Spatio-temporal Multivariate Time Series Forecast with Chosen Variables

Anonymous authors
Paper under double-blind review

Abstract

Spatio-temporal multivariate time series forecasting (STMF) leverages historical observations of spatially distributed variables to predict their future values. In real-world applications, however, budget constraints often limit the number of deployed sensors, resulting in scenarios where only a small subset of variables can be observed at inference time. Existing studies on STMF with missing variables typically assume that the observed variables are pre-determined, leaving the critical problem of how to optimally select input variables unexplored. To fill this gap, we study a new problem, termed *STMF with chosen variables (STCV)*, which aims to jointly select m out of n variables as model input and forecast future values for all variables. We propose a unified framework that tightly couples variable selection and model optimization for both forecasting accuracy and efficiency. Its design novelty centers at its learnable ability of jointly pruning model input (which reduces data collection cost with fewer variables) and pruning model parameters (which improves model efficiency), while optimizing model parameters for forecast accuracy, in contrast to the prior STMF models whose technical designs centered around optimizing model parameters alone. Extensive experiments demonstrate that the proposed method consistently outperforms state-of-the-art baselines in both forecasting accuracy and computational efficiency, highlighting the importance of jointly optimizing variable selection and spatio-temporal forecasting under sparse sensing budgets. Our source code is available at: `https://anonymous.4open.science/r/NLT-5E56/`

1 Introduction

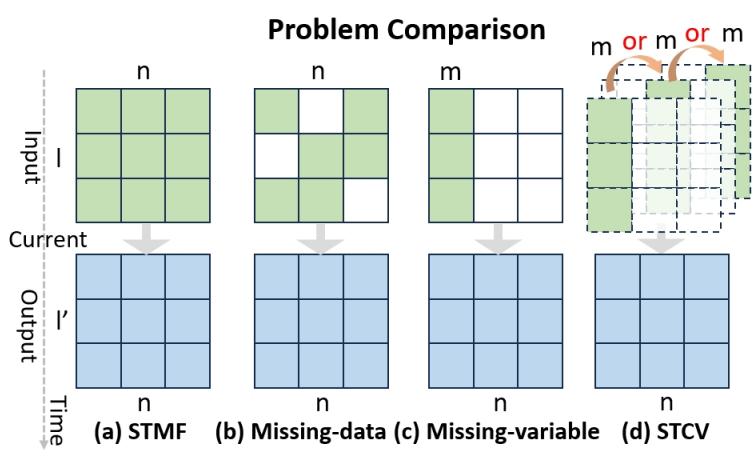

Figure 1: Problem comparison: (a) STMF, using $n \times l$ input of the past to forecast $n \times l'$ output of the future, (b) STMF with missing data, where some values in the input are missing, (c) STMF with missing variables, where some variables are missing from the input due to budget constraint, using $m \times l$ input to forecast $n \times l'$ output, (d) STCV, allowing optimal selection of the m variables in the input to maximize forecast accuracy.

Spatio-temporal multivariate time series forecasting (STMF) aims to predict future values of spatially distributed variables from recent observations, and has been widely studied in applications such as traffic forecasting and air quality monitoring (Liu et al., 2023a; Lee & Ko, 2024; Li & Zhu, 2021; Jiang et al., 2023b). Figure 1(a) illustrates the standard STMF setting, where past observations of all n variables are available as model input.

To relax this assumption, prior work has investigated STMF with *missing data*. As shown in Figure 1(b), STMF with missing data focuses on handling sporadically missing values in the input (Chen et al., 2023a; Cini et al., 2021; Zhang et al., 2023; Tang et al., 2020; Yu et al., 2025). Figure 1(c) further depicts STMF with *missing variables*, where only $m < n$ variables are observable during deployment (Gupta et al., 2023; Su et al., 2024; Yu et al., 2024). However, existing methods in both settings assume that the observed variables are *pre-determined*.

In many city-scale sensing deployments, only a small fraction of locations can be instrumented permanently due to budget constraints. For example, traffic sensing hardware and installation can cost in the order of *tens of thousands of dollars per intersection* (e.g., video-based systems or inductive-loop (Retail Sensing, 2024; Insider, 2026)), making full coverage across hundreds or thousands of intersections financially prohibitive. Yet, during *model development*, one can still collect historical data for all locations using temporary sensors at cost of hundreds of dollars a piece (LLC, 2026) which are intended for short-term use of training data collection, not for long-term permanent installation. The cost saving can be substantial if one can drive m down to a small fraction of n such as 10% (as our evaluation will achieve), where m out of n allocations are instrumented with permanent sensors. Under such a regime, the choice of the m variables for permanentn installation becomes a first-class design decision that directly determines forecasting quality, rather than an exogenous assumption by the prior work (Gupta et al., 2023; Su et al., 2024; Yu et al., 2024). This motivates **STMF with chosen variables (STCV)**: a new problem that jointly selects m input variables under a sensing budget while forecasting all n variables, as illustrated in Figure 1(d).

STCV presents several challenges. First, variable selection is combinatorial, rendering exhaustive search infeasible. Second, accurate forecasting requires extrapolating from sparse observations to a large set of unobserved variables while remaining computationally efficient. Third, dynamically changing the selected variables during training introduces instability and catastrophic forgetting.

To address these challenges, we propose *Variable-Parameter Iterative Pruning (VIP)*, a unified framework that tightly couples variable selection and model optimization. The design novelty of the framework centers at its learnable ability of jointly pruning *model input* (which reduces data collection cost with fewer variables) and *model parameters* (which improves model efficiency), while optimizing model parameters for forecast accuracy under a given budget, in contrast to the prior STMF models whose technical designs centered around optimizing model parameters alone. VIP progressively prunes uninformative variables and model parameters, propagates information from selected to unselected variables via dynamic extrapolation, and stabilizes training using prioritized replay. Experiments on five real-world traffic and air-quality datasets demonstrate that VIP achieves superior accuracy–efficiency trade-offs compared to state-of-the-art methods, validating the necessity of joint optimization in STCV.

2 Related Works

Most existing work studies the standard spatio-temporal multivariate forecasting problem shown in Figure 1(a), where all variables are observable. Representative models include recurrent neural networks (RNNs), CNNs or MLPs with graph neural networks (GNNs), neural differential equations (NDEs), transformers, and mixture-of-experts (MOE) based methods (Zhang et al., 2017; Yu et al., 2017; Li et al., 2018; Wu et al., 2021b; Li & Zhu, 2021; Jiang et al., 2023b; Yi et al., 2023; Fang et al., 2021; Liu et al., 2023b;a; Jiang et al., 2023a; Shazeer et al., 2017; Lee & Ko, 2024).

Some recent studies consider missing-data forecasting as shown in Figure 1(b), aiming to recover intermittently missing measurements caused by sensor failure or incomplete collection (Li et al., 2023; Cini et al., 2021; Chen et al., 2023a; Tang et al., 2020; Zhang et al., 2023). However, these methods still assume that sensors are deployed at all locations. Spatial extrapolation methods, such as non-negative matrix factorization and

GNN-based kriging methods, can estimate missing values from other variables within the same time period, but they do not address future forecasting under sparse sensing (Lee & Seung, 2000; Wu et al., 2021a; Zheng et al., 2023; Hu et al., 2023; Appleby et al., 2020).

More closely related are methods for STMF with missing variables, as shown in Figure 1(c), such as Frigate, STSM, and GinAR (Gupta et al., 2023; Su et al., 2024; Yu et al., 2024). These methods assume that the observed variables are pre-determined. In contrast, our work treats variable selection itself as part of the learning problem.

Sensor deployment has also been studied in wireless sensor networks (WSNs), where the objective is to optimize placement for sensing coverage and communication connectivity (Priyadarshi et al., 2020; Abidin et al., 2017; Yu et al., 2013; Al-Turjman et al., 2013; Gupta et al., 2016). This setting is fundamentally different from ours, since our goal is not coverage, but selecting informative variables for future spatio-temporal forecasting.

Another related direction is subset selection for efficient learning, which selects representative training samples to reduce computation while maintaining performance (Pooladzandi et al., 2022; Yang et al., 2023; Xia et al., 2022). However, these methods mainly focus on text or vision tasks rather than spatio-temporal forecasting under sensing budgets.

Finally, our VIP framework is also related to experience replay (ER) and prioritized experience replay (PER), which are widely used in continual learning to mitigate catastrophic forgetting (Zhang & Sutton, 2017; Aljundi et al., 2019; Kirkpatrick et al., 2017; Schaul, 2015). We adapt this line of work to stabilize iterative variable-parameter pruning in spatio-temporal forecasting.

In summary, this is the first work on STMF with chosen variables that jointly optimizes variable selection and parameter reduction for both forecasting accuracy and efficiency.

3 Preliminaries

3.1 Problem Definition

We consider a set of n spatially distributed variables, each associated with a time series. Let N denote the set of all variables, and let $M \subset N$ be a subset of m variables selected as model input. Given past observations $\mathcal{X}_{M,T} \in \mathbb{R}^{m \times l}$ over a time window $T = \{t - l + 1, \ldots, t\}$, the goal is to forecast future values $\mathcal{X}_{N,T'} \in \mathbb{R}^{n \times l'}$ over $T' = \{t + 1, \ldots, t + l'\}$ for all variables, where $\mathcal{X}_{M,T} \in \mathbb{R}^{m \times l}$ and $\mathcal{X}_{N,T'} \in \mathbb{R}^{n \times l'}$. A spatial adjacency matrix $A \in \mathbb{R}^{n \times n}$ encodes connectivity information. For example, 1 (or 0) indicates that there is (or not) a road connecting two locations (Song et al., 2020).

STMF with Chosen Variables (STCV): We define *Spatio-Temporal Multivariate Forecasting with Chosen Variables (STCV)* as the problem of jointly selecting $m \ll n$ input variables and forecasting future values for all variables. When $m = n$, STCV reduces to standard STMF (Jiang et al., 2023c; Song et al., 2020; Liu et al., 2023a; Lee & Ko, 2024). When $m < n$ and M is fixed, it corresponds to STMF with missing variables (Yu et al., 2024; Gupta et al., 2023). In contrast, STCV treats variable selection as part of the learning problem.

Following prior work (Yu et al., 2024; Gupta et al., 2023), we assume that full historical data are available during training, while only variables in M are observed at deployment, which has been justified in the introduction.

3.2 Temporal Embeddings and Node Embeddings

Learnable temporal embeddings, denoted as E_N^{temp}, are used to capture temporal patterns. They are application-dependent. For example, for the road traffic application, they may include time-of-day (tod) embeddings and day-of-week (dow) embeddings (Shao et al., 2022; Liu et al., 2023a): Discretize a day into $D = 288$ time steps of 5-minute each, and a week into $W = 7$ days. Each time step in $[0, D)$ is mapped to a trainable time-of-day embedding vector in $\mathbb{R}^{d_{tod}}$, and each day in $[0, W)$ is mapped to a trainable day-of-week embedding vector in $\mathbb{R}^{d_{dow}}$. Given the value matrix $\mathcal{X}_{N,T} \in \mathbb{R}^{n \times l}$, where n is the number of variables and l is

the length of time T, we assign temporal embeddings to each entry $\mathcal{X}_{i,j}$. This yields a time-of-day embedding matrix $E_N^{tod} \in \mathbb{R}^{n \times l \times d_{tod}}$ and a day-of-week embedding matrix $E_N^{dow} \in \mathbb{R}^{n \times l \times d_{dow}}$. We aggregate the above learnable temporal embeddings as $E_N^{temp} = E_N^{tod} || E_N^{dow}$. In addition, each node v in the graph has a learnable node embedding vector in $\mathbb{R}^{d_v}$; the node embedding matrix is denoted as $E_N^{node} \in \mathbb{R}^{n \times l \times d_v}$, which repeats the node embeddings across time T of length l.

4 Framework of Variable-Parameter Iterative Pruning

4.1 Pre-training A Base STMF Model

We begin with a base STMF model to address the spatio-temporal multivariate time series forecast problem. The model is denoted as $F(\mathcal{X}_{N,T}, A, \Theta)$, which takes past observations $\mathcal{X}_{N,T}$ from all variables N together with A as input to forecast future values $\mathcal{X}_{N,T'}$ as output, with the learned model parameters Θ. The model begins with a Multi-Layer Perceptron (MLP) that processes the input data $\mathcal{X}_{N,T}$ and produce a feature embedding $E_N^f \in \mathbb{R}^{n \times l \times d}$. It then concatenates this with the learnable temporal embeddings E_N^{temp} and the learnable node embeddings E_N^{node} to form aggregated features $E_N^{all} \in \mathbb{R}^{n \times l \times q}$, where q is the number of dimensions in the attention layers, and the details of E_N^{temp} and E_N^{node} are provided in Preliminaries 3.2. E_N^{all} is then passed through a number of temporal attention layers, each denoted as ATT_t, and a number of spatial attention layers, each denoted as ATT_s, to produce a high-dimensional representation $H_{N,T} \in \mathbb{R}^{n \times l \times q}$.

$$E_N^f = MLP(\mathcal{X}_{N,T}) \tag{1}$$

$$E_N^{all} = E_N^f || E_N^{node} || E_N^{temp} \tag{2}$$

$$H_{N,T} = ATT_s(...ATT_s(ATT_t(...ATT_t(E_N^{all})...))...) \tag{3}$$

where ATT_t and ATT_s are designed to capture the spatio-temporal correlations. Let $H_{N,T}^t \in \mathbb{R}^{n \times l \times q}$ be the output of the first temporal attention, $ATT_t(E_N^{all})$. It is calculated as follows: We obtain query Q^t, key K^t and value V^t matrices through the temporal attention layer, where W_Q^t, W_K^t, $W_V^t \in \mathbb{R}^{q \times q}$ are their learnable parameters, respectively:

$$Q^t = E_N^{all} W_Q^t,\ K^t = E_N^{all} W_K^t,\ V^t = E_N^{all} W_V^t \tag{4}$$

$$H_{N,T}^t = Softmax\left(\frac{Q^t (K^t)^\top}{\sqrt{d_t}}\right) V^t \tag{5}$$

where d_k is the number of heads in the multi-head mechanism of the attention. We repeat for additional temporal attention layers, each with its input being the output of the previous layer, using that input to substitute E_N^{all} in Eq. equation 4-equation 5. Let $H_{N,T}^s \in \mathbb{R}^{n \times l \times q}$ be the output of the spatial attention, $ATT_s((H_{N,T}^t)^\top)$, where $\top$ is for transpose. Each spatial attention is implemented similarly, with $ATT_s(H) = ATT_t(H^\top))$ for any input H. Let $H_{N,T}^s \in \mathbb{R}^{n \times l \times q}$ be the output after all spatial attentions. Finally, we use an MLP to map $\hat{H}_{N,T}^s \in \mathbb{R}^{n \times l \times q}$ to the forecast output $\hat{\mathcal{X}}_{N,T'} \in \mathbb{R}^{n \times l'}$:

$$\hat{\mathcal{X}}_{N,T} = MLP(H_{N,T}^s) \tag{6}$$

4.2 Variable-Parameter Iterative Pruning

4.2.1 From STMF to STCV

The proposed STMF model is designed to forecast future values of all time series in N based on their past values. It relies on dense neural networks, i.e., embeddings and attention mechanisms, and requires all variables in the model input.

We regard STCV as a variable reduction problem and produce its model from the pretrained STMF model by iteratively pruning insignificant variables to select a final desired set M of variables. With a reduced number of variables, we also prune insignificant dimensions of model parameters to reduce the model size and

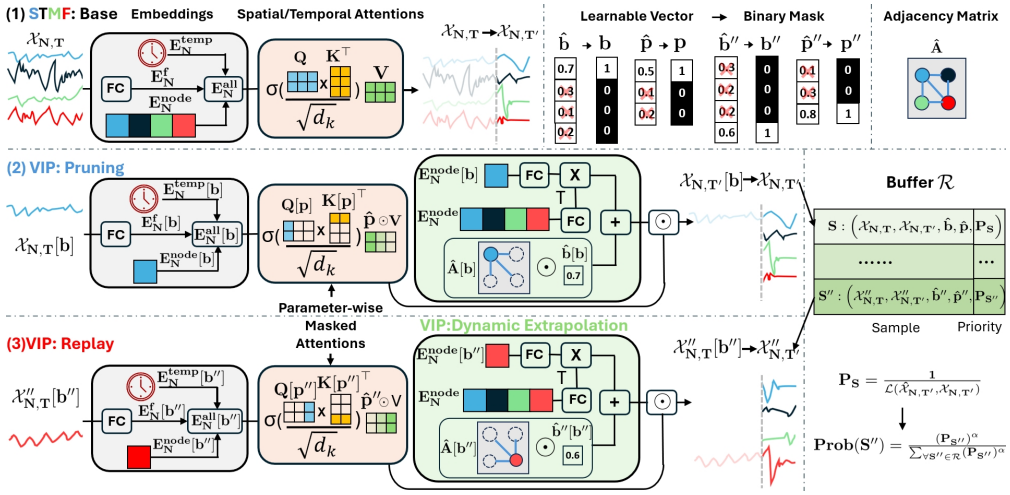

Figure 2: (1) the base STMF model, shown at the top left of the figure. (2) VIP: pruning, shown at the middle left. It learns from the current sample, $S : \left(\mathcal{X}_{N,T}, \mathcal{X}_{N,T'}, \hat{b}, \hat{p}, P_S\right)$, and then stores the sample in a buffer, shown at the middle right. (3) VIP: replay, shown at the bottom left. It learns from a replayed sample that is retrieved from the buffer, denoted as $S'' : \left(\mathcal{X}''_{N,T}, \mathcal{X}''_{N,T'}, \hat{b}'', \hat{p}'', P_{S''}\right)$, shown at the bottom right. (4) learnable variable and parameter vectors $\hat{b}$, $\hat{p}$, along with their corresponding binary masks b, p, shown at the top right. (5) adjacency matrix $\hat{A}$, shown at the top right. In (1), (2) and (3) of the figure, we illustrate the temporal embeddings E_N^{temp}, the feature embedding E_N^f, and the node embeddings E_N^{node} (gray blocks); spatial or temporal attentions (orange block in (1)); parameter-wise masked attentions (orange blocks in (2) and (3)); and dynamic extrapolation (green blocks).

thus improve its inference efficiency. Furthermore, reducing input variables and parameter dimensions not only lowers computational cost but also effectively shrinks the combinatorial search space for optimal model configurations, thus enabling more efficient learning (Frankle & Carbin, 2018).

Below we propose a new learning framework for STCV, called *Variable-Parameter Iterative Pruning (VIP)*, where each iteration jointly prunes less-important variables and parameters and the process ends after a budgeted number m of variables are reached.

4.2.2 Overview

VIP jointly tackles the challenges of variable selection and model compression in the STCV setting. It progressively reduces the number of input variables and model parameters through an iterative masking strategy. VIP is composed of the three tightly integrated components:

Masked Variable-Parameter Pruning. We use learnable masks over variables and attention parameters to identify and eliminate less-informative inputs and model dimensions by quantile-based thresholding. Such a pruning mechanism is illustrated on the left side of Fig. 2.

Dynamic Extrapolation. We define a unified training objective that integrates forecast loss, replay loss, and stochastic regularization over mask vectors to balance convergence and exploration. We enable dynamic extrapolation by propagating the features learned from chosen variables in M to missing variables in M' via the node embeddings, as shown in the green blocks of Fig. 2.

Prioritized Variable-Parameter Replay. To alleviate catastrophic forgetting induced by dynamic pruning, we employ a replay buffer that retains previously encountered low-loss samples, prioritizing those deemed more informative. This mechanism is depicted in the bottom-right corner of Fig. 2.

Together, these components form a cohesive learning framework that jointly optimizes variable selection and parameter sparsity, enabling scalable and accurate spatio-temporal forecast.

4.2.3 Masked Variable-Parameter Pruning

We first introduce variable mask and parameter mask. We then explain how to perform variable-parameter pruning.

Variable Mask. We use a binary variable mask $b \in \{0,1\}^n$ to specify variable selection: variable i is selected iff $b_i = 1$, where b_i is the ith masking bit of b, $\forall i \in [0, n)$. Given a value matrix $\mathcal{X}_{N,T}$ and a variable mask b, we define $\mathcal{X}_{N,T}[b]$ as a sub-matrix with only those time series $\mathcal{X}_{i,T}$ whose corresponding masking bits in b are 1's, i.e., $b_i = 1$, where $[b]$ is a *masking operator*: When it is applied to a matrix of size $n \times ...$, it selects the rows whose masking bits are ones; when it is applied to a vector of size n, it selects the elements whose masking bits are ones. For another example, given an adjacency matrix A and a variable mask b, $A[b]$ is a sub-adjacency matrix with only those rows (variables) whose mask bits in b are 1's.

Parameter Mask. Given the set Θ of parameters in an STMF model, with each of its attentions having q dimensions of parameters, and a binary parameter mask $p \in \{0,1\}^q$ that is applied to all attentions, we define $\Theta[p]$ as a subset of parameters with only those dimensions (in each attention) whose mask bits in p are 1's, i.e., $p_j = 1$, where p_j is the jth masking bit in p. In the sequel, this parameter masking operation $[p]$ will also be applied to other parameter sets of q dimensions.

Iterative Pruning. Given a pre-trained STMF model $F(\mathcal{X}_{N,T}, A, \Theta)$, we will adopt $F(\mathcal{X}_{N,T}[b], A[b], \Theta[p])$ as an STCV model that forecasts the future values of all variables, but with fewer input variables $\mathcal{X}_{N,T}[b]$ and fewer model parameters $\Theta[p]$, where b and p are iteratively optimized.

Starting from the initial masks of b^0 and p^0 whose bits are all ones, our task is to learn b^k and p^k together iteratively, where k is the number of iterations, such that the final masks, denoted as b^* and p^*, achieve the best performance for STCV, where $\|b^*\|_0 = m$, and $\|p^*\|_0 = q'$ for a desired target number q' of parameter dimensions in each attention, with $q' < q$.

Let $\hat{b} \in \mathbb{R}^n$ be a learnable variable importance vector, and $\hat{p} \in \mathbb{R}^q$ be a learnable parameter importance vector. These vectors are optimized jointly with the model parameters during training. The binary variable mask b^k and parameter mask p^k at the k-th iteration are derived from $\hat{b}$ and $\hat{p}$ via quantile-based thresholding.

The vector $\hat{b}$ is initialized based on the normalized graph structure: $\hat{b} = \left[\sum_{j=1}^{n} \hat{A}_{0j}, \ldots, \sum_{j=1}^{n} \hat{A}_{nj}\right]^\top$, where $\hat{A} = \tilde{D}^{-\frac{1}{2}}(A+I)\tilde{D}^{-\frac{1}{2}}$ is the symmetrically normalized adjacency matrix with self-loops. The i-th value of $\hat{b}$ is denoted $\hat{b}_i$, for $i \in [0, n)$. The vector $\hat{p}$ is initialized from a standard normal distribution, with $\hat{p}_j$ denoting the j-th element, for $j \in [0, q)$.

The binary masks b^k and p^k are updated as follows:

$$b_i^k = \begin{cases} 0, & \text{if } |\hat{b}_i| \leq \text{Quantile}(|\hat{b}[b^{k-1}]|, r_b) \\ 1, & \text{otherwise} \end{cases} \tag{7}$$

$$p_j^k = \begin{cases} 0, & \text{if } |\hat{p}_j| \leq \text{Quantile}(|\hat{p}[p^{k-1}]|, r_p) \\ 1, & \text{otherwise} \end{cases} \tag{8}$$

Here, $[\,\cdot\,]$ denotes the masking operator that selects elements whose masking bits are 1s. The operator $|\cdot|$ takes element-wise absolute values, and $\text{Quantile}(x, \beta)$ denotes the β-quantile of vector x. The scalars r_b and r_p are the pruning rates for variable and parameter masks, respectively.

From Eq. equation 7-equation 8, after the k-th iteration, we will retain $\|b^k\|_0$ variables, where $\|b^k\|_0 = max\{\lfloor n \times (1-r_b)^k \rfloor, 1\}$, and retain $\|p^k\|_0$ dimensions of parameters in each attention, where $\|p^k\|_0 = max\{\lfloor q \times (1-r_p)^k \rfloor, 1\}$.

Consider a value matrix of all variables, $\mathcal{X}_{N,T} \in \mathbb{R}^{n \times l}$. With the retained variable mask b^k, the masked value matrix is $\mathcal{X}_{N,T}[b^k] \in \mathbb{R}^{\|b^k\|_0 \times l}$. From Eq. equation 1-equation 2 in the STMF model, we can calculate its aggregated feature $E_N^{all}[b^k] \in \mathbb{R}^{\|b^k\|_0 \times l \times q}$.

To effectively retain the model's learning power while reducing the model parameters, we design a parameter-wise masked attention, which is shown in the orange blocks of Fig. 2. It applies the learnable parameter vector $\hat{p}$ and the binary parameter mask p^k to each attention in Eq. equation 4 and equation 5 as follow:

$$\hat{Q}^t = \left(E_N^{all}[b^k]\right)\left(W_Q^t[p^k]\right) \tag{9}$$

$$\hat{K}^t = \left(E_N^{all}[b^k]\right)\left(W_K^t[p^k]\right) \tag{10}$$

$$\hat{V}^t = \hat{p} \odot \left(E_N^{all}[b^k]W_V^t\right) \tag{11}$$

$$H_{N,T}^t[b^k] = Softmax\left(\frac{\hat{Q}^t(\hat{K}^t)^\top}{\sqrt{d_t}}\right)\hat{V}^t \tag{12}$$

where $W_Q^t[p^k], W_K^t[p^k] \in \mathbb{R}^{q\times\|p^k\|_0}$ and $\odot$ is the element-wise product. Spatial attentions are handled similarly except that transpose is performed on the input, i.e., $\hat{K}^t$.

4.2.4 Dynamic Extrapolation

To ensure accurate forecast, we must propagate learned representations from the variables that are selected to the variables that are not selected. We design a dynamic extrapolation mechanism, which is shown in the green blocks of Fig. 2, that constructs a global information bridge between variables in M and those in M'. It operates in two stages: (1) computing a soft similarity-based attention matrix over variable embeddings, and (2) fusing it with the original masked adjacency matrix to allow knowledge transfer across all variables.

Similarity-based Extrapolation Attention. We first compute a global attention matrix $\hat{B} \in \mathbb{R}^{m\times n}$ based on node embedding similarity:

$$\hat{B} = \sigma\left(FC(E_N^{node}[b^k]) \cdot FC(E_N^{node})^\top\right) \tag{13}$$

where $FC(\cdot)$ is a shared learnable linear projection. The activation $\sigma(\cdot)$ is the GeLU function (Hendrycks & Gimpel, 2016). This attention matrix $\hat{B}$ serves as a dense bridge from the variables selected by mask b^k to all variables.

Fusion with Masked Adjacency. We then combine $\hat{B}$ with the masked adjacency matrix to form a final extrapolated adjacency:

$$\hat{A}' = \hat{b}[b^k] \odot \hat{A}[b^k] + \hat{B} \tag{14}$$

Here, $\hat{A}[b^k]$ is the symmetrically normalized adjacency matrix for the selected variables, and $\hat{b}[b^k]$ acts as a learned variable-level weight. This fused matrix $\hat{A}' \in \mathbb{R}^{m\times n}$ enables spatial message passing from the selected variables to all variables.

Feature Propagation. The extrapolated representation for all variables is computed as:

$$H_{N,T} = \hat{A}'^\top \cdot H_{N,T}[b^k] \tag{15}$$

This mechanism allows latent features from the selected variables to influence other variables via learned attention, enhancing forecast quality under sparse input.

Finally, an MLP is used to map the high-dimensional representation $H_{N,T}$ to the forecast output $\hat{\mathcal{X}}_{N,T'}$ as in Eq. equation 6, and the loss function is given below

$$\mathcal{L}_{main}(\cdot,\cdot) = \mathcal{L}\left(F(\mathcal{X}_{N,T}[b^k], A[b^k], \Theta[p^k]), \mathcal{X}_{N,T'}\right). \tag{16}$$

4.2.5 Prioritized Variable-Parameter Replay

Within the kth iteration of VIP, mask b^k for variable selection and mask p^k for parameter selection will be updated together with the learnable vectors $\hat{b}$ and $\hat{p}$ during training. If the two masks differ significantly after each update, the model might not retain the significant patterns from the previous training, which is a phenomenon known as catastrophic forgetting (Kirkpatrick et al., 2017). The experience replay (ER) methods (Zhang & Sutton, 2017), which store some historical samples in a buffer and replay them to the current model using a first-in-first-out strategy, can mitigate the problem of catastrophic forgetting. Furthermore, the prioritized experience replay method (Schaul, 2015) assigns priority to the samples for replay and achieves better performance.

Motivated from the prior ER work, we introduce the prioritized variable-parameter replay method (PVR) to mitigate catastrophic forgetting in our proposed VIP. After a new sample is applied in training, PVR stores this sample $S : \left(\mathcal{X}_{N,T}, \mathcal{X}_{N,T'}, \hat{b}, \hat{p}, P_S\right)$ in a buffer $\mathcal{R}$, which can keep up to $\|\mathcal{R}\|$ samples, where the priority P_S of the current sample S is given as

$$P_S = \frac{1}{\mathcal{L}_{main}(\cdot,\cdot)}. \tag{17}$$

Intuitively, in the context of VIP where variable-parameter pruning can cause learned patterns to be pruned from the model, low-loss samples should be given priorities to combat catastrophic forgetting. This assumption is supported by experimental evaluation in Section 5.8, confirming that samples yielding lower $\mathcal{L}_{main}$ help retain significant patterns.

When inserting a new sample to $\mathcal{R}$, if $\mathcal{R}$ is full, PVR will probabilistically select an existing sample from $\mathcal{R}$, replay it, and then remove it. The probability for a sample $S' \in \mathcal{R}$ to be selected is

$$Prob(S') = \frac{(P_{S'})^\alpha}{\sum_{\forall S' \in \mathcal{R}} (P_{S'})^\alpha} \tag{18}$$

where α is the hyperparameter. Let $S'' : \left(\mathcal{X}''_{N,T}, \mathcal{X}''_{N,T'}, \hat{b}'', \hat{p}'', P_{S''}\right)$ denote the sample selected for replay. Its loss function is

$$\mathcal{L}_{replay}(\cdot,\cdot) = \mathcal{L}\left(F(\mathcal{X}''_{N,T}[b''^k], A[b''^k], \Theta[p''^k]), \mathcal{X}''_{N,T'}\right) \tag{19}$$

where the values of b''^k and p''^k are obtained from $\hat{b}''$ and $\hat{p}''$ through Eq. equation 7-equation 8 by replacing $\hat{b}''$ for $\hat{b}$ and $\hat{p}''$ for $\hat{p}$.

4.2.6 Co-optimized Variable-Parameter Loss

In practice, there is a balance between adjusting and stabilizing the selected variables and model parameters. On the one hand, adjusting more aggressively introduces more combinations of selection. On the other hand, it makes the selection harder to converge. Stabilizing the masks aggressively accelerates convergence, but it may lead to suboptimal forecast performance.

To address this issue, we introduce a new *co-optimized variable-parameter loss* function $\mathcal{L}_{sum}$. This loss consists of three components: the main loss $\mathcal{L}_{main}$, which aims to find the optimal selection of of variables and model parameters; the replay loss $\mathcal{L}_{replay}$, which helps the model retain knowledge of previous samples; the random regularization of $\hat{b}$ and $\hat{p}$, which encourages sparse and diverse pruning decisions during training, acting as a stochastic exploration mechanism to avoid premature convergence to suboptimal masks. The co-optimized variable-parameter loss is defined as

$$\begin{aligned}\mathcal{L}_{sum}(\cdot,\cdot) =& \mathcal{L}_{main}(\cdot,\cdot) + \gamma_1 \mathcal{L}_{replay}(\cdot,\cdot) + \\ & \gamma_2 \|\hat{b}[r_1]\|_1 + \gamma_3 \|\hat{p}[r_2]\|_1\end{aligned} \tag{20}$$

where the trainable variable vector $\hat{b}$ is optimized on a subset of variables chosen by a random mask $r_1 \in \{0,1\}^n$ with $\|r_1\|_0$ being a small number, the trainable parameter vector $\hat{p}$ is optimized on a subset of parameter dimensions chosed by another random mask $r_2 \in \{0,1\}^q$ with $\|r_2\|_0$ being a small number, and γ_1, γ_2, and γ_3 are the hyperparameters.

Algorithm 1 k-th iteration of variable-parameter iterative pruning in model learning

Input: Value matrix $\mathcal{X}_{N,T}$, adjacency matrix A, STMF model $F(\cdot, A, \Theta)$, trainable variable mask $\hat{b}$ and parameter mask $\hat{p}$, buffer $\mathcal{R}$, learning rate λ
Output: Sparsified variable mask b^k, parameter mask p^k, model parameters Θ
$\mathcal{R} \leftarrow \{\}$
repeat
  // Compute binary masks for current sample
  Compute variable mask b^k and parameter mask p^k from $\hat{b}$ and $\hat{p}$ by Eq. equation 7-equation 8
  // Forward pass on current sample
  Forward $F(\mathcal{X}_{N,T}[b^k], A[b^k], \Theta[p^k])$ to compute the main loss $\mathcal{L}_{\text{main}}(\cdot, \cdot)$ by Eq. equation 16
  // Replay sample selection and loss computation
  Select a replay sample $(\mathcal{X}''_{N,T}, \mathcal{X}''_{N,T'}, \hat{b}'', \hat{p}'')$ from $\mathcal{R}$ using Eq. equation 17 and Eq. equation 18
  Compute variable mask $b^{''k}$ and parameter mask $p^{''k}$ from $\hat{b}''$ and $\hat{p}''$ by Eq. equation 7-equation 8
  Forward $F(\mathcal{X}''_{N,T}[b^{''k}], A[b^{''k}], \Theta[p^{''k}])$ to compute the replay loss $\mathcal{L}_{\text{replay}}(\cdot, \cdot)$ by Eq. equation 19
  // Total loss and gradient update
  Obtain total loss $\mathcal{L}_{\text{sum}}(\cdot, \cdot)$ by Eq. equation 20
  Backpropagate to update $\Theta \leftarrow \Theta - \lambda \nabla_{\Theta} \mathcal{L}_{\text{sum}}(\cdot, \cdot)$
  Update $\hat{b} \leftarrow \hat{b} - \lambda \nabla_{\hat{b}} \mathcal{L}_{\text{sum}}(\cdot, \cdot)$
  Update $\hat{p} \leftarrow \hat{p} - \lambda \nabla_{\hat{p}} \mathcal{L}_{\text{sum}}(\cdot, \cdot)$
  // Store current sample into replay buffer
  Insert current sample $(\mathcal{X}_{N,T}, \mathcal{X}_{N,T'}, \hat{b}, \hat{p}, P_S)$ into $\mathcal{R}$
until all batches in epoch are processed

4.2.7 Complexity Analysis

We analyze the inference complexity of our VIP model and compare it to the base STMF model. Let m be the number of chosen variables, n the total number of variables, l the length of the time period in the model input (output), L the number of attension layers, q the original number of attention dimensions, and q' the after-prune number of attention dimensions.

Time Complexity. The total inference time complexity of the VIP model is:

$$\mathcal{O}\left(m \cdot l \cdot q + L \cdot m \cdot l \cdot q' \cdot q + m \cdot n \cdot l\right)$$

The three terms correspond to: (1) variable-wise feature embedding, (2) attention computation, and (3) extrapolation from the chosen variables to all variables. In contrast, the original STMF model has a time complexity of:

$$\mathcal{O}\left(n \cdot l \cdot q + L \cdot n \cdot l \cdot q^2\right)$$

By reducing the number of input variables from n to m, and attention dimensions from q to q', our model achieves significant computational savings, especially when $m \ll n$ and $q' \ll q$.

Memory Complexity. The total memory complexity of the STCV model is:

$$\mathcal{O}\left(m \cdot l \cdot q + L \cdot q' \cdot q + n + L \cdot q\right)$$

where the terms corresponds to the embeddings, the pruned attention layers, and the trainable masks, $\hat{b}$ and $\hat{p}$. In comparison, the memory complexity of the STMF model is:

$$\mathcal{O}\left(n \cdot l \cdot q + L \cdot q^2\right)$$

The STCV model saves both space and time by working with a sparse set of input variables and a sparse set of parameters during inference, while still forecasting for all n variables. The overall algorithm is presented in Algorithm 1.

Table 1: Performance comparison over five datasets, where n is the total number of locations and m is the number of chosen locations.

Models	PEMS04			PEMS08			PEMSBAY			METRLA			AQI		
	deployment ratio $m/n = 10\%$**, sparsity ratio** $1 - m/n = 90\%$														
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Matrix Factorization	92.13	141.41	124.29	75.80	131.17	82.16	6.51	14.29	21.09	15.61	33.39	48.51	67.63	115.43	130.18
STID	94.72	123.30	149.68	81.64	113.20	86.54	3.78	8.65	10.07	6.24	12.06	23.17	25.97	40.28	62.41
DSformer	40.31	59.18	25.62	38.79	55.34	32.61	3.72	9.06	10.23	5.02	10.15	17.69	23.17	35.72	57.31
STGODE	30.91	48.25	25.59	26.73	41.20	23.20	3.29	8.22	9.76	4.48	8.49	14.79	23.78	38.69	47.10
STSGCN	34.42	50.51	22.09	28.51	46.31	17.94	3.23	7.73	8.41	4.39	8.45	12.61	25.61	40.74	59.96
MegaCRN	36.14	52.17	23.42	34.52	52.75	24.64	3.54	8.25	9.23	4.58	8.72	13.54	22.61	35.74	53.28
ST-SSL	28.89	44.11	19.07	24.87	40.87	17.46	3.05	6.91	7.86	3.96	7.98	11.82	18.98	32.97	38.59
TESTAM	29.11	44.13	19.11	24.81	38.43	15.43	2.85	6.91	7.80	3.85	7.79	11.25	16.98	32.12	37.62
STAEFormer	28.11	43.32	18.35	24.31	39.25	15.63	2.72	6.58	6.70	4.07	8.43	12.05	17.47	32.60	39.09
STAEFormer + IGNNK	27.89	42.97	18.21	24.99	39.15	15.32	2.80	6.90	6.85	4.32	8.78	12.21	17.32	33.32	40.12
STAEFormer + STGNP	28.10	42.40	18.39	24.37	39.39	15.91	2.73	6.60	6.79	4.19	8.58	12.29	17.59	34.15	41.61
FourierGNN+GATGPT	32.16	48.83	23.65	29.11	41.44	20.61	3.28	7.44	9.07	4.33	8.37	12.28	20.18	34.17	50.94
LGnet	36.29	55.12	23.74	35.48	52.06	25.71	3.67	8.92	9.83	4.79	9.15	14.38	23.24	35.54	57.18
GC-VRNN	30.06	45.12	20.43	28.46	41.98	20.54	3.12	7.32	7.94	4.32	8.35	12.29	19.24	32.91	48.73
TriD-MAE	29.54	44.23	20.05	26.18	39.25	16.43	3.02	7.02	7.63	4.11	8.22	11.92	18.04	32.68	45.76
BiTGraph	28.71	43.37	19.08	25.01	39.06	16.18	2.83	6.75	7.42	3.91	8.03	11.78	17.06	31.85	38.17
GinAR	28.20	42.82	18.31	24.83	38.87	15.82	2.77	6.43	6.94	3.87	7.84	11.25	16.83	30.97	36.30
MaxConnectivity+GinAR	28.11	42.65	18.14	24.89	39.07	15.99	2.84	6.80	7.23	3.93	8.09	12.21	16.97	31.35	37.89
MaxValue+GinAR	28.04	42.43	18.67	24.59	38.27	15.95	2.67	6.30	6.73	3.78	7.83	11.20	16.71	31.54	37.75
Grid+GinAR	28.87	42.31	18.06	24.70	38.84	15.74	2.78	6.59	6.93	3.90	7.99	11.36	16.89	31.14	37.52
Metaheuristics+GinAR	30.54	46.83	21.12	26.98	41.62	17.81	3.10	7.19	8.22	4.51	8.93	13.45	19.37	34.97	39.33
VIP+Pre	22.99	38.51	16.23	18.62	31.98	13.01	2.43	4.89	5.79	3.80	7.91	11.05	16.23	30.03	35.92
VIP wo. pretrain	23.99	39.27	16.05	19.87	32.73	13.19	2.44	4.99	5.80	3.52	7.14	10.20	16.98	31.34	35.21
VIP w. pretrain	22.93	38.26	16.06	18.59	31.51	12.63	2.38	4.76	5.64	3.64	7.68	10.94	15.89	29.22	35.01

Table 2: Basic statistics of the datasets used in our experiments.

Dataset	**AQI**	**PEMS04**	**PEMS08**	**PEMS-BAY**	**METR-LA**
Area	China	CA, USA			
Time Span	1/1/2015 - 12/31/2022	1/1/2018 - 2/28/2018	7/1/2016 - 8/31/2016	1/1/2017 - 5/31/2017	3/1/2012 - 6/30/2012
Time Interval	1 hour	5 min			
Number of Locations, n	350	307	170	325	207
Number of Time Intervals	59,710	16,992	17,856	52,116	34,272

5 Experimental Evaluation

5.1 Settings and Datasets

Datasets: We use five widely used public datasets, including four road traffic datasets (METR-LA, PEMSBAY, PEMS04 and PEMS08) and an air quality dataset (AQI) [1], for the road traffic forecast application and the air quality forecast application in the introduction, respectively. The details of data statistics are shown in Table 2. We split METR-LA, PEMSBAY dataset into three subsets in 7:1.5:1.5 ratio for training, validation, and test and 3:1:1 for the other datasets.

These datasets are mapped to the STCV problem as follows: Each of the n locations in a dataset corresponds to a variable whose values are the time series of traffic (air quality) measurements at the location. After collecting the training/validation data subsets with cheap, temporary sensors from n locations (i.e., variables), we select m locations for permanent sensor deployment in the test data subsets for evaluation. These m

[1]The datasets are provided in the GinAR github repository at `https://github.com/GestaltCogTeam/GinAR/`

chosen locations correspond to the chosen variables in the model input; the other ($m' = n - m$) *unchosen locations* correspond to the missing variables not in the model input.

We define the sensor *deployment ratio* as $\frac{m}{n}$ and the deployment *sparcity ratio* as $1 - \frac{m}{n}$. In the application context of our evaluation description, we will use the term *location* for *variable* in the general STCV definition.

Hyperparameters: $m = 0.1n$ by default; hence, the deployment ratio is 10% and 90% by default — note that different settings of m and deployment (sparcity) ratios will be studied. $l = l' = 12$ for one hour forecast in the datasets. We use the Mean Absolute Error (MAE) as the loss function. All experiments are conducted on a server equipped with an AMD EPYC 7742 64-Core Processor @ 2.25 GHz, 500 GB RAM, and an NVIDIA A100 GPU with 80 GB memory.

We perform a systematic grid search over other hyperparameters, as summarized in Table 3. These hyperparameters are categorized into two groups: primary and secondary. Each hyperparameter is tested across four candidate values, and we explore all combinations within each group independently.

The primary group includes the node pruning rate r_b, parameter pruning rate r_p, size $|\mathcal{R}|$ of the replay buffer, priority coefficient α, learning rate, and batch size. The secondary group contains the loss-related coefficients $\gamma_1, \gamma_2, \gamma_3$, the number $|r_1|_0$ of selected elements in the node vector, and the number $|r_2|_0$ of selected elements in the parameter vector, as defined in Eq. 20.

For the remaining STMF hyperparameters, we follow the default settings established in prior work (Liu et al., 2023a), namely: $q = 152$, $d = 24$, $d_{\text{tod}} = 24$, $d_{\text{dow}} = 24$, $d_v = 80$, and $L = 6$.

Table 3: hyperparameter settings with the deployment rate of 10% on all datasets, where the bold values represent the best setting

Group	Hyperparameter	Value 1	Value 2	Value 3	Value 4
primary	r_b	0.05	**0.10**	0.20	0.40
	r_p	0.01	0.02	**0.05**	0.10
	$\|\mathcal{R}\|$	288	288*3	**288*7**	288*30
	α	0.2	0.4	**0.6**	0.8
	batch size	8	16	**64**	128
	learning rate	10^{-2}	$\mathbf{10^{-3}}$	5×10^{-4}	10^{-4}
secondary	γ_1	0.1	0.5	**1.0**	2.0
	γ_2	0.1	0.5	**1.0**	2.0
	γ_3	0.1	0.5	**1.0**	2.0
	$\|r_1\|_0$	0	1	**2**	5
	$\|r_2\|_0$	0	**1**	2	5

Baselines: The baselines for comparison include spatio-temporal forecast models, such as STGODE (Fang et al., 2021), ST-SSL (Ji et al., 2023), STID (Shao et al., 2022), STSGCN (Song et al., 2020), DSformer (Yu et al., 2023), MegaCRN (Jiang et al., 2023c), STAEFormer (Liu et al., 2023a), and TESTAM (Lee & Ko, 2024); two-stage spatial extrapolation models, which first perform spatio-temporal forecast (using past measurements from the m chosen locations to predict the future measurements at the m locations) and then perform spatial extrapolation (using the predicted future measurements at the m locations to extrapolate the future measurements at all n locations), such as DCRNN(Li et al., 2018)+GPT4TS(Zhou et al., 2023), DFDGCN(Li et al., 2024)+TimesNet(Wu et al., 2022), MTGNN(Wu et al., 2020)+GRIN(Cini et al., 2021), FourierGNN(Yi et al., 2023)+GATGPT(Chen et al., 2023b), STAEFormer+IGNNK (Wu et al., 2021a) and STAEFormer+STGNP (Hu et al., 2023); and the spatio-temporal imputation models, such as LGnet (Tang et al., 2020), GC-VRNN (Xu et al., 2023), TriD-MAE (Zhang et al., 2023), BiTGraph (Chen et al., 2023a) and GinAR (Yu et al., 2024) on STMF with missing variables, which is most related to our work; and a non-deep-learning Matrix Factorization method (Lee & Seung, 2000). Adapting the baselines to work for the problem of STCV, following GinAR (Yu et al., 2024), we set the values of unchosen variables in their input value matrix to 0s.

In addition, we extend comparison to cover sensor placement strategies from the wireless sensor network domain and other heuristic approaches by including four two-phase baselines, with the first phase selecting sensor locations (Meguerdichian et al., 2001; Gupta et al., 2016; Yu et al., 2024) and the second phase using GinAR (Yu et al., 2024) to perform forecast. These baselines are referred to as Grid(Meguerdichian et al., 2001)+GinAR(Yu et al., 2024), Metaheuristics(Gupta et al., 2016)+GinAR(Yu et al., 2024), MaxConnectivity+GinAR(Yu et al., 2024), and MaxValue+GinAR(Yu et al., 2024), where Grid segments the region (as shown in Fig. 3) into a set of rectangular grids and chooses the one with the highest connectivity within each grid for sensor deployment, Metaheuristics optimize sensor placement using a genetic algorithm (GA) (Holland, 1992), MaxConnectivity chooses the locations with the highest connections (node degrees), and MaxValue chooses the locations with the highest values (e.g., traffic rates) on average. Finally, we present a hybrid model, VIP+Pre, where half of the sensor locations are selected by MaxValue (which could be due to a government policy) and then the other half are selected by our proposed VIP. It covers the case where some chosen variables are pre-determined and others can be optimally selected.

For fair comparison over all baselines, we use the recommended hyperparameters from their official GitHub repositories. Before feeding a value matrix into the model, we preprocess it using z-score normalization, i.e., by subtracting the mean flow rate and dividing by the standard deviation. For all baselines except the VIP variants and GinAR variants, performance is reported as the average over four random seeds, where deployment locations are randomly selected in each run.

Accuracy Metrics: We evaluate the forecast accuracy of the proposed method and the baseline models using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE), averaged over all variables and all future time steps in the prediction horizon T'.

Given the ground truth $\mathcal{X}_{i,j}$ and the forecast result $\hat{\mathcal{X}}_{i,j}$, where $i \in N$ and $j \in T'$, the evaluation metrics are defined as

$$\text{MAE} = \frac{1}{n|T'|} \sum_{j \in T'} \sum_{i=1}^{n} \left| \mathcal{X}_{i,j} - \hat{\mathcal{X}}_{i,j} \right|,$$

$$\text{MAPE} = \left(\frac{1}{n|T'|} \sum_{j \in T'} \sum_{i=1}^{n} \left| \frac{\mathcal{X}_{i,j} - \hat{\mathcal{X}}_{i,j}}{\mathcal{X}_{i,j}} \right| \right) \times 100\%,$$

and

$$\text{RMSE} = \sqrt{\frac{1}{n|T'|} \sum_{j \in T'} \sum_{i=1}^{n} \left(\mathcal{X}_{i,j} - \hat{\mathcal{X}}_{i,j} \right)^2}.$$

5.2 Forecast Accuracy

Our VIP model includes two variants: VIP without pretraining (VIP wo. pretrain) and VIP with pretraining (VIP w. pretrain). VIP w. pretrain is initialized from a pretrained STMF model described in Section 4.1, where the base STMF is first trained to convergence with the best validation performance. In contrast, VIP wo. pretrain uses the same architecture without pretraining.

Table 1 shows that VIP w. pretrain consistently outperforms all baselines in forecast accuracy. For example, on PEMSBAY, it improves RMSE by 30% over the best baseline. VIP wo. pretrain achieves the second-best results on PEMS04, PEMS08, and PEMSBAY, and still clearly outperforms all baselines, although its gains on METRLA and AQI are less pronounced. It also requires less training time than most baselines, as shown in the next subsection.

The non-deep-learning matrix factorization baseline performs poorly, likely because it cannot capture complex spatio-temporal dependencies. Among the spatio-temporal forecasting baselines, STAEFormer (Liu et al., 2023a) achieves the strongest overall performance, benefiting from its embedding designs such as node, time-of-day, and day-of-week embeddings. TESTAM (Lee & Ko, 2024) performs similarly well due to its informative mixture-of-experts (MOE) architecture (Shazeer et al., 2017). The two-stage spatio-temporal extrapolation models suffer from error accumulation after the first forecasting stage. The imputation-based methods replace missing values with zeros, which may hurt performance by mixing imputed zeros with true

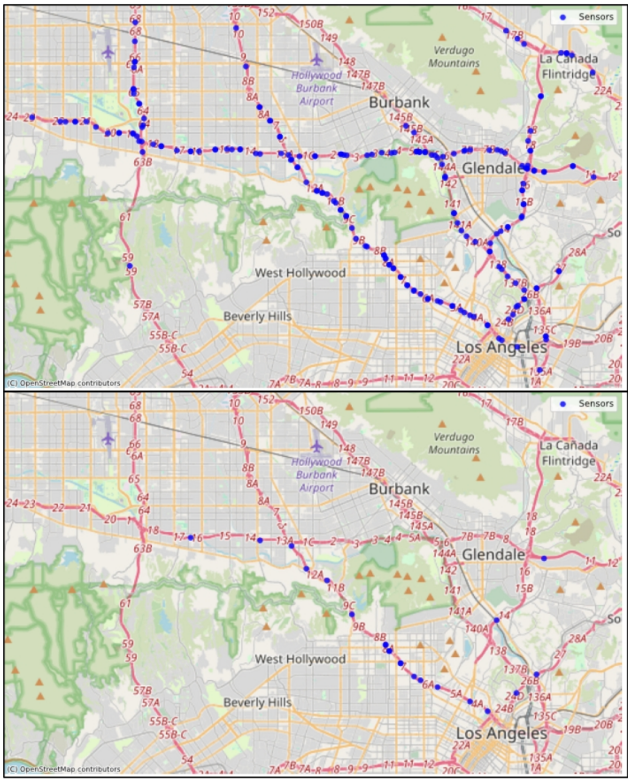

Figure 3: Visualization of sensor locations with METR-LA dataset.

zeros or low values in the input. The two-phase baselines, which first select sensor locations heuristically and then train forecasting models, also underperform our VIP model, which jointly optimizes location selection and forecasting in one iterative process.

5.3 A Case Demonstration of VIP's Location Selection

To provide an intuitive example of STMF with chosen variables, Fig. 3 visualizes VIP's location selection on the METRLA dataset. The top plot shows the full set N of candidate locations, and the bottom plot shows the subset M selected by VIP under a 10% deployment ratio.

The selected locations cover or lie near major junctions, but their density varies across routes. For example, more locations are selected along the US-101 corridor (from 24B to 13A to 24) than along the I-5 highway (from 26B to 5 to 152). This pattern likely reflects the higher topological connectivity and greater traffic variability along the US-101 route connecting central Los Angeles and Burbank, making it more informative under our pruning objective. Such spatially diverse selection helps improve forecasting accuracy even under the constrained deployment budget of 10%, as shown in Table 1.

5.4 Efficiency Study

Fig. 4 compares our VIP models with the baseline models in efficiency-accuracy tradeoff. The batch size is uniformly set to 16. Our models achieve the best RMSE among all models with relatively low computational cost. This dual advantage of high accuracy and efficiency sets our models apart in practical forecast. Note that the training time of VIP w. pretrain includes the training time of its base STMF model. The reason for VIP's efficiency is that we prune locations and parameters with the variable mask and the parameter mask, which reduces both the input size and the network size during training.

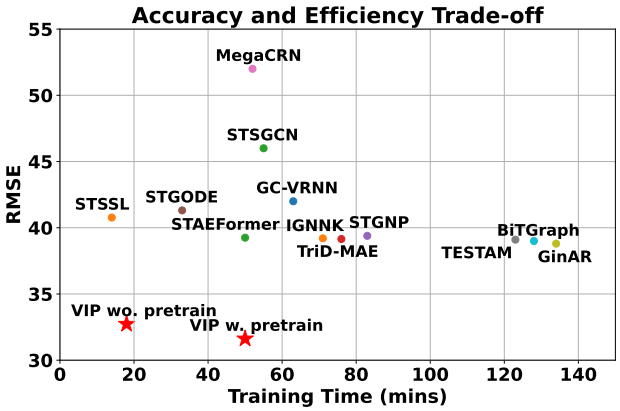

Figure 4: Comparing our VIP models with the baselines in accuracy-efficiency tradeoff, over dataset PEMS08, with a deployment ratio of $m/n = 10\%$.

Table 4: Different ablation methods on PEMS08 dataset over the average horizon when the sensor deployment rate $m/n = 10\%$.

Models	wo. pretrain			w. pretrain		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
VIP	19.87	32.73	13.19	18.59	31.33	12.63
VIP wo. extra	22.10	38.42	14.39	21.19	37.78	13.47
VIP wo. b	20.63	33.47	13.91	19.22	31.79	12.85
VIP wo. p	20.77	33.22	15.90	18.93	31.48	13.88
VIP wo. replay	20.61	33.50	15.61	19.20	32.02	13.55

5.5 Ablation Study

To investigate the effect of different components in VIP, we conduct ablation experiments over PEMS08 by taking one component away each time: **VIP wo. extra**: It does not use the dynamic extrapolation in Eq. equation 15. Instead, we use an MLP to map the representation $H_{N,T}[b]$ to $H_{N,T}$. **VIP wo. b**: It removes random regularization of the learnable mask $\hat{b}$. **VIP wo. p**: It removes random regularization of the learnable mask $\hat{p}$. **VIP wo. replay**: It removes the prioritized variable-parameter replay.

Table 4 compares VIP (wo. or w. pretrain) with all the above variants, each missing a component, over PEMS08 dataset when the deployment ratio is $m/n = 10\%$. The degraded result of VIP wo. extra demonstrates the importance to capture the spatial correlations among the locations with the help of adjacency matrix and location embedding. The degraded result of VIP wo. b (VIP wo. p) demonstrates the importance of randomly-initiated sparse learnable node mask $\hat{b}$ (parameter mask $\hat{p}$), which generates different selections for optimization, instead of stable selections that lead to sub-optimal results. Finally VIP wo. replay also performs worse than VIP because it suffers from the catastrophic forgetting problem, where the model cannot remember the previous knowledge during pruning as new knowledge is learned.

5.6 Diversity Study

In order to justify how L1 norm in Eq. (20) promotes diversity in location selection, we present experiment results in Table 5 that compares different regularization methods, including our choice of VIP with L1 regularization (VIP w. L1) and alternative choices of VIP with L2 regularization (VIP w. L2), VIP with Elastic Net regularization (Zou & Hastie, 2005) (VIP w. ElasticNet), and VIP without any regularization (VIP wo. reg). We focus on two metrics: the forecast accuracy and the Jaccard Distance (Real & Vargas, 1996), which is defined to measure the diversity of a dataset, in our case, the diversity of the location selections

Table 5: Location selection diversity and forecast accuracy under different regularization methods over all batches from the PEMS08 dataset with 10% deployment ratio.

Regularization	**Jaccard Distance**	**VIP w. pretrain**		
		MAE	RMSE	MAPE
VIP w. L1	2.8%	18.59	31.33	12.63
VIP w. L2	2.3%	19.19	32.59	13.31
VIP w. ElasticNet	2.4%	19.07	32.11	13.02
VIP wo. Reg	1.5%	20.25	33.90	14.53

Table 6: Model parameter size (in MB) under varying numbers of variables.

Model	**100**	**500**	**1,000**	**5,000**	**10,000**	**50,000**	**100,000**
GinAR	0.67	2.02	4.61	61.29	222.14	5108.94	20217.44
VIP	0.68	1.09	1.64	7.88	20.18	298.59	1096.59

(i.e., variable masks) across all batches in all iterations. Its formula is

$$\text{Jaccard Distance} = 1 - \frac{2}{K \cdot B(B-1)} \sum_{k=1}^{K} \sum_{1 \le t < t' \le B} \frac{|b_t^k \cap b_{t'}^k|}{|b_t^k \cup b_{t'}^k|}$$

where K is the total number of iterations to reach the 10% deployment ratio, B is the number of batches for each iteration, b_t^k and $b_{t'}^k$ are variable masks for the t-th batch and the t'-th batch in the k-th iteration.

The experiment results show that VIP with L1 regularization generates variable masks with the greatest Jaccard Distance, indicating better location selection diversity, and thus achieves the best accuracy when compared to other regularization methods.

5.7 Scalability Study

We present the results of the scalability studies in Table 6 and Table 7, comparing the memory footprint and inference latency of our VIP models against the most related GinAR model, respectively. All experiments were conducted on an NVIDIA B200 GPU with 180 GB memory, an Intel(R) Xeon(R) Platinum 8570 CPU, and 200 GB RAM. The batch size is set to 1, $l = l' = 12$, and the deployment ratio is 10% across all scales.

The results show that as the total number n of variables increases from 100 to 100,000, VIP exhibits significantly better scalability in both memory efficiency and inference speed. In Table 6, GinAR's model size grows rapidly from 0.67 MB at 100 variables to 20,217.44 MB at 100,000 variables, while VIP's model size increases much more slowly—from 0.68 MB to only 1,096.59 MB under the same conditions. In Table 7, VIP's inference latency is 188.6 ms per forward pass even at 100,000 variables, whereas GinAR's inference latency is 10086.76ms at 5,000 variables and it experiences out-of-memory (OOM) errors beyond 10,000 variables due to excessive memory consumption.

These results clearly demonstrate that VIP scales more favorably than GinAR in both model size and inference time, making it highly suitable for real-time spatio-temporal forecasting in large scale systems.

5.8 Different Experience Replay Methods

To further evaluate the design of our prioritized variable-parameter replay (PVR), we substitute it with two other experience replay designs: random experience replay (RAND-ER) and inverse prioritized variable-parameter replay (IN-PVR). RAND-ER randomly replays the samples from the buffer. IN-PVR gives the high-loss samples a higher probability of being replayed. Specifically, we flip the priority P_S in Eq. equation 17 to $P_S = \mathcal{L}_{main}(\cdot, \cdot)$, and the probability for a sample to be selected is proportional to P_S.

Table 8 compares PVR and RAND-ER and IN-PVR in terms of average MAE, MAPE and RMSE on dataset PEMS08 with a deployment ratio of $m/n = 10\%$. It shows that PVR achieves the best performance because

Table 7: Inference time per forward pass (in milliseconds) under varying numbers of variables.

Model	**100**	**500**	**1,000**	**5,000**	**10,000**	**50,000**	**100,000**
GinAR	75.1	141.4	328.9	10086.76	OOM	OOM	OOM
VIP	3.47	3.53	4.35	4.35	6.08	53.86	188.6

Table 8: Different experience replay methods on PEMS08 dataset with a deployment ratio of $m/n = 10\%$.

Models	**VIP wo. pretrain**			**VIP w. pretrain**		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
PVR	19.87	32.73	13.19	18.59	31.33	12.63
RAND-ER	20.47	33.62	13.53	19.47	31.92	13.08
IN-PVR	21.68	34.74	18.35	20.56	33.81	15.38

the dynamic change of variable/parameter masks causes model instability, the pruning of variables and parameters can result in forgetting the established patterns, and replaying low-loss samples helps mitigate that. That is also the reason that IN-PVR performs worse than RAND-ER because replaying high-loss samples will not do better in restoring model stability during pruning than replaying random samples.

5.9 RMSE under Different Sparsity Ratios

Fig. 5 presents the forecast accuracy results (in terms of RMSE) of VIP w. pretrain under different sparsity ratios, with the pruning variable rate $r_b = 0.1$ and the pruning parameter rate $r_p = 0.05$. The baseline for comparison is GinAR, which is the most related work. Each plot in the figure is for a different dataset. The horizontal dashed lines show the RMSE values achieved by the baseline GinAR under sparsity ratios of 25%, 50%, 75%, and 90%, respectively. The stars on the curve show the RMSE values achieved by VIP w. pretrain under these ratios, respectively. Each vertical line segment from a star shows the RMSE improvement of VIP over the baseline. The results demonstrate that VIP is more accurate than GinAR under different sparsity ratios over different datasets. Note that a higher sparsity ratio means a lower deployment ratio.

5.10 Accuracy Performance comparison under other Node Sparsity

Tables 9 present complementary experiments to Table 1. Note that the results are copied directly from the GinAR (Yu et al., 2024) paper. Our model consistently outperforms the baselines in different node sparsity and datasets.

6 Conclusion

This paper studies a new problem of spatio-temporal multivariate time series forecast with chosen variables (STCV). Our variable-parameter iterative pruning model (VIP) jointly optimizes variable selection and parameter sparsification via a co-optimized training scheme. By integrating a dynamic extrapolation strategy and a prioritized replay mechanism, VIP maintains stable learning under iterative pruning. Experiments across five real-world datasets validate the effectiveness and efficiency of our proposed model.

References

H Zainol Abidin, Norashidah Md Din, NA Mohamed Radzi, and Zairi Ismael Rizman. A review on sensor node placement techniques in wireless sensor networks. *International Journal on Advanced Science, Engineering and Information Technology*, 7(1):190–197, 2017.

Fadi M Al-Turjman, Hossam S Hassanein, and Mohamad Ibnkahla. Quantifying connectivity in wireless sensor networks with grid-based deployments. *Journal of Network and Computer Applications*, 36(1): 368–377, 2013.

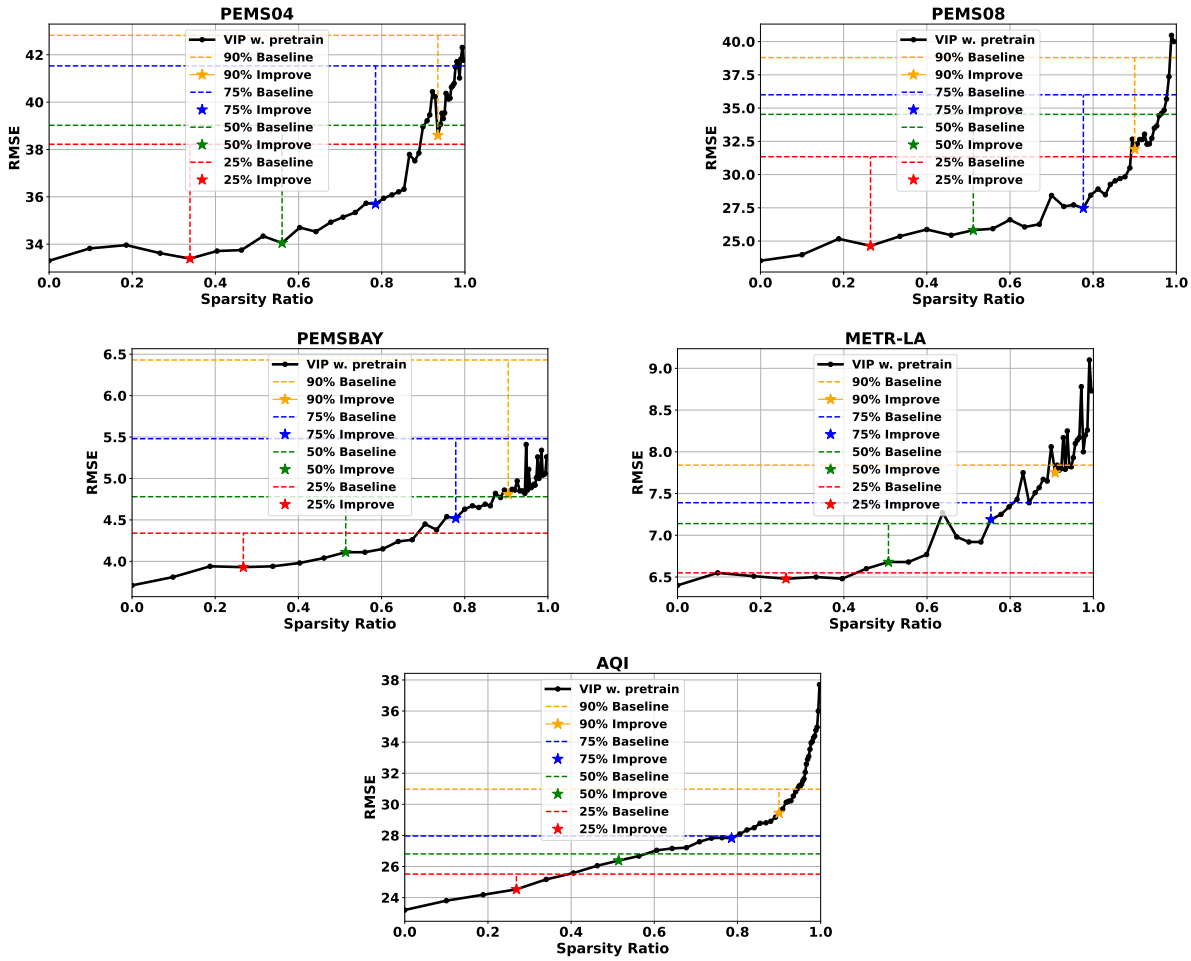

Figure 5: Visualization of node sparsity in different datasets.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11254–11263, 2019.

Gabriel Appleby, Linfeng Liu, and Li-Ping Liu. Kriging convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3187–3194, 2020.

Xiaodan Chen, Xiucheng Li, Bo Liu, and Zhijun Li. Biased temporal convolution graph network for time series forecasting with missing values. In *The Twelfth International Conference on Learning Representations*, 2023a.

Yakun Chen, Xianzhi Wang, and Guandong Xu. Gatgpt: A pre-trained large language model with graph attention network for spatiotemporal imputation. *arXiv preprint arXiv:2311.14332*, 2023b.

Andrea Cini, Ivan Marisca, and Cesare Alippi. Filling the g_ap_s: Multivariate time series imputation by graph neural networks. *arXiv preprint arXiv:2108.00298*, 2021.

Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. Spatial-temporal graph ode networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 364–373, 2021.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Mridul Gupta, Hariprasad Kodamana, and Sayan Ranu. Frigate: Frugal spatio-temporal forecasting on road networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, pp. 649–660, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701030. doi: 10.1145/3580305.3599357. URL https://doi.org/10.1145/3580305.3599357.

Suneet K Gupta, Pratyay Kuila, and Prasanta K Jana. Genetic algorithm for k-connected relay node placement in wireless sensor networks. In *Proceedings of the Second International Conference on Computer and Communication Technologies: IC3T 2015, Volume 1*, pp. 721–729. Springer, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

John H Holland. Genetic algorithms. *Scientific american*, 267(1):66–73, 1992.

Junfeng Hu, Yuxuan Liang, Zhencheng Fan, Hongyang Chen, Yu Zheng, and Roger Zimmermann. Graph neural processes for spatio-temporal extrapolation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 752–763, 2023.

SNS Insider. Traffic sensor market size, share growth dreives report 2032, 2026. URL https://www.snsinsider.com/reports/traffic-sensor-market-1886#:~:text=Deploying%20traffic%20sensor%20systems%20requires,Cloud%20Computing%20and%20Big%20Data. Accessed: 2026-01-28.

Jiahao Ji, Jingyuan Wang, Chao Huang, Junjie Wu, Boren Xu, Zhenhe Wu, Junbo Zhang, and Yu Zheng. Spatio-temporal self-supervised learning for traffic flow prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 4356–4364, 2023.

Jiawei Jiang, Chengkai Han, Wayne Xin Zhao, and Jingyuan Wang. Pdformer: propagation delay-aware dynamic long-range transformer for traffic flow prediction. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023a. ISBN 978-1-57735-880-0. doi: 10.1609/aaai.v37i4.25556. URL https://doi.org/10.1609/aaai.v37i4.25556.

Renhe Jiang, Zhaonan Wang, Jiawei Yong, Puneet Jeph, Quanjun Chen, Yasumasa Kobayashi, Xuan Song, Shintaro Fukushima, and Toyotaro Suzumura. Spatio-temporal meta-graph learning for traffic forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(7):8078–8086, Jun. 2023b. doi: 10.1609/aaai.v37i7.25976. URL https://ojs.aaai.org/index.php/AAAI/article/view/25976.

Renhe Jiang, Zhaonan Wang, Jiawei Yong, Puneet Jeph, Quanjun Chen, Yasumasa Kobayashi, Xuan Song, Shintaro Fukushima, and Toyotaro Suzumura. Spatio-temporal meta-graph learning for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 8078–8086, 2023c.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Daniel Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13, 2000.

Hyunwook Lee and Sungahn Ko. TESTAM: A time-enhanced spatio-temporal attention model with mixture of experts. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=N0nTk5BSvO.

Mengzhang Li and Zhanxing Zhu. Spatial-temporal fusion graph neural networks for traffic flow forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5):4189–4196, May 2021. doi: 10.1609/aaai.v35i5.16542. URL https://ojs.aaai.org/index.php/AAAI/article/view/16542.

Xiao Li, Huan Li, Hua Lu, Christian S Jensen, Varun Pandey, and Volker Markl. Missing value imputation for multi-attribute sensor data streams via message propagation. *Proceedings of the VLDB Endowment*, 17 (3):345–358, 2023.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJiHXGWAZ.

Yujie Li, Zezhi Shao, Yongjun Xu, Qiang Qiu, Zhaogang Cao, and Fei Wang. Dynamic frequency domain graph convolutional network for traffic forecasting. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5245–5249. IEEE, 2024.

Hangchen Liu, Zheng Dong, Renhe Jiang, Jiewen Deng, Jinliang Deng, Quanjun Chen, and Xuan Song. Spatio-temporal adaptive embedding makes vanilla transformer sota for traffic forecasting. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 4125–4129, 2023a.

Zibo Liu, Parshin Shojaee, and Chandan K. Reddy. Graph-based multi-ODE neural networks for spatio-temporal traffic forecasting. *Transactions on Machine Learning Research*, 2023b. ISSN 2835-8856. URL https://openreview.net/forum?id=Oq5XKRVYpQ.

Selectum LLC. Monnit mns2-9-in-vs-tt alta industrial vehicle detect-counter (900 mhz), 2026. URL https://selectumllc.com/i/monnit-mns2-9-in-vs-tt?srsltid=AfmBOoqxe1LLRwFGQBk3QjWUzsZl_BgN4sPHljH8FMG0yQ20rJOLKhXJVh0. Accessed: 2026-01-28.

Seapahn Meguerdichian, Farinaz Koushanfar, Miodrag Potkonjak, and Mani B Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *Proceedings IEEE INFOCOM 2001. Conference on computer communications. Twentieth annual joint conference of the IEEE computer and communications society (Cat. No. 01CH37213)*, volume 3, pp. 1380–1387. IEEE, 2001.

Omead Pooladzandi, David Davini, and Baharan Mirzasoleiman. Adaptive second order coresets for data-efficient machine learning. In *International Conference on Machine Learning*, pp. 17848–17869. PMLR, 2022.

Rahul Priyadarshi, Bharat Gupta, and Amulya Anurag. Deployment techniques in wireless sensor networks: a survey, classification, challenges, and future research issues. *The Journal of Supercomputing*, 76:7333–7373, 2020.

Raimundo Real and Juan M Vargas. The probabilistic basis of jaccard's index of similarity. *Systematic biology*, 45(3):380–385, 1996.

Retail Sensing. Vehicle detection: Ten ways compared, August 2024. URL https://www.retailsensing.com/people-counting/vehicle-detection-count-traffic/. Accessed: 2025-06-24.

Tom Schaul. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Zezhi Shao, Zhao Zhang, Fei Wang, Wei Wei, and Yongjun Xu. Spatial-temporal identity: A simple yet effective baseline for multivariate time series forecasting. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 4454–4458, 2022.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. 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, pp. 914–921, 2020.

X. Su, J. Qi, E. Tanin, Y. Chang, and M. Sarvi. Spatial-temporal forecasting for regions without observations. *Advances in Database Technology - EDBT*, 27(3):488–500, 2024. doi: 10.48786/edbt.2024.42.

Xianfeng Tang, Huaxiu Yao, Yiwei Sun, Charu Aggarwal, Prasenjit Mitra, and Suhang Wang. Joint modeling of local and global temporal dynamics for multivariate time series forecasting with missing values. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5956–5963, 2020.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The eleventh international conference on learning representations*, 2022.

Yuankai Wu, Dingyi Zhuang, Aurelie Labbe, and Lijun Sun. Inductive graph neural networks for spatiotemporal kriging. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4478–4485, 2021a.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 753–763, 2020.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pp. 1907–1913. AAAI Press, 2021b. ISBN 9780999241141.

Xiaobo Xia, Jiale Liu, Jun Yu, Xu Shen, Bo Han, and Tongliang Liu. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *The Eleventh International Conference on Learning Representations*, 2022.

Yi Xu, Armin Bazarjani, Hyung-gun Chi, Chiho Choi, and Yun Fu. Uncovering the missing pattern: Unified framework towards trajectory imputation and prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9632–9643, 2023.

Yu Yang, Hao Kang, and Baharan Mirzasoleiman. Towards sustainable learning: Coresets for data-efficient deep learning. In *International Conference on Machine Learning*, pp. 39314–39330. PMLR, 2023.

Kun Yi, Qi Zhang, Wei Fan, Hui He, Liang Hu, Pengyang Wang, Ning An, Longbing Cao, and Zhendong Niu. Fouriergnn: Rethinking multivariate time series forecasting from a pure graph perspective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

Chengqing Yu, Fei Wang, Zezhi Shao, Tao Sun, Lin Wu, and Yongjun Xu. Dsformer: A double sampling transformer for multivariate time series long-term prediction. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pp. 3062–3072, 2023.

Chengqing Yu, Fei Wang, Zezhi Shao, Tangwen Qian, Zhao Zhang, Wei Wei, and Yongjun Xu. Ginar: An end-to-end multivariate time series forecasting model suitable for variable missing. *arXiv preprint arXiv:2405.11333*, 2024.

Chengqing Yu, Fei Wang, Chuanguang Yang, Zezhi Shao, Tao Sun, Tangwen Qian, Wei Wei, Zhulin An, and Yongjun Xu. Merlin: Multi-view representation learning for robust multivariate time series forecasting with unfixed missing rates. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 3633–3644, 2025.

Xiangyu Yu, Ninghao Liu, Weipeng Huang, Xin Qian, and Tao Zhang. A node deployment algorithm based on van der waals force in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 9 (10):505710, 2013.

Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017. doi: 10.1609/aaai.v31i1.10735. URL `https://ojs.aaai.org/index.php/AAAI/article/view/10735`.

Kai Zhang, Chao Li, and Qinmin Yang. Trid-mae: A generic pre-trained model for multivariate time series with missing values. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 3164–3173, 2023.

Shangtong Zhang and Richard S Sutton. A deeper look at experience replay. *arXiv preprint arXiv:1712.01275*, 2017.

Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, Jianzhong Qi, Chaochao Chen, and Longbiao Chen. Increase: Inductive graph representation learning for spatio-temporal kriging. In *Proceedings of the ACM Web Conference 2023*, pp. 673–683, 2023.

Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Universal time series analysis by pretrained lm and specially designed adaptors. *arXiv preprint arXiv:2311.14782*, 2023.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320, 2005.

Table 9: Performance Comparison under Different Pruning Rates

Sets	Methods	Node Sparsity 25%			Node Sparsity 50%			Node Sparsity 75%		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
04	STID	41.96	71.39	50.88	63.73	95.09	93.63	82.47	113.02	124.11
	DSformer	32.86	50.15	20.88	33.31	51.51	21.35	37.28	54.91	23.25
	MegaCRN	28.26	42.22	20.18	31.48	47.07	21.29	33.58	47.95	22.03
	DCRNN	25.17	40.03	17.77	26.56	41.64	18.21	28.54	43.71	19.18
	DFDGCN	24.43	39.48	17.40	26.09	41.18	18.49	28.29	42.81	19.91
	MTGNN	24.84	39.67	18.71	26.95	41.91	19.18	28.04	44.36	20.90
	FourierGNN	25.58	40.92	19.61	27.31	42.35	20.55	29.87	45.17	22.20
	LGnet	26.53	42.64	18.42	30.81	46.39	21.30	33.94	52.05	24.83
	GC-VRNN	23.57	39.75	16.82	26.43	41.34	17.83	27.72	43.82	18.67
	TriD-MAE	24.15	39.83	16.98	25.89	40.65	17.52	26.95	41.90	18.04
	BiTGraph	23.01	38.94	16.75	24.15	40.03	17.34	26.33	41.69	17.92
	GinAR	22.52	38.22	16.45	23.78	39.02	17.04	25.98	41.53	17.58
	NIP	**21.00**	**33.82**	**14.05**	**20.59**	**34.01**	**14.59**	**21.61**	**35.82**	**14.32**
08	STID	32.88	58.81	31.71	49.71	79.41	51.74	69.06	101.63	74.47
	DSformer	27.74	38.38	19.24	30.47	42.49	23.79	35.21	51.57	25.18
	MegaCRN	26.04	39.29	17.42	30.68	43.43	21.30	32.23	48.37	21.34
	DCRNN	24.64	36.58	15.96	27.96	41.79	18.73	29.62	44.82	19.79
	DFDGCN	24.26	36.29	15.56	25.39	39.05	19.42	28.30	42.67	20.74
	MTGNN	24.08	36.65	15.17	25.78	37.64	17.41	27.45	40.51	18.52
	FourierGNN	23.77	35.54	15.35	25.53	37.42	16.86	26.89	39.44	17.95
	LGnet	26.51	37.54	22.18	32.04	47.23	21.91	33.65	50.38	21.43
	GC-VRNN	23.25	35.17	14.69	24.27	36.40	15.85	26.32	39.67	16.06
	TriD-MAE	21.53	33.15	14.25	23.18	35.95	15.32	24.89	37.64	15.58
	BiTGraph	20.65	31.89	14.05	22.44	35.06	14.62	23.38	36.98	15.04
	GinAR	20.41	31.34	13.76	22.01	34.53	14.21	23.10	36.04	14.77
	NIP	**15.79**	**24.94**	**11.07**	**16.04**	**25.22**	**11.26**	**16.76**	**27.48**	**11.43**
BAY	STID	3.01	7.13	7.78	3.39	7.86	8.21	3.51	8.26	9.24
	DSformer	2.91	6.32	7.15	3.08	6.46	7.73	3.45	8.15	9.06
	MegaCRN	2.85	5.93	7.03	3.02	7.36	7.75	3.35	7.75	8.77
	DCRNN	2.63	5.54	6.24	2.87	6.22	6.75	3.09	7.14	7.82
	DFDGCN	2.58	5.39	6.17	2.74	6.17	7.03	3.07	6.96	7.59
	MTGNN	2.65	5.78	5.73	2.83	6.19	6.55	3.02	7.06	7.75
	FourierGNN	2.40	5.21	5.46	2.71	5.93	5.96	2.98	6.85	7.58
	LGnet	2.88	6.02	7.19	3.14	6.74	8.15	3.43	8.08	9.18
	GC-VRNN	2.39	4.93	5.37	2.64	5.34	5.86	2.87	6.08	6.94
	TriD-MAE	2.46	5.17	5.48	2.69	5.53	6.12	2.79	5.97	6.85
	BiTGraph	2.17	4.52	5.16	2.44	5.06	6.07	2.61	5.79	6.68
	GinAR	2.10	4.34	4.90	2.35	4.78	5.88	2.54	5.48	6.17
	NIP	**1.95**	**3.95**	**4.56**	**2.05**	**4.07**	**4.84**	**2.24**	**4.51**	**5.44**
LA	STID	5.05	10.31	17.85	5.35	10.73	18.88	5.84	11.35	20.57
	DSformer	4.02	7.69	11.05	4.38	8.27	12.78	4.77	9.21	14.59
	MegaCRN	3.81	7.43	10.47	3.94	7.87	11.02	4.24	8.28	12.13
	DCRNN	3.78	7.41	10.72	3.98	7.91	11.61	4.15	8.16	11.93
	DFDGCN	3.72	7.42	10.42	3.89	7.68	11.45	4.14	8.11	11.75
	MTGNN	3.69	7.28	10.48	3.77	7.43	11.22	4.12	8.05	11.58
	FourierGNN	3.71	7.40	10.87	3.82	7.84	11.75	4.21	8.25	12.25
	LGnet	3.95	7.52	10.97	4.17	8.03	11.83	4.42	8.52	13.09
	GC-VRNN	3.68	7.04	10.51	3.87	7.73	10.98	4.17	8.19	11.79
	TriD-MAE	3.64	7.15	10.37	3.79	7.58	11.07	3.90	7.92	11.13
	BiTGraph	3.61	6.74	10.25	3.69	7.32	10.79	3.74	7.63	11.04
	GinAR	3.56	6.55	10.12	3.61	7.14	10.42	3.70	7.39	10.71
	NIP	**3.28**	**6.50**	**9.27**	**3.31**	**6.58**	**9.56**	**3.47**	**7.20**	**9.98**
AQI	STID	18.54	31.97	45.19	20.32	34.28	48.48	22.79	36.36	55.39
	DSformer	17.73	28.22	44.81	19.06	30.35	48.09	20.63	33.22	52.05
	MegaCRN	15.32	28.41	33.35	16.51	29.52	35.86	19.66	33.09	48.96
	DCRNN	15.14	28.48	32.18	16.82	30.83	35.24	17.82	31.99	37.22
	DFDGCN	14.62	26.33	29.73	15.76	29.30	32.85	17.85	30.51	38.68
	MTGNN	14.89	27.13	33.29	16.37	30.02	37.82	17.94	31.97	39.29
	FourierGNN	14.65	27.17	32.82	16.13	30.77	38.37	18.01	31.86	39.90
	LGnet	16.02	27.76	38.92	18.39	31.03	44.95	20.36	34.09	49.63
	GC-VRNN	14.66	26.88	31.88	15.70	28.67	34.21	16.99	30.57	37.66
	TriD-MAE	14.51	26.18	29.09	15.74	28.96	32.94	16.79	29.84	35.76
	BiTGraph	13.85	25.79	28.94	14.52	27.45	31.36	15.62	29.01	33.58
	GinAR	13.72	25.51	28.27	14.33	26.81	29.56	15.39	27.96	31.86
	NIP	**13.41**	**24.48**	**27.51**	**14.10**	**26.09**	**29.03**	**15.21**	**27.83**	**31.80**