / 13.365	14.874 / 15.472 / 18.823	9.733 / 11.192 / 11.986	8.335 / 9.902 / 10.713	4.929 / 6.531 / 7.303
PatchTST	22.640 / 23.598 / 23.713	13.327 / 13.772 / 13.916	15.376 / 16.042 / 18.374	10.288 / 11.748 / 12.533	8.842 / 9.633 / 11.394	5.462 / 7.072 / 7.934
Supervised	18.010 / 19.753 / 21.283	8.162 / 8.960 / 10.898	13.596 / 15.657 / 18.749	7.587 / 9.434 / 11.647	6.492 / 7.291 / 9.276	3.966 / 4.641 / 5.480
Fine-tuning	**17.189 / 18.942 / 20.469**	**7.945 / 8.158 / 10.282**	**12.770 / 14.845 / 17.862**	**7.279 / 8.630 / 10.791**	**6.233 / 7.131 / 9.130**	**3.737 / 4.419 / 5.414**

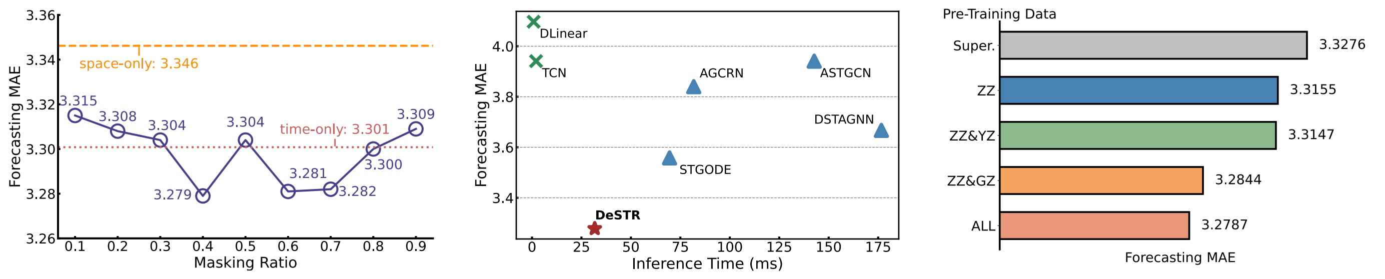
performance between MICE and the simpler MEAN method, which may be caused by the weak inter-variable correlations and complex missing patterns of the data. We also notice that BRITS and GRIN demonstrate remarkable results, indicating bidirectional sequence modeling as a promising choice for time series imputations. For pre-training models, we observe a significant decrease in performance with methods based on contrastive learning. This decline is attributed to these methods mainly employing a linear probing protocol, which leverages offline-inferred fixed representations as input to downstream shallow architectures. As a result, the crucial non-linear features in the time series are not fully pursued.

C. Main Properties

Mask sampling strategy. In Figure 3a, we compare the forecasting performance of DeSTR on the *Zhuzhou* dataset, considering different mask sampling strategies and a range of masking ratios. The *space-only* masking, which removes all valid signals from sampled nodes, yields the poorest performance. Since no temporal information is present, the reconstruction evolves into a zero-shot problem, which is extremely hard for time series representation learning. We also study *time-only* masking, wherein observations are uniformly sampled across the temporal dimension for all nodes. Despite its simplicity, its performance still falls short compared to *spacetime-agnostic* augmentation. In the *spacetime-agnostic* approach, an increase in the mask ratio first leads to a decrease in prediction error, followed by an increase, with the optimal mask ratio lying between $40-70\%$. This trend suggests a notable information redundancy in time series as opposed to *text*, where the best practice of masking ratio is 15% [49]. We also observe a slight upward bump when the mask ratio hits 50%, possibly due to the exact halving of valid values, disrupting the information balance the model is adapted to.

Computational Costs. To empirically validate the efficiency of DeSTR, we assess its performance by visually depicting the trade-off between time series forecasting error and model inference time. The analysis is performed on the largest *Yizhuang* dataset and presented in Figure 3b. We observe that DeSTR can achieve a low prediction error (3.278) while ensuring a fast inference time (31ms). Compared with deep sequential models, DeSTR provides significant performance enhancements at a reasonable computational cost. For the STGNN-based models, the real-time propagation of temporal information between adjacent nodes results in a considerable inference latency, especially for those that include additional adaptive graph learning, such as ASTGCN. Conversely, DeSTR benefits from its architecture of decoupled representations, eschewing online message passing, and achieves a $5.6\times$ speed-up compared to DSTAGNN and a $2.6\times$ speed-up compared to STGODE. Therefore, it can be concluded that DeSTR emerges as a superior solution in terms of both accuracy and computational efficiency.

Data scaling. We explore the data scaling of DeSTR by varying the size of the pre-training dataset and comparing the forecasting results of both supervised training and fine-tuning on the *Zhuzhou* dataset in Figure 3. Compared to end-to-end supervised training, the results indicate a consistent performance improvement for fine-tuned models, whether the pre-training is applied to a single *Zhuzhou* or multiple datasets. Moreover, we can also observe a similar phenomenon of data scaling commonly witnessed in NLP. Specifically, as the amount of pre-training data increases, the performance of models pre-trained on multi-region spatiotemporal data shows



(a) Masking Ratio. (b) MAE vs. Inference Time. (c) Data Scaling Capacity.

Fig. 3: Main properties study of (a) mask sampling strategy, where *space-only* and *time-only* are sampled at a 40% masking ratio; (b) computation costs comparison among deep sequential models, spatial-temporal graph neural networks and our DeSTR; (c) data scaling capability, where ZZ, GZ, YZ are the abbreviations for Zhuzhou, Guangzhou, and Yizhuang.

TABLE V: The comparison results for long-term time series forecasting on Zhuzhou.

Models	RMSE	MAE
FC-LSTM	8.562 / 8.878 / 8.934 / 9.372	5.074 / 5.241 / 5.279 / 5.404
TCN	9.032 / 9.116 / 9.372 / 9.726	6.224 / 6.227 / 6.557 / 6.861
DLinear	8.233 / 8.729 / 8.862 / 9.170	4.392 / 4.650 / 4.732 / 5.282
ASTGCN	8.115 / 8.597 / 8.876 / 9.228	4.386 / 4.695 / 4.784 / 5.272
AGCRN	8.265 / 8.526 / 8.627 / 8.964	4.381 / 4.518 / 4.553 / 4.781
STGODE	6.866 / 6.925 / 7.181 / 7.775	4.058 / 4.145 / 4.159 / 4.293
DSTAGNN	6.952 / 7.130 / 7.829 / 8.023	4.086 / 4.164 / 4.248 / 4.325
TNC	7.603 / 7.677 / 7.694 / 7.854	4.239 / 4.343 / 4.510 / 4.712
TS2Vec	7.146 / 7.262 / 7.311 / 7.672	4.061 / 4.136 / 4.283 / 4.356
PatchTST	7.383 / 7.436 / 7.506 / 7.798	4.246 / 4.270 / 4.275 / 4.323
Supervised	6.759 / 6.822 / 7.137 / 7.492	3.975 / 4.084 / 4.127 / 4.229
Fine-tuning	**6.668 / 6.784 / 6.806 / 7.059**	**3.818 / 4.027 / 4.065 / 4.105**

remarkable improvements. These encouraging results suggest that DeSTR is a general spatiotemporal learner and provides a promising path for gathering more distinct regional data to further enhance the effectiveness of the pre-training model.

Long-term forecasting. To validate the effectiveness of our approach in long-term time series forecasting tasks, we also conducted experiments to forecast the observations of future 96/192/336/720 timesteps based on the past 96 timesteps, aligning with the prior studies [12]. Table V reports the comparative results on the *Zhuzhou* dataset. We can find that despite a relative decline in predictive accuracy for long-term forecasting compared to short-term scenarios, DeSTR consistently outperforms all other models. The pre-training still benefits DeSTR under long-term forecasting, yielding an approximate 3.0% enhancement in the MAE metric for 720-step forecasts. In addition, it is worth noting that, in contrast to 12-step forecasts, the performance of DeSTR in 96-step forecasting experiences a smaller reduction in MAE by 22.2%, a decrement that is notably smaller than that of the best baseline model STGODE, which diminishes by about 25.7%. This indicates that as the prediction window increases, DeSTR maintains a more stable performance and demonstrates superior robustness.

D. Ablation Study

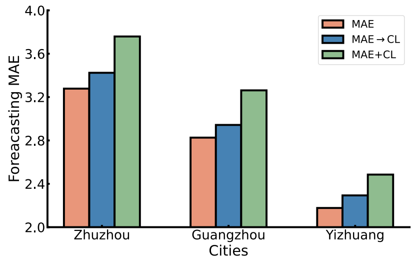
Main components. To validate the contribution of several key components of DeSTR, we carry out ablation studies focusing on the masked causal convolution (MCC) and SGTT block. We conduct end-to-end supervised training on the *Zhuzhou* dataset and present the comparison in Table VI. Specifically, we substitute MCC with three alternatives: (1) TCN, (2) LSTM, and (3) MLP for token embedding as utilized in PatchTST [12]. Among these, MCC yields the most superior results in both tasks and MLP performs the worst. Additionally, we also notice a comparable performance when using LSTM to capture temporal dynamics. However, this RNN architecture is not as efficient as MCC due to its limitations on parallelization.

TABLE VI: Ablation study on the main modules, where MAE on 12-steps forecasting and 70% imputation is reported.

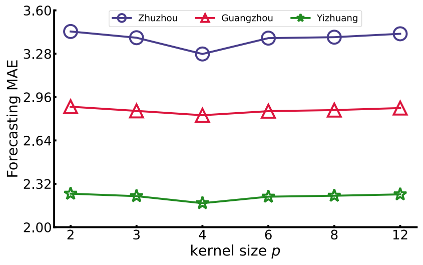
DeSTR Variants	Forecasting	Imputation
MCC$\rightarrow$TCN	3.335	3.570
MCC$\rightarrow$LSTM	3.331	3.552
MCC$\rightarrow$MLP	3.384	3.553
SGTT *w/o* spatial	3.334	3.585
SGTT *w/o* stop-grad	3.331	3.608
DeSTR *w/o* $\mathcal{L}_{valid}$	3.338	3.580
DeSTR	**3.323**	**3.546**

To investigate the efficacy of incorporating static spatial discriminative information into temporal features, we devise two variants: The first one eliminates the spatial embedding and employs Transformer to process the temporal token representations. The other variant excludes the *stop-grad* operation, in this context, the spatial embedding can be seen as a *variate-specific* [CLS] which plays a similar role with the [CLS] token in the classic Transformer architecture. We observe a considerable performance drop when back-propagating the gradients from the spatial module to the temporal module, potentially due to unstable training. Moreover, the lack of spatial information notably diminishes the discrimination capability of DeSTR across various variables, leading to suboptimal performance. We also assess the effectiveness of DeSTR without optimizing for the reconstruction of valid values, which results in a slight performance decline. To sum up, this empirical evidence well supports the effectiveness of MCC in extracting temporal features from signals with irregular missing patterns and highlights the benefits of SGTT for integrating spatial and temporal representations.

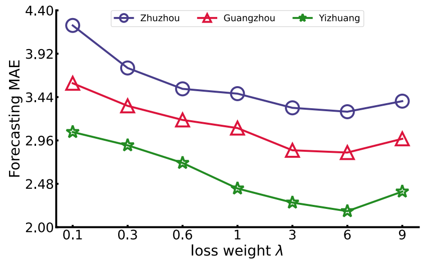
Pretext Task. To investigate the potential advantages of employing Contrastive Learning (CL) in the pre-training of time series data, we extend our analysis to include two distinct experimental setups: The first variant transitions from the MAE framework to a purely CL-based approach for self-supervised learning, denoted as MAE$\rightarrow$CL. The second variant integrates



(a) Contrastive Learning for Pre-training.



(b) Parameter Sensitivity on Kernel Size p.



(c) Parameter Sensitivity on Loss Weight λ.

Fig. 4: (a) Ablation study of integrating contrastive learning as optimization objective in the pre-training phase; (b) parameter sensitivity study on the kernel size p in the temporal convolution stem; (c) parameter sensitivity study on the loss weight λ for masked data reconstruction.

both MAE and CL as the optimization objectives through a weighted sum of their losses, denoted as MAE+CL. In these experiments, we adopt the debiased contrastive objective from the state-of-the-art method TNC [9], which aims to ensure that the signal distribution from within a specific neighborhood is discernible from that of signals outside the neighborhood. The results of 12-step time series forecasting are shown in Figure 4a. We can observe that both variants, MAE→CL and MAE+CL, perform worse than using MAE only, which indicates that the contrastive learning-based methods fall short of MAE-based methods in the pre-training of multivariate time series, especially in real-world scenarios characterized by frequent missing observations. Moreover, the MAE+CL variant, which combines denoising autoencoder principles with contrastive learning, yields the worst results. This underperformance is attributed to the different optimization objectives of these two approaches: MAE is designed to reconstruct masked data, whereas CL aims to enhance discriminability among distinct instances. This divergence potentially hinders model convergence in the training phase, preventing the achievement of optimal performance states.

E. Parameter Sensitivity

We conducted a detailed investigation into the sensitivity of the parameter p, representing the kernel size in the temporal convolution stem, as well as the impact of the weight $\mathcal{L}_{mask}$ within the reconstruction loss. Specifically, we adjusted the parameter p across the set $\{2, 3, 4, 6, 8, 12\}$ to patchfy the 24-step inputs into $\{12, 8, 6, 4, 3, 2\}$ tokens, assessing its effect on time series forecasting performance within an end-to-end training scheme. As depicted in Figure 4b, we can observe a trend across all three datasets where the performance of DeSTR gradually improves with an increase in p. However, performance begins to decline when p reaches 4. We hypothesize that as p increases, the input tokens to the SGTT become more compressed, facilitating a more effective capture of temporal patterns by the transformer architecture. Beyond a certain threshold, further compression leads to the loss of fine-grained features, thereby diminishing forecast accuracy. In addition, we also tuned the hyper-parameter λ across the values $\{0.1, 0.2, 0.3, 1, 3, 6, 9\}$ during the pre-training phase and evaluated its impact on time series forecasting. As shown in Figure 4c, the optimal performance is observed at $\lambda = 6$, with larger or smaller weights would detrimentally affect the effectiveness of DeSTR. This emphasizes the advantages of integrating the reconstruction objectives for both masked and valid data within the optimization. Furthermore, it indicates that the constraints applied to the masked portions exert a more significant influence on the efficacy of DeSTR during the pre-training phase.

V. Conclusion

In this study, we presented DeSTR, a pioneering pre-training framework tailored for multivariate time series across multi-region spatiotemporal data. We introduced a novel backbone network based on decoupled spatial-temporal representations, to address the challenges posed by deep sequential models and emerging STGNNs. Moreover, we utilized mask autoencoding as the foundational pre-training paradigm which incorporates the spacetime-agnostic augmentation. We successfully pre-trained a unified representation learning framework across three distinct real-world datasets. Extensive experiments were conducted on two common downstream tasks, forecasting and imputation, to compare the performance of this pre-trained model against both state-of-the-art specifically-crafted methods and other pre-training models. The empirical results clearly demonstrate the high capability of DeSTR as an effective and general spatiotemporal learner. Additionally, it is promising to note that scaling the pre-training data provides a remarkable performance boost, akin to the trends observed in LLM. We anticipate that this insight will markedly propel further exploration in spatial-temporal pre-training.

Acknowledgment

This work was supported in part by the grants from National Natural Science Foundation of China (No.62222213, U22B2059, U23A20319, 62072423), and the USTC Research Funds of the Double First-Class Initiative (No.YD2150002009). Enhong Chen, Tong Xu, and Le Zhang are the corresponding authors.

References

[1] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *NeurIPS*, vol. 33, pp. 1877–1901, 2020.

[3] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked autoencoders are scalable vision learners,” in *CVPR*, 2022, pp. 16 000–16 009.

[4] Z. Tong, Y. Song, J. Wang, and L. Wang, “Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training,” *NeurIPS*, vol. 35, pp. 10 078–10 093, 2022.

[5] Z. Zhang and D. Crandall, “Hierarchically decoupled spatial-temporal contrast for self-supervised video representation learning,” in *WACV*, 2022, pp. 3235–3245.

[6] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” *NeurIPS*, vol. 33, pp. 12 449–12 460, 2020.

[7] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, “Hubert: Self-supervised speech representation learning by masked prediction of hidden units,” *TASLP*, vol. 29, pp. 3451–3460, 2021.

[8] E. Eldele, M. Ragab, Z. Chen, M. Wu, C. K. Kwoh, X. Li, and C. Guan, “Time-series representation learning via temporal and contextual contrasting,” *arXiv preprint arXiv:2106.14112*, 2021.

[9] S. Tonekaboni, D. Eytan, and A. Goldenberg, “Unsupervised representation learning for time series with temporal neighborhood coding,” in *ICLR*, 2020.

[10] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, and B. Xu, “Ts2vec: Towards universal representation of time series,” in *AAAI*, vol. 36, no. 8, 2022, pp. 8980–8987.

[11] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, “A transformer-based framework for multivariate time series representation learning,” in *KDD*, 2021, pp. 2114–2124.

[12] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, “A time series is worth 64 words: Long-term forecasting with transformers,” in *The Eleventh ICLR*, 2022.

[13] J. Oskarsson, P. Sidén, and F. Lindsten, “Temporal graph neural networks for irregular data,” in *AISTATS*. PMLR, 2023, pp. 4515–4531.

[14] J. Chen, T. Ma, and C. Xiao, “Fastgcn: Fast learning with graph convolutional networks via importance sampling,” in *ICLR*, 2018.

[15] E. Delasalles, A. Ziat, L. Denoyer, and P. Gallinari, “Spatio-temporal neural networks for space-time data modeling and relation discovery,” *KIS*, vol. 61, pp. 1241–1267, 2019.

[16] B. Pang, K. Zha, H. Cao, J. Tang, M. Yu, and C. Lu, “Complex sequential understanding through the awareness of spatial and temporal concepts,” *NMI*, vol. 2, no. 5, pp. 245–253, 2020.

[17] S. Bai, J. Z. Kolter, and V. Koltun, “An empirical evaluation of generic convolutional and recurrent networks for sequence modeling,” *arXiv preprint arXiv:1803.01271*, 2018.

[18] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *NeurIPS*, vol. 30, 2017.

[21] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *AAAI*, vol. 35, no. 12, 2021, pp. 11 106–11 115.

[22] H. Wu, J. Xu, J. Wang, and M. Long, “Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting,” *NeurIPS*, vol. 34, pp. 22 419–22 430, 2021.

[23] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, “Attention based spatial-temporal graph convolutional networks for traffic flow forecasting,” in *AAAI*, vol. 33, no. 01, 2019, pp. 922–929.

[24] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, “Adaptive graph convolutional recurrent network for traffic forecasting,” *NeurIPS*, vol. 33, pp. 17 804–17 815, 2020.

[25] R.-G. Cirstea, T. Kieu, C. Guo, B. Yang, and S. J. Pan, “Enhancenet: Plugin neural networks for enhancing correlated time series forecasting,” in *ICDE*. IEEE, 2021, pp. 1739–1750.

[26] Z. Fang, Q. Long, G. Song, and K. Xie, “Spatial-temporal graph ode networks for traffic flow forecasting,” in *KDD*, 2021, pp. 364–373.

[27] R.-G. Cirstea, B. Yang, C. Guo, T. Kieu, and S. Pan, “Towards spatio-temporal aware traffic time series forecasting,” in *ICDE*. IEEE, 2022, pp. 2900–2913.

[28] S. Lan, Y. Ma, W. Huang, W. Wang, H. Yang, and P. Li, “Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting,” in *ICML*. PMLR, 2022, pp. 11 906–11 917.

[29] N. Kim, J. Song, S. Lee, J. Choe, K. Han, S. Park, and S.-W. Kim, “Apots: A model for adversarial prediction of traffic speed,” in *ICDE*. IEEE, 2022, pp. 3353–3359.

[30] Y. Yao, D. Li, H. Jie, H. Jie, T. Li, J. Chen, J. Wang, F. Li, and Y. Gao, “Simplets: An efficient and universal model selection framework for time series forecasting,” *VLDB*, vol. 16, no. 12, pp. 3741–3753, 2023.

[31] Y. Cui, K. Zheng, D. Cui, J. Xie, L. Deng, F. Huang, and X. Zhou, “Metro: a generic graph neural network framework for multivariate time series forecasting,” *VLDB*, vol. 15, no. 2, pp. 224–236, 2021.

[32] L. Beretta and A. Santaniello, “Nearest neighbor imputation algorithms: a critical evaluation,” *BMC medical informatics and decision making*, vol. 16, no. 3, pp. 197–208, 2016.

[33] F. V. Nelwamondo, S. Mohamed, and T. Marwala, “Missing data: A comparison of neural network and expectation maximization techniques,” *Current Science*, pp. 1514–1521, 2007.

[34] A. Cichocki and A.-H. Phan, “Fast local algorithms for large scale nonnegative matrix and tensor factorizations,” *IEICE transactions on fundamentals of electronics, communications and computer sciences*, vol. 92, no. 3, pp. 708–721, 2009.

[35] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, “Brits: Bidirectional recurrent imputation for time series,” *NeurIPS*, vol. 31, 2018.

[36] J. Yoon, J. Jordon, and M. Schaar, “Gain: Missing data imputation using generative adversarial nets,” in *ICML*. PMLR, 2018, pp. 5689–5698.

[37] W. Du, D. Côté, and Y. Liu, “Saits: Self-attention-based imputation for time series,” *Expert Systems with Applications*, vol. 219, p. 119619, 2023.

[38] Y. Wu, X. Miao, Z. Li, S. He, X. Yuan, and J. Yin, “An efficient generative data imputation toolbox with adversarial learning,” in *ICDE*. IEEE, 2023, pp. 3651–3654.

[39] Y. Li, R. Yu, C. Shahabi, and Y. Liu, “Diffusion convolutional recurrent neural network: Data-driven traffic forecasting,” in *ICLR*, 2018.

[40] P. Bansal, P. Deshpande, and S. Sarawagi, “Missing value imputation on multidimensional time series,” *VLDB*, vol. 14, no. 11, pp. 2533–2545, 2021.

[41] A. Cini, I. Marisca, and C. Alippi, “Filling the g_ap_s: Multivariate time series imputation by graph neural networks,” in *ICLR*, 2021.

[42] I. Marisca, A. Cini, and C. Alippi, “Learning to reconstruct missing data from spatiotemporal graphs with sparse observations,” *NeurIPS*, vol. 35, pp. 32 069–32 082, 2022.

[43] J.-Y. Franceschi, A. Dieuleveut, and M. Jaggi, “Unsupervised scalable representation learning for multivariate time series,” *NeurIPS*, vol. 32, 2019.

[44] L. Yang and S. Hong, “Unsupervised time-series representation learning with iterative bilinear temporal-spectral fusion,” in *ICML*. PMLR, 2022, pp. 25 038–25 054.

[45] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, “Graph wavenet for deep spatial-temporal graph modeling,” in *IJCAI*, 2019, pp. 1907–1913.

[46] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” in *ICLR*, 2018.

[47] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, “Graph transformer networks,” *NeurIPS*, vol. 32, 2019.

[48] S. M. Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, S. Wu, C. Smyth, P. Poupart, and M. Brubaker, “Time2vec: Learning a vector representation of time,” *arXiv preprint arXiv:1907.05321*, 2019.

[49] J. D. M.-W. C. Kenton and L. K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *NAACL-HLT*, 2019, pp. 4171–4186.

[50] Y. Zheng, X. Yi, M. Li, R. Li, Z. Shan, E. Chang, and T. Li, “Forecasting fine-grained air quality based on big data,” in *KDD*, 2015, pp. 2267–2276.

[51] A. Zeng, M. Chen, L. Zhang, and Q. Xu, “Are transformers effective for time series forecasting?” in *AAAI*, vol. 37, no. 9, 2023, pp. 11 121–11 128.

[52] I. R. White, P. Royston, and A. M. Wood, “Multiple imputation using chained equations: issues and guidance for practice,” *Statistics in medicine*, vol. 30, no. 4, pp. 377–399, 2011.

[53] I. Loshchilov and F. Hutter, “Sgdr: Stochastic gradient descent with warm restarts,” in *ICLR*, 2016.

[54] R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian *et al.*, “A cookbook of self-supervised learning,” *arXiv preprint arXiv:2304.12210*, 2023.