Ada-MSHyper: Adaptive Multi-Scale Hypergraph Transformer for Time Series Forecasting

Zongjiang Shang, Ling Chen,* Binqing Wu, Dongliang Cui

State Key Laboratory of Blockchain and Data Security
College of Computer Science and Technology
Zhejiang University
{zongjiangshang, lingchen, binqingwu, runnercdl}@cs.zju.edu.cn

Abstract

Although transformer-based methods have achieved great success in multi-scale temporal pattern interaction modeling, two key challenges limit their further development: (1) Individual time points contain less semantic information, and leveraging attention to model pair-wise interactions may cause the information utilization bottleneck. (2) Multiple inherent temporal variations (e.g., rising, falling, and fluctuating) entangled in temporal patterns. To this end, we propose **Ada**ptive **M**ulti-**S**cale **Hyper**graph Transformer (Ada-MSHyper) for time series forecasting. Specifically, an adaptive hypergraph learning module is designed to provide foundations for modeling group-wise interactions, then a multi-scale interaction module is introduced to promote more comprehensive pattern interactions at different scales. In addition, a node and hyperedge constraint mechanism is introduced to cluster nodes with similar semantic information and differentiate the temporal variations within each scales. Extensive experiments on 11 real-world datasets demonstrate that Ada-MSHyper achieves state-of-the-art performance, reducing prediction errors by an average of 4.56%, 10.38%, and 4.97% in MSE for long-range, short-range, and ultra-long-range time series forecasting, respectively. Code is available at https://github.com/shangzongjiang/Ada-MSHyper.

1 Introduction

Time series forecasting has demonstrated its wide applications across many fields [30, 37], e.g., energy consumption planning, traffic and economics prediction, and disease propagation forecasting. In these real-world applications, the observed time series often demonstrate complex and diverse temporal patterns at different scales[6, 9, 28]. For example, due to periodic human activities, traffic occupation and electricity consumption show clear daily patterns (e.g., afternoon or evening), weekly patterns (e.g., weekday or weekend), and even monthly patterns (e.g., summer or winter).

Recently, deep models have achieved great success in time series forecasting. To tackle intricate temporal patterns and their interactions at different scales, numerous foundational backbones have emerged, including recurrent neural networks (RNNs) [7, 9, 10], convolutional neural networks (CNNs) [1, 25], graph neural networks (GNNs) [4, 8], and transformers [20, 39]. Particularly, due to the capabilities of depicting pair-wise interactions and extracting multi-scale representations in sequences, transformers are widely used in time series forecasting. However, some recent studies show that even simple multi-scale MLP [11, 35] or naïve series decomposition methods [3, 15] can outperform transformer-based methods on various benchmarks. We argue the challenges that limit the effectiveness of transformers in time series forecasting are as follows.

*Corresponding author: Ling Chen.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-1050

The first one is *semantic information sparsity*. Different from natural language processing (NLP) and computer vision (CV), individual time point in time series contains less semantic information [5, 29]. Compared to pair-wise interactions, group-wise interactions among time points with similar semantic information (e.g., neighboring time points or distant but strongly correlated time points) are more emphasized in time series forecasting. To address the problem of semantic information sparsity, some recent works employ patch-based approaches [12, 23] and hypergraph structures [26] to enhance locality and capture group-wise interactions. However, simple partitioning of patches and predefined hypergraph structures may introduce a large amount of noise and be hard to discover implicit interactions.

The second one is *temporal variations entanglement*. Due to the complexity and non-stationary of real-world time series, the temporal patterns of observed time series often contain a large number of inherent variations (e.g., rising, falling, and fluctuating), which may mix and overlap with each other. Especially when there are distinct temporal patterns at different scales, multiple temporal variations are deeply entangled, bringing extreme challenges for time series forecasting. To tackle the problem of temporal variations entanglement, recent studies employ series decomposition [30, 39] and multi-periodicity analysis [27, 29] to differentiate temporal variations at different scales. However, existing methods lack the ability to differentiate temporal variations within each scale, making temporal variations within each scale overlap and become entangled with redundant information.

Motivated by the above, we propose Ada-MSHyper, an **Ada**ptive **M**ulti-**S**cale **Hyper**graph Transformer for time series forecasting. Specifically, Ada-MSHyper map the input sequence into multi-scale feature representations, then by treating the multi-scale feature representations as nodes, an adaptive multi-scale hypergraph structure is introduced to discover the abundant and implicit group-wise node interactions at different scales. To the best of our knowledge, Ada-MSHyper is the first work that incorporates adaptive hypergraph modeling into time series forecasting. The main contributions are summarized as follows:

- We design an adaptive hypergraph learning (AHL) module to model the abundant and implicit group-wise node interactions at different scales and a multi-scale interaction module to perform hypergraph convolution attention, which empower transformers with the ability to model group-wise pattern interactions at different scales.
- We introduce a node and hyperedge constraint (NHC) mechanism during hypergraph learning phase, which utilizes semantic similarity to cluster nodes with similar semantic information and leverages distance similarity to differentiate the temporal variations within each scales.
- We conduct extensive experiments on 11 real-world datasets. The experimental results demonstrate that Ada-MSHyper achieves state-of-the-art (SOTA) performance, reducing error by an average of 4.56%, 10.38%, and 4.97% in MSE for long-range, short-range, and ultra-long-range time series forecasting, respectively, compared to the best baseline.

2 Related Work

Deep Models for Time Series Forecasting. Deep models have shown promising results in time series forecasting. To model temporal patterns at different scales and their interactions, a large number of specially designed backbones have emerged. TAMS-RNNs [10] captures periodic temporal dependencies through multi-scale recurrent structures with different update frequencies. TimesNet [29] extends the 1D time series into the 2D space, and models multi-scale temporal pattern interactions through 2D convolution inception blocks. Benefiting from the attention mechanism, transformers have gone beyond contemporaneous RNN- and CNN-based methods and achieved promising results in time series forecasting. FEDformer [39] combines mixture of expert and frequency attention to capture multi-scale temporal dependencies. Pyraformer [20] extends the input sequence into multi-scale representations and models the interactions between nodes at different scales through pyramidal attention. Nevertheless, with the rapid emergence of linear forecasters [22, 35, 11], the effectiveness of transformers in this direction is being questioned.

Recently, some methods have attempted to fully utilize transformers and paid attention to the inherent properties of time series. Some of these methods are dedicated to addressing the problem of semantic information sparsity in time series forecasting. PatchTST [23] segments the input sequence into subseries-level patches to enhance locality and capture group-wise interactions. MSHyper [26]

models group-wise interactions through multi-scale hypergraph structures and introduces k-hop connections to aggregate information from different range of neighbors. However, constrained by the fixed windows and predefined rules, these methods cannot discover implicit interactions. Others emphasize on addressing the problem of temporal variations entanglement in time series forecasting. FilM [38] differentiates temporal variations at different scales by decomposing the input series into different period lengths. iTransformer [21] combines inverted structures with transformer to learn entangled global temporal variations. However, these methods cannot differentiate temporal variations within each scale, making temporal variations within each scale overlap and become entangled with redundant information.

Hypergraph Neural Networks. As a generalized form of GNNs, hypergraph neural networks (HGNNs) have been applied in different fields, e.g., video object segmentation [17], stock selection [24], multi-agent trajectory prediction [31], and time series forecasting [26]. HyperGCN [32] is the first work that incorporates convolution operation into hypergraphs, which demonstrates the superiority of HGNNs over ordinary GNNs in capture group-wise interactions. Recent studies [2, 33] show that HGNNs are promising to model group-wise pattern interactions. LBSN2Vec++ [34] uses hypergraphs for location-based social networks, which leverages heterogeneous hypergraph embeddings to capture mobility and social relationship pattern interactions. GroupNet [31] utilizes multi-scale hypergraph for trajectory prediction, which combines relational reasoning with hypergraph structures to capture group-wise pattern interactions among multiple agents.

Considering the capability of HGNNs in modeling group-wise interactions, in this work, an adaptive multi-scale hypergraph transformer framework is proposed to model the group-wise pattern interactions at different scales. Specifically, an AHL model is designed to model the abundant and implicit group-wise node interactions. In addition, a NHC mechanism is introduced to cluster nodes with similar semantic information and differentiate temporal variations within each scale, respectively.

3 Preliminaries

Hypergraph. A hypergraph is defined as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{E} = \{e_1, \ldots, e_m, \ldots, e_M\}$ is the hyperedge set and $\mathcal{V} = \{v_1, \ldots, v_n, \ldots, v_N\}$ is the node set. Each hyperedge represents group-wise interactions by connecting a set of nodes $\{v_1, v_2, \ldots, v_n\} \subseteq \mathcal{V}$. The topology of hypergraph can be represented as an incidence matrix $\mathbf{H} \in \mathbb{R}^{N \times M}$, with entries $\mathbf{H}_{nm}$ defined as follows:

$$\mathbf{H}_{nm} = \begin{cases} 1, & v_n \in e_m \\ 0, & v_n \notin e_m \end{cases} \tag{1}$$

The degree of the nth node is defined as $d(v_n) = \sum_{m=1}^{M} \mathbf{H}_{nm}$ and the degree of the mth hyperedge is defined as $d(v_m) = \sum_{n=1}^{N} \mathbf{H}_{nm}$. Further, the node degrees and hyperedge degrees are sorted in diagonal matirces $\mathbf{D}_{\mathrm{v}} \in \mathbb{R}^{N \times N}$ and $\mathbf{D}_{\mathrm{e}} \in \mathbb{R}^{M \times M}$, respectively.

Problem Formulation. Given the input sequence $\mathbf{X}^{\mathrm{I}}_{1:T} = \left\{ \boldsymbol{x}_t \mid \boldsymbol{x}_t \in \mathbb{R}^D, t \in [1, T] \right\}$, where $\boldsymbol{x}_t$ represents the values at time step t, T is the input length, and D is the feature dimension. The task of time series forecasting is to predict the future H steps, which can be formulated as follows:

$$\widehat{\mathbf{X}}^{\mathrm{O}}_{T+1:T+H} = \mathcal{F}\left(\mathbf{X}^{\mathrm{I}}_{1:T}; \theta\right) \in \mathbb{R}^{H \times D}, \tag{2}$$

where $\widehat{\mathbf{X}}^{\mathrm{O}}_{T+1:T+H}$ denotes the forecasting results, $\mathcal{F}$ denotes the mapping function, and θ denotes the learnable parameters of $\mathcal{F}$. The description of the key notations are given in Appendix A.

4 Ada-MSHyper

As previously mentioned, the core of Ada-MSHyper is to promote more comprehensive pattern interactions at different scales. To accomplish this goal, we first map the input sequence into subsequences at different scales through the multi-scale feature extraction (MFE) module. Then, by treating multi-scale feature representations as nodes, the AHL module is introduced to model the abundant and implicit group-wise node interactions at different scales. Finally, the multi-scale interaction module is introduced to model group-wise pattern interactions at different scales. Notably, during the hypergraph learning phase, an NHC mechanism is introduced to cluster nodes with similar

 https://doi.org/10.52202/079017-1050

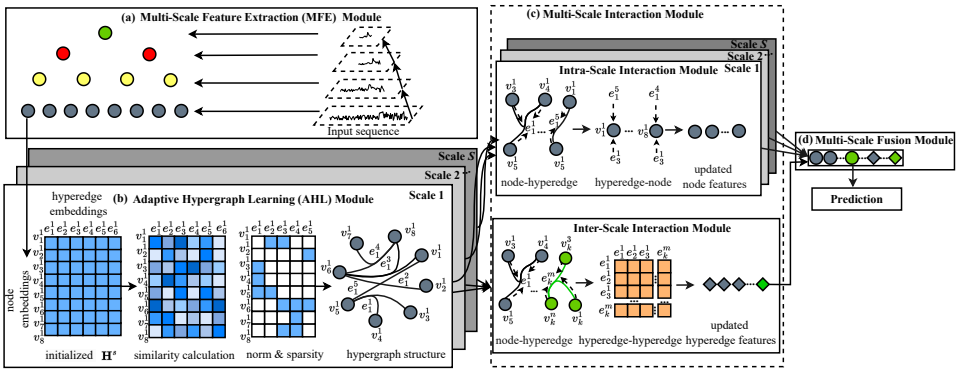


Figure 1: The framework of Ada-MSHyper.

semantic information and differentiate temporal variations within each scale. The overall framework of Ada-MSHyper is shown in Figure 1.

4.1 Multi-Scale Feature Extraction (MFE) Module

The MFE module is designed to get the feature representations at different scales. As shown in Figure 1(a), suppose $\mathbf{X}^s = \{\boldsymbol{x}_t^s | \boldsymbol{x}_t^s \in \mathbb{R}^D, t \in [1, N^s]\}$ denotes the sub-sequence at scale s, where $s = 1, ..., S$ denotes the scale index and S is the total number of scales. $N^s = \left\lfloor \frac{N^{s-1}}{l^{s-1}} \right\rfloor$ is the number of nodes at scale s and l^{s-1} denotes the size of the aggregation window at scale $s-1$. $\mathbf{X}^1 = \mathbf{X}_{1:T}^{\mathrm{I}}$ is the raw input sequence and the aggregation process can be formulated as follows:

$$\mathbf{X}^s = Agg(\mathbf{X}^{s-1}; \theta^{s-1}) \in \mathbb{R}^{N^s \times D}, s \geq 2, \tag{3}$$

where Agg is the aggregation function, e.g., 1D convolution or average pooling, and θ^{s-1} denotes the learnable parameters of the aggregation function at scale $s-1$.

4.2 Adaptive Hypergraph Learning (AHL) Module

The AHL module automatically generates incidence matrices to model implicit group-wise node interactions at different scales. As shown in Figure 1(b), we first initialize two kinds of parameters, i.e., node embeddings $\boldsymbol{E}_{\mathrm{node}}^s \in \mathbb{R}^{N^s \times D}$ and hyperedge embeddings $\boldsymbol{E}_{\mathrm{hyper}}^s \in \mathbb{R}^{M^s \times D}$ at scale s, where M^s is hyperparameters, representing the number of hyperedges at scale s. Then, we can obtain the scale-specific incidence matrix $\mathbf{H}^s$ by similarity calculation, which can be formulated as follows:

$$\mathbf{H}^s = SoftMax(ReLU(\boldsymbol{E}_{\mathrm{node}}^s (\boldsymbol{E}_{\mathrm{hyper}}^s)^T)), \tag{4}$$

where the $ReLU$ activation function is used to eliminate weak connections and the $SoftMax$ function is applied to normalize the value of $\mathbf{H}^s$. In order to reduce subsequent computational costs and noise interference, the following strategy is designed to sparsify the incidence matrix:

$$\mathbf{H}_{nm}^s = \begin{cases} \mathbf{H}_{nm}^s, & \mathbf{H}_{nm}^s \in TopK(\mathbf{H}_{n*}^s, \eta) \\ 0, & \mathbf{H}_{nm}^s \notin TopK(\mathbf{H}_{n*}^s, \eta) \end{cases} \tag{5}$$

where η is the threshold of $TopK$ function and denotes the max number of neighboring hyperedges connected to a node. The final values of $\mathbf{H}_{nm}^s$ can be obtained as follows:

$$\mathbf{H}_{nm}^s = \begin{cases} 1, & \mathbf{H}_{nm}^s > \beta \\ 0, & \mathbf{H}_{nm}^s < \beta \end{cases} \tag{6}$$

where β denotes the threshold, and the final scale-specific incidence matrices can be represented as $\{\mathbf{H}^1, \cdots, \mathbf{H}^s, \cdots, \mathbf{H}^S\}$. Compared to previous methods, our adaptive hypergraph learning is novel from two aspects. Firstly, our methods can capture group-wise interactions at different scales, while most previous methods [5, 21] can only model pair-wise interactions at a single scale. Secondly, our methods can model abundant and implicit interactions, while many previous methods [23, 26] depend on fixed windows and predefined rules.

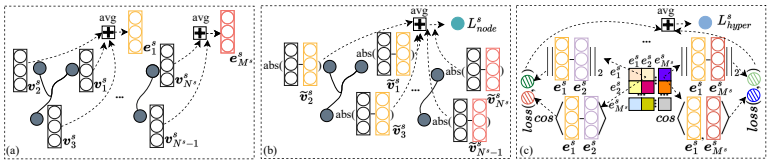


Figure 2: The node and hyperedge constraint mechanism.

4.3 Node and Hyperedge Constraint (NHC) Mechanism

Although the AHL module can help discover implicit group-wise node interactions at different scales, we argue that the pure data-driven approach faces two limitations, i.e., unable to efficiently cluster nodes with similar semantic information and differentiate temporal variations within each scale. To tackle the above dilemmas, we introduce the NHC mechanism during hypergraph learning phase.

Given the multi-scale feature representations $\{\mathbf{X}^1, \cdots, \mathbf{X}^s, \cdots, \mathbf{X}^S\}$ generated from the MFE module, and the scale-specific incidence matrices $\{\mathbf{H}^1, \cdots, \mathbf{H}^s, \cdots, \mathbf{H}^S\}$ generated from the AHL module, we first get the initialized node feature representations $\boldsymbol{\mathcal{V}}^s = f(\mathbf{X}^s)$ at scale s, where f can be implemented by the multi-layer perceptron (MLP). As shown in Figure 2(a), the initialized hyperedge feature representations can be obtained by the aggregation operation based on $\mathbf{H}^s$. Specifically, for the ith hyperedge e_i^s at scale s, its feature representations $\boldsymbol{e}_i^s$ can be formulated as follows:

$$\boldsymbol{e}_i^s = avg(\sum\nolimits_{v_j^s \in \mathcal{N}(e_i^s)} \boldsymbol{v}_j^s) \in \mathbb{R}^D, \tag{7}$$

where avg is the average operation, $\mathcal{N}(e_i)$ represents the neighboring nodes connected by e_i^s at scale s, and $\boldsymbol{v}_j^s \in \boldsymbol{\mathcal{V}}^s$ is the jth node feature representations at scale s. The initialized hyperedge feature representations at different scales can be represented as $\{\boldsymbol{\mathcal{E}}^1, \cdots, \boldsymbol{\mathcal{E}}^s, \cdots, \boldsymbol{\mathcal{E}}^S\}$. Then, based on semantic similarity and distance similarity, we introduce node constraint to cluster nodes with similar semantic information and leverage hyperedge constraint to differentiate the temporal variations of temporal patterns.

Node Constraint. In the data-driven hypergraph, we observe that some nodes connected by the same hyperedge contain distinct semantic information. To cluster nodes with similar semantic information and reduce irrelevant noise interference, we introduce node constraint based on the semantic similarity between nodes and their corresponding hyperedges. As shown in Figure 2(b), for the jth node at scale s, we first obtain its semantic similarity difference $\widetilde{\boldsymbol{v}_j^s}$ with its corresponding hyperedges:

$$\widetilde{\boldsymbol{v}_j^s} = \{abs(\boldsymbol{v}_j^s - \boldsymbol{e}_i^s) | v_j^s \in \mathcal{N}(e_i^s)\}, \tag{8}$$

where abs refers to the operation of calculating the absolute value. The node loss L_{node}^s at scale s based on node constraint can be formulated as follows:

$$L_{node}^s = \frac{1}{N^s} \sum_{i=1}^{N^s} \widetilde{\boldsymbol{v}_j^s}, \tag{9}$$

where N^s is the number of nodes at scale s. Empowered by the node constraint, our method can enjoy more advantageous group-wise semantic information than pure data-driven hypergraph. Further experimental results and visualization analysis in Section 5.3 and Appendix H demonstrate the effectiveness of the node constraint in clustering nodes with similar semantic information.

Hyperedge Constraint. Since time series is a collection of data points arranged in chronological order, some recent works [23, 26] show that connecting multiple nodes sequentially through patches or hyperedges can represent specific temporal variations. Therefore, to deal with the problem of temporal variations entanglement, we introduce hyperedge constraint based on distance similarity. As shown in Figure 2(c), we first compute the cosine similarity to reflect the correlation of any two hyperedge representations at scale s, which can be formulated as follows:

$$\alpha_{i,j} = \frac{\boldsymbol{e}_i^s (\boldsymbol{e}_j^s)^T}{\|\boldsymbol{e}_i^s\|_2 \|\boldsymbol{e}_j^s\|_2}, \tag{10}$$

https://doi.org/10.52202/079017-1050

where $\alpha_{i,j}$ represents the correlation weight. $\boldsymbol{e}_i^s$ and $\boldsymbol{e}_j^s$ are the ith and jth hyperedge representation at scale s, respectively. Then, we use Euclidean distance $D_{i,j}$ to measure the differentiation magnitude between any two hyperedge representations, which can be formulated as follows:

$$D_{i,j} = \left\| \boldsymbol{e}_i^s - \boldsymbol{e}_j^s \right\|_2 = \sqrt{\sum_{d=1}^{D} ((\boldsymbol{e}_i^s)^d - (\boldsymbol{e}_j^s)^d)^2}, \tag{11}$$

The hyperedge loss L_{hyper}^s at scale s based on the correlation weight and Euclidean distance can be formulated as follows:

$$L_{hyper}^s = \frac{1}{(M^s)^2} \sum_{i=1}^{M^s} \sum_{j=1}^{M^s} \left(\alpha_{i,j} D_{i,j} + (1 - \alpha_{i,j}) max(\gamma - D_{i,j}, 0) \right), \tag{12}$$

where $\gamma > 0$ denotes the threshold. Notably, when $\alpha_{i,j} = 1$, indicating that e_i^s and e_k^s are deemed similar, the hyperedge loss turns to $L_{hyper} = \frac{1}{(M^s)^2} \sum_{i=1}^{M^s} \sum_{j=1}^{M^s} \alpha_{i,j} D_{i,j}$, where the loss will increase if $D_{i,j}$ becomes large. Conversely, when $\alpha_{i,j} = 0$, meaning e_i and e_k are regarded as dissimilar, the hyperedge loss turns to $L_{hyper} = \frac{1}{(M^s)^2} \sum_{i=1}^{M^s} \sum_{j=1}^{M^s} (1 - \alpha_{i,j}) max(s - D_{i,j}, 0)$, where the loss will increase if $D_{i,j}$ falls below the threshold and turns smaller. Other cases lie between the above circumstances. We further provide the visualization results in Section 5.3 and appendix H to verify that our constraint loss can differentiate temporary variations of temporary patterns within each scale and promote forecasting performance. The final constraint loss L_{const} based on node constraint and hyperedge constraint can be formulated as follows:

$$L_{const} = \lambda \sum_{s=1}^{S} L_{node}^s + (1 - \lambda) \sum_{s=1}^{S} L_{hyper}^s, \tag{13}$$

wherer λ denotes the hyperparameter controlling the balance between node loss and hyperedge loss.

4.4 Multi-Scale Interaction Module

To promote more comprehensive pattern interactions at different scales, a direct way is to mix multi-scale node feature representations at different scales. However, we argue that intra-scale interactions and inter-scale interactions reflect different aspects of pattern interactions, where intra-scale interactions mainly depict detailed interactions between nodes with similar semantic information and inter-scale interactions highlight macroscopic variations interactions[9, 27]. Therefore, instead of directly mixing multi-scale pattern information as a whole, we introduce the multi-scale interaction module to perform inter-scale interactions and intra-scale interactions.

Intra-Scale Interaction Module. Due to the semantic information sparsity of time series, traditional pair-wise attention may may cause the information utilization bottleneck [5]. In contrast, some recent studies [23, 26] show that group-wise interactions can provide more informative insights in time series forecasting. To capture group-wise interactions among nodes with similar semantic information within each scale, we introduce hypergraph convolution attention within the intra-scale interaction module. Specifically, given $\mathbf{H}^s$, we first use attention mechanism to capture the interaction strength of each node $v_i^s \in \boldsymbol{\mathcal{V}}^s$ and its related hyperedges at scale s, which can be formulated as follows:

$$\boldsymbol{\mathcal{H}}_{ij}^s = \frac{\exp(\sigma(f_t[\boldsymbol{v}_i^s, \boldsymbol{e}_j^s]))}{\sum_{e_k^s \in \mathcal{N}(v_i^s)} exp(\sigma(f_t[\boldsymbol{v}_i^s, \boldsymbol{e}_k^s])}, \tag{14}$$

where $[.,.]$ denotes the concatenation operation of the ith node and its related hyperedges. f_t is a trainable MLP, and $\mathcal{N}(v_i^s)$ is the neighboring hyperedges connected to v_i^s, which can be accessed using $\mathbf{H}^s$. Then, considering the symmetric normalized hypergraph Laplacian convolution $\Delta = \mathbf{D}_v^{-1/2} \mathbf{H} \mathbf{W} \mathbf{D}_e^{-1} \mathbf{H}^T \mathbf{D}_v^{-1/2}$ used in HGNN [13], the multi-head hypergraph convolution attention can be formulated as follows:

$$\widetilde{\boldsymbol{\mathcal{V}}}^s = \bigoplus_{j=1}^{\mathcal{J}} (\sigma(\mathbf{D}_{v^s}^{-1/2} \boldsymbol{\mathcal{H}}_j^s \mathbf{D}_{e^s}^{-1} (\boldsymbol{\mathcal{H}}_j^s)^{\mathrm{T}} \mathbf{D}_{v^s}^{-1/2} \boldsymbol{\mathcal{V}}^s \mathbf{P}_j^s)) \in \mathbb{R}^{N^s \times D}, \tag{15}$$

where $\widetilde{\boldsymbol{\mathcal{V}}}^s$ is the updated node feature representations at scale s, $\bigoplus$ is the aggregation function used for combing the outputs of multi-head, e.g., concatenation or average pooling. σ is the activation

function, e.g., LeakyReLU and ELU. $\boldsymbol{\mathcal{H}}_j^s$ and $\mathbf{P}_j^s$ are the enriched incidence matrix and the learnable weight matrix of the jth head at scale s, respectively. $\mathcal{J}$ is the number of heads.

Inter-Scale Interaction Module. The inter-scale interaction module is introduced to capture pattern interactions at different scales. To achieve this goal, a direct way is to model group-wise node interactions across all scales. However, detailed group-wise node interactions across all scales can introduce redundant information and increase computation complexity. Therefore, we adopt a hyperedge attention within the inter-scale interaction module to capture macroscopic variations interactions at different scales. Technically, based on the hyperedge representations $\boldsymbol{\mathcal{E}} = \{\boldsymbol{\mathcal{E}}^1, \cdots, \boldsymbol{\mathcal{E}}^s, \cdots, \boldsymbol{\mathcal{E}}^S\}$, we first adopt linear projections to get queries, keys, and values $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{M \times D}$. Then the hyperedge attention can be formulated as follows:

$$\tilde{\mathbf{V}} = softmax(\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{D_K}})\mathbf{V}, \tag{16}$$

where $\tilde{\mathbf{V}}$ is the updated hyperedge feature representations.

4.5 Prediction Module & Loss Function

After obtaining the updated node and hyperedge feature representations, we concatenate them and feed them into a linear layer for prediction. We choose Mean Squared Error (MSE) as our forecasting loss, which can be formulated as follows:

$$L_{mse} = \frac{1}{H} \left\| \widehat{\mathbf{X}}^{\mathrm{O}}_{T+1:T+H} - \mathbf{X}^{\mathrm{O}}_{T+1:T+H} \right\|_2^2, \tag{17}$$

where $\mathbf{X}^{\mathrm{O}}_{T+1:T+H}$ and $\widehat{\mathbf{X}}^{\mathrm{O}}_{T+1:T+H}$ are ground truth and forecasting results, respectively. Notably, during training phase, L_{mse} is used to regulate the overall learning process, while L_{const} is only used to constrain hypergraph learning process.

4.6 Complexity Analysis

For the MFE module, the time complexity is $\mathcal{O}(Nl)$, where N is the number of nodes at the finest scale and N is equal to the input length T. l is the aggregation window size at the finest scale. For the AHL module, the time complexity is $\mathcal{O}(MN + M^2)$, where M is the number of hypergraphs at the finest scale. For the intra-scale interaction module, since $\mathbf{D}_v$ and $\mathbf{D}_e$ are diagonal matrices, the time complexity is $\mathcal{O}(MN)$. For the inter-scale interaction module, the time complexity is M^2. In practical operation, M and l is the hyperparameter and is much smaller than N. As a result, the total time complexity of Ada-MSHyper is bounded by $\mathcal{O}(N)$.

5 Experiment

5.1 Experimental Setup

Datasets. For long-range time series forecasting, we conduct experiments on 7 commonly used benchmarks, including ETT (ETTh1, ETTh2, ETTm1, and ETTm2), Traffic, Electricity, and Weather datasets following [30, 21, 26]. For short-range time series forecasting, we adopt 4 benchmarks from PEMS (PEMS03, PEMS04, PEMS07, and PEMS08) following [21, 27]. For ultra-long-range time series forecasting, we adopt ETT datasets following [18]. Table 1 gives the dataset statistics. In addition, the forecastability is derived from one minus the entropy of the Fourier decomposition of a time series[27, 14]. Higher values mean greater forecastability.

Table 1: Dataset statistics.

Dataset	# Variates	Prediction Length	Frequency	Forecastability	Information
ETT (4 subsets)	7	(96, 192, 336,720)	(15 mins, Hourly)	(0.38-0.55)	Temperature
Weather	21	(96, 192, 336,720)	10 mins	0.75	Weather
Electricity	321	(96, 192, 336,720)	Hourly	0.77	Electricity
Traffic	862	(96, 192, 336,720)	Hourly	0.68	Transportation
PEMS (4 subsets)	(170-883)	(12, 24, 48)	5 mins	(0.43-0.58)	Traffic network

Baselines. We compare Ada-MSHyper with 15 competitive baselines, i.e., iTransformer [21], MSHyper [26], PatchTST [23], TimeMixer [27], MSGNet [4], CrossGNN [16], TimesNet [29], WITRAN [18], SCINet [19], Crossformer [36], FiLM [38], DLinear [35], FEDformer [39], Pyraformer [20], and Autoformer [30].

https://doi.org/10.52202/079017-1050

Experimental Settings. Ada-MSHyper is trained/tested on a single NVIDIA Geforce RTX 3090 GPU. MSE and MAE are used as evaluation metrics and lower values mean better performance. Adam is set as the optimizer with the initial learning rate of 10^{-4}. It is notable that the above mentioned baseline results cannot be used directly due to different input and output lengths. For a fair comparison, we set the commonly used input length $T = 96$ and output lengths $H \in \{96, 192, 336, 720\}$ for long-range forecasting, $H \in \{12, 24, 48\}$ for short-range forecasting, and $H \in \{1080, 1440, 1800, 2160\}$ for ultra-long-range forecasting. More descriptions about datasets, baselines, and experimental settings are given in Appendix B, C, and D, respectively.

5.2 Main Results

Long-Range Forecasting. Table 2 shows the results of long-range time series forecasting under multivariate settings. We can observe that: (1) Ada-MSHyper achieves the SOTA results in all datasets, with an average error reduction of 4.56% and 3.47% compared to the best baseline in MSE and MAE, respectively. (2) FEDformer and Autoformer exhibit relatively poor predictive performance. This may be that vanilla attention and simplistic decomposition techniques are insufficient in capturing multi-scale pattern interactions. (3) By considering multi-scale pattern interactions, TimeMixer achieves competitive results. However, its performance deteriorates on the datasets with low forecastability (e.g., ETTh1 and ETTh2 datasets). In contrast, Ada-MSHyper still maintains superiority on low forecastability datasets by modeling group-wise pattern interactions. Notably, despite modeling group-wise pattern interactions, the performance of MSHyper and PatchTST still lags behind that of Ada-MSHyper, indicating that predefined rules may overlook implicit interactions and introduce noise interference for forecasting. Moreover, for long-range time series forecasting under univariate settings, Ada-MSHyper gives an average error reduction of 7.57% and 4.65% compared to the best baseline in MSE and MAE, respectively. The univariate results are given in Appendix E.

Table 2: Results of long-range time series forecasting under multivariate settings. The best results are **bolded** and the second best results are underlined. Results are averaged from all prediction lengths. Full results are listed in Appendix E.

Models	Ada-MSHyper (Ours)		iTransformer (2024)		MSHyper (2024)		TimeMixer (2024)		MSGNet (2024)		CrossGNN (2023)		PatchTST (2023)		Crossformer (2023)		TimesNet (2023)		DLinear (2023)		FiLM (2022)		FEDformer (2022)		Autoformer (2021)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	**0.233**	**0.259**	0.258	0.278	0.250	0.279	0.245	0.276	0.249	0.278	0.247	0.289	0.259	0.281	0.259	0.315	0.259	0.287	0.265	0.317	0.253	0.309	0.309	0.360	0.338	0.382
Electricity	**0.167**	**0.259**	0.178	0.270	0.191	0.283	0.182	0.273	0.194	0.300	0.201	0.300	0.205	0.290	0.244	0.334	0.193	0.295	0.212	0.300	0.223	0.302	0.214	0.327	0.227	0.364
ETTh1	**0.418**	**0.426**	0.454	0.448	0.455	0.445	0.460	0.445	0.452	0.452	0.437	0.434	0.469	0.455	0.529	0.522	0.458	0.450	0.456	0.452	0.516	0.483	0.440	0.460	0.496	0.487
ETTh2	**0.371**	**0.394**	0.383	0.407	0.385	0.408	0.393	0.412	0.396	0.417	0.393	0.418	0.387	0.407	0.942	0.684	0.414	0.427	0.559	0.515	0.402	0.420	0.437	0.449	0.450	0.459
ETTm1	**0.365**	**0.390**	0.407	0.410	0.412	0.405	0.384	0.397	0.398	0.411	0.393	0.404	0.387	0.400	0.513	0.495	0.400	0.406	0.403	0.407	0.411	0.402	0.448	0.452	0.588	0.517
ETTm2	**0.263**	**0.322**	0.288	0.332	0.296	0.336	0.278	0.325	0.288	0.330	0.282	0.330	0.281	0.326	0.757	0.611	0.291	0.333	0.350	0.401	0.288	0.329	0.305	0.349	0.327	0.371
Traffic	**0.415**	**0.262**	0.428	0.282	0.433	0.283	0.492	0.304	0.641	0.370	0.583	0.323	0.481	0.304	0.550	0.304	0.620	0.336	0.625	0.383	0.637	0.384	0.610	0.376	0.628	0.379

Short-Range Forecasting. Table 3 summarizes the results of short-range time series forecasting under multivariate settings. It is notable that the PEMS datasets record multiple time series of citywide traffic networks and show complex spatial-temporal correlations among multiple variates. We adopt the same settings as iTransformer [26] and TimeMixer [20]. Ada-MSHyper still achieves the best performance in PEMS datasets, verifying its effectiveness in handling complex multivariate short-range time series forecasting. Specifically, Ada-MSHyper gives an average error reduction of 10.38% and 3.82% compared to the best baseline in terms of MSE and MAE, respectively.

Table 3: Results of short-range time series forecasting under multivariate settings. Results are averaged from all prediction lengths. Full results are listed in Appendix E.

Models	Ada-MSHyper (Ours)		iTransformer (2024)		MSHyper (2024)		TimeMixer (2024)		PatchTST (2023)		TimesNet (2023)		DLinear (2023)		Crossformer (2023)		SCINet (2022)		FEDformer (2022)		Autoformer (2021)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	**0.085**	**0.193**	0.096	0.204	0.123	0.226	0.188	0.361	0.151	0.265	0.119	0.225	0.219	0.328	0.138	0.253	0.093	0.203	0.167	0.291	0.546	0.536
PEMS04	**0.080**	**0.189**	0.098	0.207	0.147	0.250	0.183	0.363	0.162	0.279	0.109	0.220	0.242	0.350	0.145	0.267	0.085	0.194	0.195	0.308	0.510	0.537
PEMS07	**0.076**	**0.177**	0.088	0.190	0.128	0.234	0.172	0.351	0.166	0.270	0.106	0.208	0.241	0.343	0.181	0.272	0.112	0.211	0.133	0.252	0.304	0.409
PEMS08	**0.110**	**0.210**	0.127	0.212	0.220	0.260	0.189	0.374	0.238	0.289	0.150	0.244	0.281	0.366	0.232	0.276	0.133	0.225	0.234	0.323	0.623	0.573

Ultra-Long-Range Forecasting. Table 4 summarizes the results of ultra-long-range time series forecasting under multivariate settings. We can see that: (1) Ada-MSHyper achieves SOTA results in

almost all benchmarks, with an average error reduction of 4.97% and 2.21% compared to the best baseline in MSE and MAE, respectively. (2) Compared with other baselines, PatchTST and MSHyper achieve competitive results. The reason may be that group-wise interactions can help mitigate the issue of semantic information sparsity. (3) Compared to PatchTST and MSHyper, Ada-MSHyper achieves superior performance, the reason may be that the inter-scale interaction module can help capture macroscopic variations interactions, especially for the ultra-long-rang time series.

Table 4: Results of ultra-long-range time series forecasting under multivariate settings. Results are averaged from all prediction lengths. Full results are listed in Appendix E.

Models	Ada-MSHyper (Ours)		iTransformer (2024)		MSHyper (2024)		TimeMixer (2024)		WITRAN (2023)		PatchTST (2023)		DLinear (2023)		Crossformer (2023)		FEDformer (2022)		Pyraformer (2022)		Autoformer (2021)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.655**	**0.567**	0.766	0.611	0.745	0.610	0.840	0.631	0.734	0.833	0.699	0.588	0.696	0.624	0.921	1.091	0.765	0.637	1.083	0.831	0.808	0.675
ETTh2	**0.480**	**0.480**	0.541	0.518	0.513	0.495	0.563	0.523	0.547	0.537	0.508	0.498	1.218	0.787	2.530	1.233	0.625	0.574	3.263	1.509	0.658	0.648
ETTm1	**0.484**	0.463	0.554	0.495	0.544	0.480	0.523	0.483	0.532	0.476	0.503	**0.460**	0.540	0.498	3.555	1.483	0.522	0.501	1.093	0.811	0.631	0.550
ETTm2	**0.425**	**0.434**	0.468	0.449	0.464	0.447	0.465	0.449	0.446	**0.434**	0.462	0.448	0.655	0.574	3.555	1.483	0.487	0.475	4.566	1.745	0.516	0.491

5.3 Ablation Studies

AHL Module. To investigate the effectiveness of the AHL model, we conduct ablation studies by designing the following three variations: (1) Replacing the AHL module with adaptive graph learning module (-AGL). (2) Replacing the AHL model with one incidence matrix to capture group-wise node interactions at different scales (-one). (3) Replacing the AHL module with predefined multi-scale hypergraphs (-PH), i.e., each hyperedge connected a fixed number of nodes (4 in our experiment) in chronological order. The experimental results on ETTh1 dataset are shown in Table 5. We can observe that -AGL gets the worst forecasting results, indicating the importance of modeling group-wise interactions. In addition, -PH and -one perform worse than Ada-MSHyper, showing the effectiveness of adaptive hypergraph and multi-scale hypergraph, respectively.

Table 5: Results of different adaptive hypergraph learning methods and constraint mechanisms.

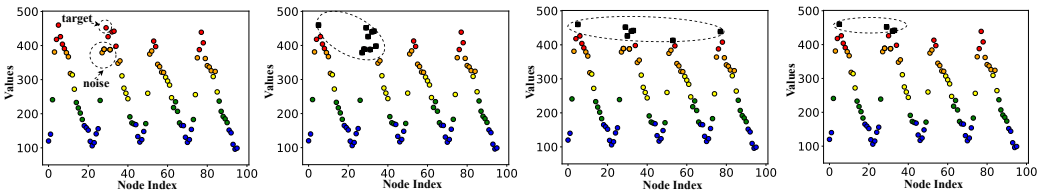
Variation	AGL		one		PH		w/o NC		w/o HC		w/o NHC		Ada-MSHyper	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
96	0.542	0.560	0.422	0.437	0.386	0.403	0.390	0.403	0.384	0.416	0.393	0.422	**0.372**	**0.393**
336	–	–	0.559	0.502	0.448	0.452	0.423	0.437	0.430	0.435	0.425	0.441	**0.422**	**0.433**
720	–	–	0.563	0.617	0.456	0.458	0.448	**0.457**	0.451	0.460	0.449	0.466	**0.445**	0.459

NHC Mechanism. To investigate the effectiveness of the NHC mechanism, we conduct ablation studies by designing the following three variations: (1) Removing the node constraint (-w/o NC). (2) Removing the hyperedge constraint (-w/o HC). (3) Removing the NHC mechanism (-w/o NHC). The experimental results on ETTh1 dataset are shown in Table 5. We can observe that Ada-MSHyper performs better than -w/o NC and -w/o HC, showing the effectiveness of node constraint and hyperedge constraint, respectively. In addition, -w/o NHC gets the worst forecasting results, which demonstrates the superiority of the NHC mechanism in adaptive hypergraph learning. More results about ablation studies are shown in Appendix F.

To further demonstrate the effectiveness of the node constraint in clustering nodes with similar semantic information, we present case visualization with -w/o NHC and -w/o HC on Electricity dataset. We randomly select one sample and plot the node values at the finest scale. We categorize the nodes into four groups based on the node values. Nodes with the same color indicate that they may have similar semantic information. As shown in Figure 3a, for the target node, nodes of other colors may be considered as noise. We drew the nodes related to the target node in black color based on incidence matrix $\mathbf{H}^1$. As shown in Figure 3b, due to the lack of node constraint, -w/o NHC can only capture the interactions among the target node and neighboring nodes and cannot distinguish nuanced noise information. In Figure 3c, with the node constraint, -w/o HC can cluster neighboring and distant but still strongly correlated nodes. In Figure 3d, with the NHC mechanism, Ada-MSHyper cannot only cluster nodes with similar semantic information but can differentiate temporal variations. The full visualization results are shown in Appendix H.

5.4 Parameter Studies

https://doi.org/10.52202/079017-1050



(a) Input sequence (b) -w/o NHC (c) -w/o HC (d) Ada-MSHyper

Figure 3: Visualization the node constraint effect on Electricity dataset.

We perform parameter studies to measure the impact of the number of scales (#scales) and the max number of hyperedges connected to a node (#hyperedges). The experimental results on ETTh1 dataset are shown in Figure 4, we can see that: (1) the best performance can be obtained when #scales is 3. The reason is that smaller #scales cannot provide sufficient pattern information and larger #scales may introduce excessive parameters and result in overfitting problems. (2) The optimal #hyperedges is 5. The reason is that smaller #hyperedges cannot capture group-wise interactions sufficiently and larger #hyperedges may introduce noise. More results about parameter studies are shown in Appendix G.

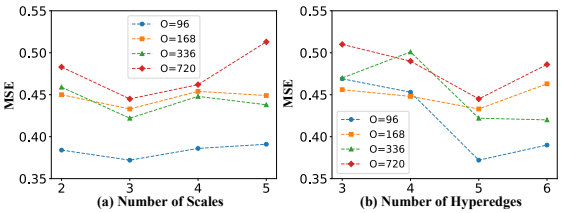


Figure 4: The impact of hyperparameters.

5.5 Computational Cost

We compare Ada-MSHyper with the two latest transformer-based methods, i.e., iTransformer and PatchTST, on traffic datasets with the output length of 96. The experimental results are shown in Table 6. Although we have a larger number of parameters, we achieve lower training time and lower GPU occupation due to the matrix sparsity operation in the model and the optimization of hypergraph computation provided by *torch_geometry*[2]. Considering the forecasting performance and the computation cost, Ada-MSHyper demonstrates its superiority over existing methods.

Table 6: Computation cost.

Methods	Training Time	# Parameters	GPU Occupation	MSE results
Ada-MSHyper	**6.499s**	8,965,392	**6,542MB**	**0.384**
iTransformer	7.863s	6,731,984	6,738MB	0.395
PatchTST	17.603s	**548,704**	9,788MB	0.526

6 Conclusions and Future Work

In this paper, we propose Ada-MSHyper with an adaptive multi-scale hypergraph for time series forecasting. Empowered by the AHL module and multi-scale interaction module, Ada-MSHyper can promote more comprehensive multi-scale group-wise pattern interactions, addressing the problem of semantic information sparsity. Experimentally, Ada-MSHyper achieves the SOTA performance, reducing prediction errors by an average of 4.56%, 10.38%, and 4.97% in MSE for long-range, short-range, and ultra-long-range time series forecasting, respectively. In addition, the visualization analysis and the ablation studies demonstrate the effectiveness of NHC mechanism in clustering nodes with similar semantic information and in addressing the issue of temporal variations entanglement.

In the future, this work can be extended in the following two aspects. First, since 2D spectrogram data may offer a better representation for time series forecasting, we will adapt our framework to the 2D spectrogram data in time-frequency domain. Second, since the features extracted by the MFE module may contain redundant information, we will design a disentangled multi-scale feature extraction module to extract more independent and representative features.

https://doi.org/10.52202/079017-1050

7 Acknowledgement

This work was supported by the Science Foundation of Donghai Laboratory (Grant No. DH-2022ZY0013).

References

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[2] Song Bai, Feihu Zhang, and Philip HS Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021.

[3] George EP Box and Gwilym M Jenkins. Some recent advances in forecasting and control. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 17(2):91–109, 1968.

[4] Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. MSGNet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 11141–11149, 2024.

[5] Haizhou Cao, Zhenhao Huang, Tiechui Yao, Jue Wang, Hui He, and Yangang Wang. InParformer: Evolutionary decomposition transformers with interactive parallel attention for long-term time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 6906–6915, 2023.

[6] Donghui Chen, Ling Chen, Zongjiang Shang, Youdong Zhang, Bo Wen, and Chenghu Yang. Scale-aware neural architecture search for multivariate time series forecasting. *ACM Transactions on Knowledge Discovery from Data*, 2024.

[7] Donghui Chen, Ling Chen, Youdong Zhang, Bo Wen, and Chenghu Yang. A multiscale interactive recurrent network for time-series forecasting. *IEEE Transactions on Cybernetics*, 52(9):8793–8803, 2021.

[8] Ling Chen, Donghui Chen, Zongjiang Shang, Binqing Wu, Cen Zheng, Bo Wen, and Wei Zhang. Multi-scale adaptive graph neural network for multivariate time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, pages 10748–10761, 2023.

[9] Ling Chen and Jiahua Cui. TPRNN: A top-down pyramidal recurrent neural network for time series forecasting. *arXiv preprint arXiv:2312.06328*, 2023.

[10] Zipeng Chen, Qianli Ma, and Zhenxi Lin. Time-aware multi-scale RNNs for time series modeling. In *IJCAI*, pages 2285–2291, 2021.

[11] Abhimanyu Das, Weihao Kong, Andrew Leach, Shaan K Mathur, Rajat Sen, and Rose Yu. Long-term forecasting with TiDE: Time-series dense encoder. *Transactions on Machine Learning Research*, 2023.

[12] Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 459–469, 2023.

[13] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3558–3565, 2019.

[14] Georg Goerg. Forecastable component analysis. In *International conference on machine learning*, pages 64–72. PMLR, 2013.

[15] Hansika Hewamalage, Klaus Ackermann, and Christoph Bergmeir. Forecast evaluation for data scientists: common pitfalls and best practices. *Data Mining and Knowledge Discovery*, 37(2):788–832, 2023.

 https://doi.org/10.52202/079017-1050

[16] Qihe Huang, Lei Shen, Ruixin Zhang, Shouhong Ding, Binwu Wang, Zhengyang Zhou, and Yang Wang. CrossGNN: Confronting noisy multivariate time series via cross interaction refinement. *Advances in Neural Information Processing Systems*, 36:46885–46902, 2023.

[17] Yuchi Huang, Qingshan Liu, and Dimitris Metaxas.] video object segmentation by hypergraph cut. In *2009 IEEE conference on computer vision and pattern recognition*, pages 1738–1745. IEEE, 2009.

[18] Yuxin Jia, Youfang Lin, Xinyan Hao, Yan Lin, Shengnan Guo, and Huaiyu Wan. WITRAN: Water-wave information transmission and recurrent acceleration network for long-range time series forecasting. *Advances in Neural Information Processing Systems*, 36, 2024.

[19] Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. SCINet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022.

[20] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *Proceedings of the International Conference on Learning Representations*, 2021.

[21] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2023.

[22] Yong Liu, Chenyu Li, Jianmin Wang, and Mingsheng Long. Koopa: Learning non-stationary time series dynamics with koopman predictors. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 12271–12290. Curran Associates, Inc., 2023.

[23] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2022.

[24] Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, Tyler Derr, and Rajiv Ratn Shah. Stock selection via spatiotemporal hypergraph attention network: A learning to rank approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 497–504, 2021.

[25] Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. *Advances in neural information processing systems*, 32, 2019.

[26] Zongjiang Shang and Ling Chen. MSHyper: Multi-scale hypergraph transformer for long-range time series forecasting. *arXiv preprint arXiv:2401.09261*, 2024.

[27] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and JUN ZHOU. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2023.

[28] Binqing Wu, Weiqi Chen, Wengwei Wang, Bingqing Peng, Liang Sun, and Ling Chen. WeatherGNN: Exploiting meteo- and spatial-dependencies for local numerical weather prediction bias-correction. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 2433–2441, 2024.

[29] 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.

[30] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, pages 22419–22430, 2021.

[31] Chenxin Xu, Maosen Li, Zhenyang Ni, Ya Zhang, and Siheng Chen. GroupNet: Multiscale hypergraph neural networks for trajectory prediction with relational reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6507, 2022.

[32] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. HyperGCN: A new method for training graph convolutional networks on hypergraphs. *Advances in Neural Information Processing Systems*, 32, 2019.

[33] Yichao Yan, Jie Qin, Jiaxin Chen, Li Liu, Fan Zhu, Ying Tai, and Ling Shao. Learning multi-granular hypergraphs for video-based person re-identification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2899–2908, 2020.

[34] Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudré-Mauroux. LBSN2Vec++: Heterogeneous hypergraph embedding for location-based social networks. *IEEE Transactions on Knowledge and Data Engineering*, 34(4):1843–1855, 2020.

[35] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*, 2022.

[36] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *Proceedings of the International Conference on Learning Representations*, 2023.

[37] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11106–11115, 2021.

[38] Tian Zhou, Ziqing Ma, Qingsong Wen, Liang Sun, Tao Yao, Wotao Yin, Rong Jin, et al. FiLM: Frequency improved legendre memory model for long-term time series forecasting. *Advances in Neural Information Processing Systems*, 35:12677–12690, 2022.

[39] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proceedings of the International Conference on Machine Learning*, pages 27268–27286, 2022.

 https://doi.org/10.52202/079017-1050

A Descriptions of Notations

To help understand the symbols used throughout the paper, we provide a detailed list of the key notations in Table 7.

Table 7: Description of the key notations.

Notation	Descriptions
$\mathcal{G}$	Hypergraph
$\mathcal{E}$	Hyperedge set
$\mathcal{V}$	Node set
N	Number of nodes
M	Number of hyperedges
T	Input length
H	Output length
D	Feature dimension
S	Total number of temporal scales
s	Scale index
$\mathbf{X}^{\mathrm{I}}_{1:T}$	Historical input sequence
$\mathbf{X}^{s}$	Sub-sequence at scale s
$\boldsymbol{x}_t$	Values at time step t
$\hat{\mathbf{X}}^{\mathrm{O}}_{T+1:T+H} \in \mathbb{R}^{H\times D}$	Forecasting results
$\boldsymbol{E}^{s}_{\mathrm{node}} \in \mathbb{R}^{N^s\times D}$	Node embeddings at scale s
$\boldsymbol{E}^{s}_{\mathrm{hyper}} \in \mathbb{R}^{M^s\times D}$	Hyperedge embeddings at scale s
e^{s}_{i}	ith hyperedge at scale s
v^{s}_{i}	ith node at scale s
$\boldsymbol{e}^{s}_{i}$	ith hyperedge feature representation at scale s
$\boldsymbol{v}^{s}_{i}$	ith node feature representation at scale s
η	Threshold of $TopK$ function
β	Threshold of the scale-specific incidence matrices
γ	Threshold of the Euclidean distance
λ	Balancing hyperparameter between node loss and hyperedge loss
$\mathbf{H}^{s}$	Incidence matrix at scale s
l^{s-1}	Size of the aggregation window at scale $s-1$
$\boldsymbol{\mathcal{V}}^{s}$	Initialized node feature representations at scale s
$\boldsymbol{\mathcal{E}}^{s}$	Initialized hyperedge feature representations at scale s
$\tilde{\boldsymbol{\mathcal{V}}}^{s}$	Updated node feature representations at scale s
$\tilde{\mathbf{V}}$	Updated hyperedge feature representations
$\mathcal{J}$	Number of heads
$\boldsymbol{\mathcal{H}}^{s}_{\jmath}$	Enriched incidence matrix of the $\jmath$th head at scale s
$\mathcal{N}(v^{s}_{i})$	Neighboring hyperedges connected to v^{s}_{i}
$\mathcal{N}(e^{s}_{i})$	Neighboring nodes connected by e^{s}_{i}
$\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{M\times D}$	Queries, keys, and values
$\alpha_{i,j}$	Correlation weight between ith and jth hyperedge representations
$D_{i,j}$	Euclidean distance between ith and jth hyperedge representations
L^{s}_{node}	Node constraint loss at scale s
L^{s}_{hyper}	Hyperedges constraint loss at scale s
L_{const}	Final constraint loss
L_{mse}	MSE loss

B Descriptions of Datasets

Datasets. For long-range time series forecasting, we conduct experiments on 7 commonly used benchmarks, including Electricity Transformers Temperature (ETT), Traffic[2], Electricity[3], and Weather[4] datasets following [30, 21, 26]. ETT datasets include data from two counties in the same Chinese province, each data point comprising seven variables: the target variable "oil temperature" and six power load features. The datasets vary in granularity, with "h" indicating hourly data and "m" indicating 15-minute intervals. Weather dataset contains 21 weather indicators collected every 10 minutes from a weather station in Germany. Electricity dataset records hourly electricity consumption of 321 clients. Traffic dataset provides hourly road occupancy rates from 821 freeway sensors. For short-range time series forecasting, we use four benchmarks from PEMS (PEMS03, PEMS04, PEMS07, and PEMS08), as referenced in [27, 19]. These datasets capture 5-minute traffic flow data from freeway sensors. For ultra-long-range time series forecasting, we adopt ETTh1, ETTh2, ETTm1, and ETTm2 following [18]. Table 8 gives the detailed dataset statistics. In addition, the forecastability is derived from one minus the entropy of the Fourier decomposition of a time series[27, 14]. Higher values mean greater forecastability.

Table 8: Detailed dataset statistics.

Task	Dataset	# Variates	Prediction Length	Frequency	Forecastability	Information
Long-term	ETTh1, ETTh2	7	(96, 192, 336,720)	Hourly	0.38, 0.45	Temperature
	ETTm1, ETTm2	7	(96, 192, 336,720)	15 mins	0.46, 0.55	Temperature
	Weather	21	(96, 192, 336,720)	10 mins	0.75	Weather
	Electricity	321	(96, 192, 336,720)	Hourly	0.77	Electricity
	Traffic	862	(96, 192, 336,720)	Hourly	0.68	Transportation
Short-term	PEMS03	358	12	5min	0.65	Transportation
	PEMS04	307	12	5mins	0.45	Transportation
	PEMS07	883	12	5mins	0.58	Transportation
	PEMS08	170	12	5mins	0.52	Transportation

We adopt the same data processing and train-validation-test split protocol as in existing works [26, 21, 23]. We split each dataset into training, validation, and test sets based on chronological order. For PEMS (PEMS03, PEMS04, PEMS07, and PEMS08) dataset and ETT (ETTh1, ETTh2, ETTm1, and ETTm2) dataset, the train-validation-test split ratio is 6:2:2. For Weather, Traffic, and Electricity dataset, the train-validation-test split ratio is 7:2:1.

Metric details. Following existing methods [26, 21], we employ Mean Squared Error (MSE) and Mean Absolute Error (MAE) as our evaluation metrics, which can be formulated as follows:

$$L_{mse} = \frac{1}{H} \left\| \widehat{\mathbf{X}}^{\mathrm{O}}_{T+1:T+H} - \mathbf{X}^{\mathrm{O}}_{T+1:T+H} \right\|_2^2 \tag{18}$$

$$L_{mae} = \frac{1}{H} \left| \widehat{\mathbf{X}}^{\mathrm{O}}_{T+1:T+H} - \mathbf{X}^{\mathrm{O}}_{T+1:T+H} \right|, \tag{19}$$

where T and H are the input and output lengths, $\widehat{\mathbf{X}}^{\mathrm{O}}_{T+1:T+H}$ and $\mathbf{X}^{\mathrm{O}}_{T+1:T+H}$ are the predicted results and ground truth.

C Descriptions of Baselines

We compare Ada-MSHyper with 15 competitive baselines. Below are brief descriptions of the baselines: (1) iTransformer [21]: Applies the attention and feed-forward network on the inverted dimensions, i.e., the time points of individual series are embedded into variate tokens, and the feed-forward network is applied for each variate token to learn nonlinear representations. (2) MSHyper [26]: Utilizes rule-based multi-scale hypergraphs to model high-order pattern interactions in univariate time series. (3) PatchTST [23]: Uses channel-independent techniques and treats subseries-level patches as

[2] http://pems.dot.ca.gov

[3] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

[4] https://www.bgc-jena.mpg.de/wetter/

https://doi.org/10.52202/079017-1050

input tokens to a Transformer, facilitating semantic extraction of multiple time steps in time series. (4) TimesMixer [27]: Employs a fully MLP-based architecture with past-decomposable-mixing and future-multipredictor-mixing blocks to leverage disentangled multiscale series. (5) MSGNet [4]: Leverages frequency domain analysis to extract periodic patterns and combines an attention mechanism with adaptive graph convolution to capture multi-scale pattern interactions. (6) CrossGNN [16]: Uses an adaptive multi-scale identifier to construct multi-scale representations and utilizes a cross-scale GNN to capture multi-scale pattern interactions. (7) TimesNet [29]: Conducts multi-periodicity analysis by extending 1D time series into a set of 2D tensors, modeling complex temporal variations from a 2D perspective. (8) WITRAN [18]: Proposes an RNN-based architecture that handles univariate input sequences from a 2D space perspective, maintaining a fixed scale throughout the processing. (9) SCINet [19]: Uses a recursive downsample-convolve-interact architecture to extract temporal features from downsampled sub-sequences or features. (10) Crossformer [36]: Adopts cross-dimension attention to capture inter-series dependencies for multivariate time series forecasting. (11) FiLM [38]: Applies Legendre polynomial projections to approximate historical information, uses Fourier projections to remove noise, and adds a low-rank approximation to speed up computation. (12) DLinear [35]: Decomposes time series into two different components and uses a single linear layer for each component to model temporal dependencies. (13) FEDformer [39]: Utilizes a seasonal-trend decomposition method to capture the global profile of time series and a frequency-enhanced Transformer to capture more detailed structures. (14) Pyraformer [20]: Utilizes a pyramidal attention module to extract inter-scale features at different resolutions and intra-scale features at different ranges with linear complexity. (15) Autoformer [30]: Uses an auto-correlation mechanism based on series periodicity to capture features at the sub-series level.

D Experimental Settings

We repeat all experiments 3 times and use the mean of the metrics as the final results. The training process is early stopped when there is no improvement within 5 epochs. Following existing works [21, 27, 35], we use instance normalization to normalize all datasets. The max number of scale S is set to 3. We use 1D convolution as our aggregation function. For other hyperparameters, we use Neural Network Intelligence (NNI)[5] toolkit to automatically search the best hyperparameters, which can greatly reduce computation cost compared to the grid search approach. The detailed search space of hyperparameters is given in Table 9. The source code of Ada-MSHyper is released on GitHub [6].

Table 9: The search space of hyperparameters.

Parameters	Choise
Batch size	{8, 16, 32, 64, 128}
Number of hyperedges at scale 1	{10, 20, 30, 50}
Number of hyperedges at scale 2	{5, 10, 15, 20}
Number of hyperedges at scale 3	{1, 2, 4, 5, 8, 12}
Aggregation window at scale 1	{2, 4, 8}
Aggregation window at scale 2	{2, 4}
η	{1, 3, 5, 10, 15, 20}
β	{0.2, 0.3, 0.4, 0.5}
γ	{0.2, 0.3, 0.4, 0.5}

E Full Results

We compare Ada-MSHyper with 13 baselines across four tasks: long-range forecasting for multivariate time series, long-range forecasting for univariate time series, ultra-long-range forecasting for multivariate time series, and short-range forecasting for multivariate time series. For a fair comparison, we evaluate Ada-MSHyper and baselines under unified experimental settings of each task. The average results from all prediction lengths are presented in tables, where the best results are **bolded**

[5] https://nni.readthedocs.io/en/latest/

[6] https://github.com/shangzongjiang/Ada-MSHyper

and the second best results are underlined. * indicates that some baselines do not meet our settings, thus we rerun these baselines using their official code and fine-tune their key hyperparameters.

Long-Range Time Series Forecasting Under Multivariate Settings. Table 10 summarizes the results of long-range time series forecasting under multivariate settings, where the results of baselines without * are cited from iTransformer [21]. We can see from Table 10 that Ada-MSHyper achieves the SOTA results on all datasets. Specifically, Ada-MSHyper gives an average error reduction of 4.56% and 3.47% compared to the best baseline in MSE and MAE, respectively.

Table 10: Full results of long-range time series forecasting under multivariate settings.

Models		Ada-MSHyper (Ours)		iTransformer (2024)		MSHyper* (2024)		TimeMixer* (2024)		MSGNet* (2024)		CrossGNN* (2023)		PatchTST (2023)		Crossformer (2023)		TimesNet (2023)		DLinear (2023)		FiLM* (2022)		FEDformer (2022)		Autoformer (2021)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.157**	**0.195**	0.174	0.214	0.170	0.223	0.163	0.210	0.163	0.212	0.159	0.218	0.177	0.218	0.158	0.230	0.172	0.220	0.196	0.255	0.199	0.262	0.217	0.296	0.266	0.336
	192	0.218	0.259	0.221	0.254	0.218	**0.253**	0.212	0.257	0.212	0.254	0.211	0.266	0.225	0.259	**0.206**	0.277	0.219	0.261	0.237	0.296	0.228	0.288	0.276	0.336	0.307	0.367
	336	**0.251**	**0.252**	0.278	0.296	0.269	0.300	0.263	0.292	0.272	0.299	0.267	0.310	0.278	0.297	0.272	0.335	0.280	0.306	0.283	0.335	0.267	0.323	0.339	0.380	0.359	0.395
	720	**0.304**	**0.328**	0.358	0.347	0.343	0.341	0.343	0.345	0.350	0.348	0.352	0.362	0.354	0.348	0.398	0.418	0.365	0.359	0.345	0.381	0.319	0.361	0.403	0.428	0.419	0.428
Electricity	96	**0.135**	**0.238**	0.148	0.240	0.176	0.261	0.153	0.247	0.165	0.274	0.173	0.275	0.181	0.270	0.219	0.314	0.168	0.272	0.197	0.282	0.198	0.274	0.193	0.308	0.201	0.317
	192	**0.152**	**0.239**	0.162	0.253	0.173	0.260	0.166	0.256	0.184	0.292	0.195	0.288	0.188	0.274	0.231	0.322	0.184	0.289	0.196	0.285	0.198	0.278	0.201	0.315	0.222	0.334
	336	**0.168**	**0.266**	0.178	0.269	0.195	0.297	0.185	0.277	0.195	0.302	0.206	0.300	0.204	0.293	0.246	0.337	0.198	0.300	0.209	0.301	0.217	0.300	0.214	0.329	0.231	0.338
	720	**0.212**	**0.293**	0.225	0.317	0.219	0.315	0.225	0.310	0.231	0.332	0.231	0.335	0.246	0.324	0.280	0.363	0.220	0.320	0.245	0.333	0.278	0.356	0.246	0.355	0.254	0.361
ETTh1	96	**0.372**	**0.393**	0.386	0.405	0.392	0.407	0.385	0.402	0.390	0.411	0.382	0.398	0.414	0.419	0.423	0.448	0.384	0.402	0.386	0.400	0.438	0.433	0.376	0.419	0.449	0.459
	192	0.433	**0.417**	0.441	0.436	0.440	0.426	0.443	0.430	0.442	0.442	0.427	0.425	0.460	0.445	0.471	0.474	0.436	0.429	0.437	0.432	0.493	0.466	**0.420**	0.448	0.500	0.482
	336	**0.422**	**0.433**	0.487	0.458	0.480	0.453	0.512	0.470	0.480	0.468	0.465	0.445	0.501	0.466	0.570	0.546	0.491	0.469	0.481	0.459	0.547	0.495	0.459	0.465	0.521	0.496
	720	**0.445**	**0.459**	0.503	0.491	0.508	0.493	0.498	0.476	0.494	0.488	0.472	0.468	0.500	0.488	0.653	0.621	0.521	0.500	0.519	0.516	0.586	0.538	0.506	0.507	0.514	0.512
ETTh2	96	**0.283**	**0.332**	0.297	0.349	0.300	0.351	0.296	0.347	0.328	0.371	0.309	0.359	0.302	0.348	0.745	0.584	0.340	0.374	0.333	0.387	0.322	0.364	0.358	0.397	0.346	0.388
	192	**0.358**	**0.374**	0.380	0.400	0.384	0.400	0.376	0.394	0.402	0.414	0.390	0.406	0.388	0.400	0.877	0.656	0.402	0.414	0.477	0.476	0.404	0.414	0.429	0.439	0.456	0.452
	336	0.428	0.437	0.428	**0.432**	0.443	0.438	0.434	0.443	0.435	0.443	**0.426**	0.444	**0.426**	0.433	1.043	0.731	0.452	0.452	0.594	0.541	0.435	0.445	0.496	0.487	0.482	0.486
	720	0.413	**0.432**	0.427	0.445	**0.412**	0.441	0.464	0.464	0.417	0.441	0.445	0.464	0.431	0.446	1.104	0.763	0.462	0.468	0.831	0.657	0.447	0.458	0.463	0.474	0.515	0.511
ETTm1	96	**0.301**	**0.354**	0.334	0.368	0.348	0.369	0.318	0.356	0.319	0.366	0.335	0.373	0.329	0.367	0.404	0.426	0.338	0.375	0.345	0.372	0.353	0.370	0.379	0.419	0.505	0.475
	192	**0.345**	**0.375**	0.377	0.391	0.392	0.391	0.366	0.385	0.376	0.397	0.372	0.390	0.367	0.385	0.450	0.451	0.374	0.387	0.380	0.389	0.389	0.387	0.426	0.441	0.553	0.496
	336	**0.375**	**0.397**	0.426	0.420	0.426	0.410	0.396	0.404	0.417	0.422	0.403	0.411	0.399	0.410	0.532	0.515	0.410	0.411	0.413	0.413	0.421	0.408	0.445	0.459	0.621	0.537
	720	**0.437**	**0.435**	0.491	0.459	0.483	0.448	0.454	0.441	0.481	0.458	0.461	0.442	0.454	0.439	0.666	0.589	0.478	0.450	0.474	0.453	0.481	0.441	0.543	0.490	0.671	0.561
ETTm2	96	**0.165**	**0.257**	0.180	0.264	0.183	0.267	0.175	0.258	0.177	0.262	0.176	0.266	0.175	0.259	0.287	0.366	0.187	0.267	0.193	0.292	0.183	0.266	0.203	0.287	0.255	0.339
	192	**0.230**	0.307	0.250	0.309	0.257	0.313	0.241	0.304	0.247	0.307	0.240	0.307	0.241	**0.302**	0.414	0.492	0.249	0.309	0.284	0.362	0.248	0.305	0.269	0.328	0.281	0.340
	336	**0.282**	**0.328**	0.311	0.348	0.335	0.361	0.303	0.343	0.312	0.346	0.304	0.345	0.305	0.343	0.597	0.542	0.321	0.351	0.369	0.427	0.309	0.343	0.325	0.366	0.339	0.372
	720	**0.375**	0.396	0.412	0.407	0.410	0.402	0.391	**0.394**	0.414	0.403	0.406	0.400	0.402	0.400	1.730	1.042	0.408	0.403	0.554	0.522	0.410	0.400	0.421	0.415	0.433	0.432
Traffic	96	**0.384**	**0.248**	0.395	0.268	0.413	0.272	0.473	0.288	0.605	0.344	0.570	0.310	0.462	0.295	0.522	0.290	0.593	0.321	0.650	0.396	0.647	0.384	0.587	0.366	0.613	0.388
	192	**0.401**	**0.258**	0.417	0.276	0.422	0.274	0.473	0.296	0.613	0.359	0.577	0.321	0.466	0.296	0.530	0.293	0.617	0.336	0.598	0.370	0.600	0.361	0.604	0.373	0.616	0.382
	336	**0.423**	**0.261**	0.433	0.283	0.438	0.292	0.508	0.312	0.642	0.376	0.588	0.324	0.482	0.304	0.558	0.305	0.629	0.336	0.605	0.373	0.610	0.367	0.621	0.383	0.622	0.337
	720	**0.453**	**0.282**	0.467	0.302	0.457	0.292	0.512	0.318	0.702	0.401	0.597	0.337	0.514	0.322	0.589	0.328	0.640	0.350	0.645	0.394	0.691	0.425	0.626	0.382	0.660	0.408

Long-Range Time Series Forecasting Under Univariate Settings. Table 11 and Table 12 summarize the average results and full results of long-range time series forecasting under univariate settings, where the results of baselines without * are cited from DLinear [35]. Following existing works [39, 26, 23, 30], we set the univariate forecasting on ETT as only predicting a target variate "oil temperature" given inputs from all variables. We can see from Table 11 that Ada-MSHyper achieves the SOTA results on all datasets. Specifically, Ada-MSHyper gives an average error reduction of 7.57% and 4.65% compared to the best baseline in terms of MSE and MAE, respectively.

Table 11: Long-range time series forecasting results under univariate settings.

Models	Ada-MSHyper (Ours)		iTransfomer* (2024)		MSHyper* (2024)		TimeMixer* (2024)		PatchTST* (2023)		DLinear (2023)		Crossformer* (2023)		Pyraformer (2022)		FEDformer (2022)		Autoformer (2021)		Informer (2021)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.071**	**0.203**	0.076	0.213	0.080	0.218	0.074	0.210	0.082	0.221	0.104	0.247	0.099	0.246	0.170	0.335	0.111	0.257	0.105	0.252	0.199	0.377
ETTh2	**0.171**	**0.329**	0.199	0.352	0.187	0.338	0.198	0.350	0.198	0.348	0.198	0.350	0.201	0.349	0.215	0.373	0.206	0.350	0.218	0.364	0.243	0.400
ETTm1	**0.047**	**0.159**	0.053	0.174	0.053	0.172	0.053	0.173	0.052	0.171	0.054	0.168	0.065	0.196	0.255	0.392	0.069	0.202	0.081	0.221	0.281	0.441
ETTm2	**0.103**	**0.230**	0.128	0.268	0.120	0.258	0.121	0.258	0.121	0.258	0.112	0.248	0.119	0.256	0.133	0.273	0.119	0.262	0.130	0.271	0.175	0.320

Table 12: Full results of long-range time series forecasting under univariate settings.

Models		Ada-MSHyper (Ours)		iTransfomer* (2024)		MSHyper* (2024)		TimeMixer* (2024)		PatchTST* (2023)		DLinear (2023)		Crossformer (2023)		Pyraformer (2022)		FEDformer (2022)		Autoformer (2021)		Informer (2021)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.057**	**0.173**	0.059	0.185	**0.056**	0.181	0.057	0.181	**0.056**	0.181	**0.056**	0.180	0.076	0.216	0.099	0.277	0.079	0.215	0.071	0.206	0.193	0.377
	192	**0.072**	**0.198**	0.073	0.208	0.076	0.211	0.072	0.204	0.076	0.210	**0.071**	0.204	0.085	0.225	0.174	0.346	0.104	0.245	0.114	0.262	0.217	0.395
	336	**0.070**	**0.213**	0.084	0.223	0.090	0.236	0.085	0.227	0.094	0.242	0.098	0.244	0.106	0.257	0.198	0.370	0.119	0.270	0.107	0.258	0.202	0.381
	720	0.085	0.228	0.089	0.236	0.096	0.245	**0.083**	**0.227**	0.101	0.250	0.189	0.359	0.128	0.287	0.209	0.348	0.142	0.299	0.126	0.283	0.183	0.355
ETTh2	96	**0.116**	**0.262**	0.136	0.287	0.117	0.266	0.133	0.283	0.130	0.276	0.131	0.279	0.125	0.273	0.152	0.303	0.128	0.271	0.153	0.306	0.213	0.373
	192	**0.168**	**0.323**	0.187	0.342	0.172	0.325	0.190	0.341	0.181	0.331	0.176	0.329	0.187	0.334	0.197	0.370	0.185	0.330	0.204	0.351	0.227	0.387
	336	**0.177**	**0.350**	0.219	0.374	0.211	0.362	0.226	0.379	0.226	0.379	0.209	0.367	0.227	0.377	0.238	0.385	0.231	0.378	0.246	0.389	0.242	0.401
	720	**0.221**	**0.380**	0.253	0.403	0.248	0.398	0.241	0.396	0.253	0.406	0.276	0.426	0.266	0.410	0.274	0.435	0.278	0.420	0.268	0.409	0.291	0.439
ETTm1	96	**0.027**	**0.118**	0.029	0.127	0.029	0.127	0.029	0.128	0.029	0.126	0.028	0.123	0.035	0.145	0.127	0.281	0.033	0.140	0.056	0.183	0.109	0.277
	192	**0.038**	**0.148**	0.045	0.162	0.044	0.159	0.044	0.160	0.043	0.158	0.045	0.156	0.055	0.180	0.205	0.343	0.058	0.186	0.081	0.216	0.151	0.310
	336	**0.052**	**0.165**	0.059	0.189	0.059	0.186	0.058	0.185	0.056	0.183	0.061	0.182	0.072	0.209	0.302	0.457	0.084	0.231	0.076	0.218	0.427	0.591
	720	**0.071**	**0.206**	0.080	0.218	0.080	0.217	0.081	0.218	0.080	0.217	0.080	0.210	0.097	0.248	0.387	0.485	0.102	0.250	0.110	0.267	0.438	0.586
ETTm2	96	**0.051**	**0.163**	0.071	0.193	0.071	0.194	0.068	0.187	0.071	0.192	0.063	0.183	0.058	0.183	0.074	0.208	0.067	0.198	0.065	0.189	0.088	0.225
	192	**0.089**	**0.207**	0.109	0.248	0.102	0.238	0.101	0.236	0.102	0.237	0.092	0.227	0.105	0.237	0.116	0.252	0.102	0.245	0.118	0.256	0.132	0.283
	336	**0.114**	**0.240**	0.141	0.289	0.129	0.274	0.133	0.278	0.130	0.274	0.119	0.261	0.133	0.280	0.143	0.295	0.130	0.279	0.154	0.305	0.180	0.336
	720	**0.156**	**0.310**	0.190	0.343	0.176	0.324	0.183	0.332	0.179	0.328	0.175	0.320	0.181	0.324	0.197	0.338	0.178	0.325	0.182	0.335	0.300	0.435

Short-Range Time Series Forecasting Under Multivariate Settings. Table 13 summarizes the results of short-range time series forecasting under multivariate settings, where the results of baselines

https://doi.org/10.52202/079017-1050

without * are cited from iTransformer [21]. We can see from Table 13 that Ada-MSHyper achieves the SOTA results on all datasets. Specifically, Ada-MSHyper gives an average error reduction of 10.38% and 3.82% compared to the best baseline in terms of MSE and MAE, respectively.

Table 13: Full results of short-range time series forecasting under multivariate settings.

Models		Ada-MSHyper (Ours)		iTransformer* (2024)		MSHyper* (2024)		TimeMixer* (2024)		PatchTST (2023)		TimesNet (2023)		DLinear (2023)		Crossformer (2023)		SCINet (2022)		FEDformer (2022)		Autoformer (2021)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	12	**0.060**	**0.165**	0.071	0.174	0.106	0.207	0.161	0.323	0.099	0.216	0.085	0.192	0.122	0.243	0.090	0.203	0.066	0.172	0.126	0.251	0.272	0.385
	24	**0.075**	**0.184**	0.093	0.201	0.126	0.207	0.181	0.352	0.142	0.259	0.118	0.223	0.201	0.317	0.121	0.240	0.085	0.198	0.149	0.275	0.334	0.440
	48	**0.120**	**0.230**	0.125	0.236	0.138	0.265	0.222	0.407	0.211	0.319	0.155	0.260	0.333	0.425	0.202	0.317	0.127	0.238	0.227	0.348	1.032	0.782
PEMS04	12	**0.068**	**0.173**	0.078	0.183	0.103	0.197	0.168	0.344	0.105	0.224	0.087	0.195	0.148	0.272	0.098	0.218	0.073	0.177	0.138	0.262	0.424	0.491
	24	**0.080**	**0.189**	0.095	0.205	0.148	0.245	0.183	0.362	0.153	0.275	0.103	0.215	0.224	0.340	0.131	0.256	0.084	0.193	0.177	0.293	0.459	0.509
	48	**0.093**	**0.204**	0.120	0.233	0.191	0.308	0.199	0.383	0.229	0.339	0.136	0.250	0.355	0.437	0.205	0.326	0.099	0.211	0.270	0.368	0.646	0.610
PEMS07	12	**0.055**	**0.154**	0.067	0.165	0.137	0.256	0.151	0.322	0.095	0.207	0.082	0.181	0.115	0.242	0.094	0.200	0.068	0.171	0.109	0.225	0.199	0.336
	24	**0.065**	**0.172**	0.088	0.190	0.111	0.225	0.169	0.348	0.150	0.262	0.101	0.204	0.210	0.329	0.139	0.247	0.119	0.225	0.125	0.244	0.323	0.420
	48	**0.107**	**0.204**	0.110	0.215	0.137	0.221	0.196	0.384	0.253	0.340	0.134	0.238	0.398	0.458	0.311	0.369	0.149	0.237	0.165	0.288	0.390	0.470
PEMS08	12	**0.063**	**0.165**	0.079	0.182	0.113	0.209	0.162	0.337	0.168	0.232	0.112	0.212	0.154	0.276	0.165	0.214	0.087	0.184	0.173	0.273	0.436	0.485
	24	**0.109**	0.229	0.115	**0.219**	0.230	0.248	0.181	0.364	0.224	0.281	0.141	0.238	0.248	0.353	0.215	0.260	0.122	0.221	0.210	0.301	0.467	0.502
	48	**0.159**	0.238	0.186	**0.235**	0.317	0.324	0.224	0.422	0.321	0.354	0.198	0.283	0.440	0.470	0.315	0.355	0.189	0.270	0.320	0.394	0.966	0.733

Ultra-Long-Range Time Series Forecasting Under Multivariate Settings. We conduct ultra-long-range time series forecasting by taking fixed input length ($T = 96$) to predict ultra-long lengths ($H = \{1080, 1440, 1800, 2160\}$). We run all results by ourselves. Table 14 summarizes the results of ultra-long-range time series forecasting under multivariate settings, where - - indicates that the method fails to produce any results on that prediction length due to the out-of-memory problems. We can see from Table 14 that Ada-MSHyper achieves SOTA results on almost all datasets. Specifically, Ada-MSHyper gives an average error reduction of 4.97% and 2.21% compared to the best baseline in terms of MSE and MAE, respectively.

Table 14: Full results of ultra-long-range time series forecasting results under multivariate settings.

Models		Ada-MSHyper (Ours)		iTransformer* (2024)		MSHyper* (2024)		TimeMixer* (2024)		WITRAN* (2023)		PatchTST* (2023)		Dlinear* (2023)		Crossformer* (2023)		FEDformer* (2022)		Pyraformer* (2022)		Autoformer* (2021)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	1080	**0.534**	**0.509**	0.562	0.521	0.557	0.517	0.682	0.569	0.602	0.660	0.549	0.512	0.593	0.564	0.877	1.204	0.699	0.615	1.015	0.798	0.695	0.626
	1440	**0.616**	**0.498**	0.620	0.556	0.667	0.578	0.793	0.625	0.705	0.878	0.619	0.553	0.661	0.607	0.863	1.175	0.621	0.567	1.075	0.833	0.876	0.696
	1800	**0.689**	0.627	0.780	0.631	0.758	0.624	0.877	0.643	0.775	**0.623**	0.775	**0.623**	0.746	0.658	0.849	1.163	0.806	0.649	1.111	0.844	0.852	0.704
	2160	**0.779**	**0.635**	1.102	0.736	0.998	0.721	1.007	0.686	0.852	1.171	0.851	0.665	0.783	0.667	1.095	0.821	0.935	0.717	1.129	0.847	–	–
ETTh2	1080	**0.426**	**0.461**	0.486	0.488	0.464	0.469	0.483	0.480	0.432	0.474	0.453	0.468	0.730	0.617	1.481	0.918	0.514	0.526	3.224	1.458	0.559	0.547
	1440	**0.465**	**0.437**	0.512	0.507	0.524	0.506	0.547	0.510	0.472	0.443	0.513	0.501	1.144	0.770	1.901	1.044	0.578	0.546	3.254	1.548	0.638	0.708
	1800	**0.503**	0.505	0.565	0.529	0.522	**0.496**	0.606	0.544	0.626	0.610	0.517	0.503	1.327	0.840	3.109	1.486	0.645	0.584	3.328	1.565	0.776	0.689
	2160	**0.527**	0.515	0.600	0.546	0.542	**0.510**	0.616	0.557	0.657	0.619	0.547	0.519	1.670	0.919	3.630	1.485	0.762	0.639	3.246	1.465	–	–
ETTm1	1080	**0.460**	**0.445**	0.534	0.483	0.520	0.465	0.502	0.465	0.464	0.459	0.494	0.459	0.514	0.479	2.588	1.236	0.513	0.499	1.071	0.793	0.651	0.551
	1440	**0.473**	**0.449**	0.556	0.495	0.542	0.477	0.523	0.488	0.543	0.467	0.508	0.467	0.534	0.491	2.946	1.349	0.511	0.494	1.136	0.834	0.602	0.542
	1800	**0.492**	0.475	0.571	0.501	0.564	0.490	0.526	0.487	0.550	0.497	0.504	**0.434**	0.556	0.507	4.113	1.602	0.514	0.496	1.111	0.812	0.641	0.558
	2160	0.510	0.483	0.555	0.499	0.550	0.487	0.542	0.491	0.569	**0.481**	**0.507**	**0.481**	0.556	0.515	4.574	1.743	0.551	0.516	1.054	0.804	–	–
ETTm2	1080	**0.404**	**0.416**	0.463	0.438	0.464	0.439	0.450	0.432	0.415	0.434	0.449	0.432	0.559	0.519	2.587	1.236	0.501	0.468	4.879	1.733	0.527	0.489
	1440	**0.413**	**0.429**	0.475	0.452	0.475	0.449	0.471	0.452	0.442	0.442	0.475	0.452	0.699	0.593	2.946	1.349	0.495	0.480	4.429	1.708	0.519	0.489
	1800	**0.435**	0.432	0.468	0.453	0.454	0.449	0.464	0.452	0.479	**0.410**	0.456	0.449	0.721	0.612	4.113	1.602	0.477	0.474	4.502	1.780	0.503	0.496
	2160	0.449	0.457	0.467	0.454	0.463	0.451	0.473	0.459	**0.447**	**0.449**	0.466	0.457	0.639	0.572	4.574	1.743	0.473	0.477	4.454	1.758	–	–

F Ablation Studies

To investigate the performance of Ada-MSHyper on longer prediction lengths, we compare the forecasting results of Ada-MSHyper with those of six variations (i.e., AGL, one, PH, -w/o NC, -w/o HC, and -w/o NHC) on ETTh1 dataset. The experimental results are shown in Table 15. We can observe that for longer prediction lengths, -w/o NC has smaller performance degradation than other variations. The reason may be that when the prediction length increases, the model tends to focus more on macroscopic variation interactions and diminishes its emphasis on fine-grained node constraint. In addition, Ada-MSHyper performs better than other six variations even with longer prediction length, showing the effectiveness of our AHL module and NHC mechanism.

Table 15: Results of different adaptive hypergraph learning methods and constraint mechanisms.

Variation	AGL		one		PH		-w/o NC		-w/o HC		-w/o NHC		Ada-MSHyper	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
1080	-	-	0.685	0.679	0.640	0.591	0.539	0.515	0.574	0.516	0.597	0.525	**0.534**	**0.509**
1440	-	-	0.855	0.857	0.783	0.673	0.621	0.503	0.679	0.568	0.734	0.585	**0.616**	**0.498**

To investigate the impact of node and hypergraph constraints mechanism on the adaptive hypergraph learning (AHL) module, we design two variants: (1) Removing the NHC mechanism (-w/o NHC).

(2) Only optimizing the hypergraph learning module (-OH). We illustrate these two variants for better understanding in Figure 5.

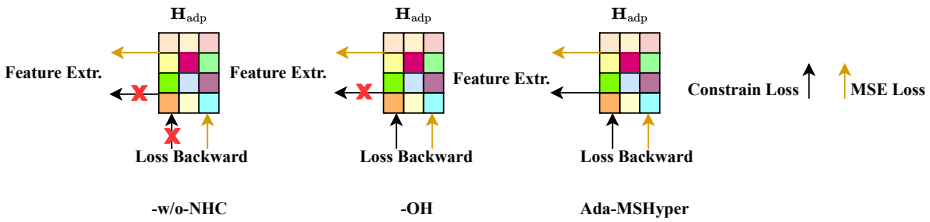


Figure 5: Different optimization strategies for AHL.

To investigate the impact of the multi-scale feature extraction (MFE) module, we design two variants: (1) Replacing the aggregation function in the MFE module with average pooling (-avg). (2) Replacing the aggregation function in the MFE module with max pooling (-max).

The results for the four variants are shown in Table 16. We can observe that: (1) -w/o NHC gets the worst results. This may be because lacking the constraints makes the model fail to capture implicit semantic features of the clustered nodes and learned hyperedges. (2) Compared to -OH, Ada-MSHyper yields slightly better results. The reason may be that the NHC mechanism facilitates the MFE model to more effectively aggregate similar features during multi-scale feature extraction and reduce the interference of noise. (3) -avg and -max get relatively worse performance. This may be because the lack of parameters leads to the reduction of the representative ability of Ada-MSHyper.

Table 16: Results of different AHL, MFE and multi-scale interaction methods.

Methods	-w/o NHC		-OH		-avg		-max		-r/att		Ada-MSHyper	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
96	0.393	0.422	0.379	0.397	0.372	0.413	0.382	0.408	0.418	0.419	**0.372**	**0.393**
336	0.486	0.459	0.423	0.439	0.429	0.440	0.426	0.437	0.483	0.454	**0.422**	**0.433**
720	0.515	0.487	0.447	0.460	0.453	0.462	0.448	0.464	0.514	0.507	**0.445**	**0.459**

To investigate the effectiveness of hypergraph convolution attention, we design one variant: Replacing the hypergraph convolution attention with the attention mechanism used in the inter-scale interaction module to update node features (-r/att). The experimental results on ETTh1 dataset are shown in Table 16. We can observe that Ada-MSHyper performs better than -r/ att, which demonstrates the effectiveness of the hyperedge convolution attention used in the intra-scale interaction module.

G Parameter Studies

We perform parameter studies to measure the impact of the threshold η, which influences the effectiveness of the sparsity strategy. The experimental results on ETTh1 dataset are shown in Table 17. We can see that the best performance can be obtained when η is 3. The reason is that a small η may filter out useful information and a large η would introduce noise interference.

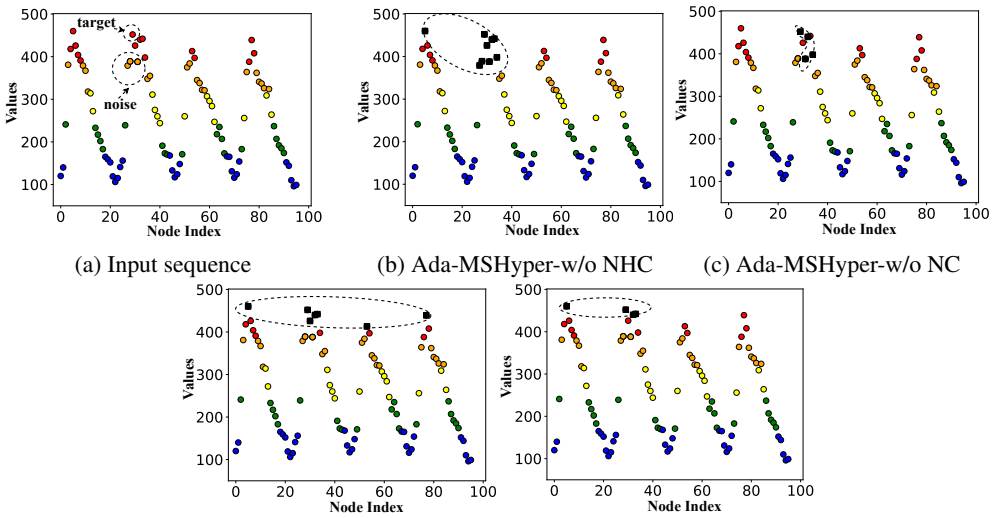
Table 17: Results of Ada-MSHyper with different η.

Hyparameter	η=1		η=2		η=3		η=4		η=5	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
96	0.407	0.415	0.390	0.397	**0.372**	**0.393**	0.387	0.396	0.419	0.418
336	0.547	0.500	0.476	0.443	**0.422**	**0.433**	0.438	0.435	0.560	0.510
720	0.450	0.463	0.476	0.465	**0.445**	**0.459**	0.460	0.459	0.473	0.474

H Visualization

Visualization of Node Constraint. As shown in Figure 6a, each time step is denoted a node of the hypergraph at the finest scale. We categorize the nodes into four groups based on the node values of

https://doi.org/10.52202/079017-1050



(a) Input sequence (b) Ada-MSHyper-w/o NHC (c) Ada-MSHyper-w/o NC

(d) Ada-MSHyper-w/o HC (e) Ada-MSHyper

Figure 6: Visualization the node constrain effect on Electricity dataset.

original inputs, and draw them using different colors. For a target node, nodes of the same color may be regarded as those sharing similar semantic information with the target node, while nodes of other colors may be regarded as noise. Then, we draw the nodes related to the target node based on the incidence matrix $\mathbf{H}^1$ of the learned hypergraph in the black color.

We random select samples at the same time step from three variants, i.e., without node constraint (-w/o NC), without hyperedge constraint (-w/o HC), and without node and hyperedges constraints (-w/o NHC), and plot these three samples with samples from original inputs and Ada-MHyper.

We can observe that: (1) In Figure 6b and Figure 6c, the related nodes of the target node are almost neighboring nodes. However, some noise, plotted as orange, is included as well. The reason may be that -w/o NHC and -w/o NC cannot distinguish noise information without node constraint, i.e., cannot consider the semantic similarity to cluster nodes. (2) In Figure 6d and Figure 6e, since -w/o HC and the proposed Ada-MSHyper have node constraint, both of them can cluster neighboring and distant but strongly correlated nodes, and they can also mitigate the interference of noise, indicating the effectiveness of node constraint.

Visualization of Hyperedge Constraint. We use the samples which are used in the visualization of node constraint. We visualize the sequentially connecting nodes that belong to the same hyperedges whose indices are $\{4, 8, 12\}$. Figure 7 shows three types of temporal variations learned by -w/o NC, -w/o HC, -w/o NHC, and Ada-MSHyper, respectively. We can observe that: (1) -w/o HC and -w/o NHC exhibit irregular temporal variations in comparison to -w/o NC and Ada-MSHyper. The reason may be that without the hyperedge constraint, these methods are unable to adequately differentiate temporal variations entangled in temporal patterns. (2) Compared to -w/o NC, Ada-MSHyper exhibits relatively simple temporal variations. The reason may be that influenced by node constraint, the temporal variations extracted by Ada-MSHyper contain less noise.

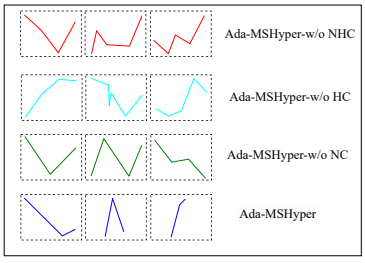
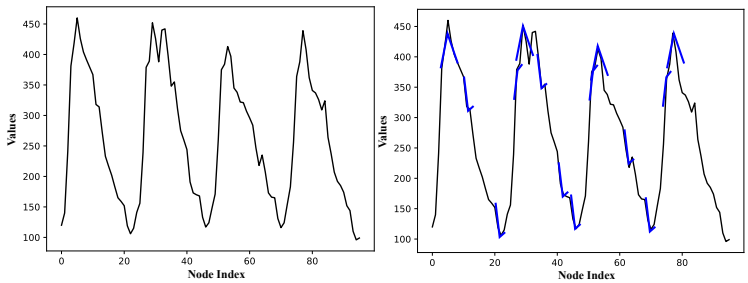


Figure 7: Different temporal variations learned by different methods.

https://doi.org/10.52202/079017-1050

We also matched the temporal variations extracted by Ada-MSHyper to the sample sequences. As shown in Figure 8b, we can observe that these variations can represent inherent changes. We speculate that by introducing hyperedge constraint, the model will treat temporal variations with different shapes as distinct positive and negative examples. In addition, the differentiated temporal variations are like a kind of Shapelet, akin to those used in NLP and CV, enabling a better representation of temporal patterns within time series.



(a) Input sequence (b) Ada-MSHyper

Figure 8: Visualization the hyperedge constraint effect on Electricity dataset.

I Limitations and Future Works

In the future, we will extend our work in the following directions. Firstly, due to our NHC mechanism can cluster nodes with similar semantic information and differentiate temporal variations within each scales, It is interesting to correlate the inherent temporal variations with corpora used in natural language processing, and leverage large language models to investigate deeper correlations between corpora and TS data. Secondly, compared to natural language processing and computer vision, time series analysis has access to fewer datasets, which may limit the expressive power of the models. Therefore, in the future, we plan to compile larger datasets to validate the generalization capabilities of our models on more extensive data.

J Broader Impacts

In this paper, we propose Ada-MSHyper for time series forecasting. Extensive experimental results demonstrate the effectiveness of Ada-MSHyper. Our paper mainly focuses on scientific research and has no obvious negative social impact.

https://doi.org/10.52202/079017-1050

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the contributions and scope of the paper (see Abstract and Introduction)

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in AppendixI.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The paper provides corresponding experimental validation in Section5.2 and AppendixE, provides ablation studies in Section 5.3 and Appendix F, and provides visualization analysis in Appendix H to support the claimed capabilities of the model.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We have provided the details regarding computational platforms, dataset descriptions, network architectures, hyper-parameter settings, and the training process of our method in Section B in the main paper and Appendix B, C, and D. In addition, we provide source codes on anonymous Github as stated in the abstract.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

https://doi.org/10.52202/079017-1050

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our codes are released at anonymous Github as stated in the abstract. The download links of the public datasets are provided in the project homepage and pre-processing functions are included in the codes. The hyper-parameter settings are given in Appendix D.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have provided the details regarding computational platforms, dataset descriptions, network architectures, hyper-parameter settings, and the training process of our method in Section B in the main paper and Appendix B, C, and D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We repeat all experiments 3 times and use the mean of the metrics as the final results as illustrated in Appedix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have stated the experimental platforms (i.e., GPUs and CPUs) and software (the version of Pytorch) used in our paper in Section 5.1 and Appendix D, and we have analyzed the computational efficiency in Appendix **??**.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and we have ensured to preserve anonymity.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

https://doi.org/10.52202/079017-1050

Justification: We discuss both potential positive societal impacts and negative societal impacts of our work in Appendix J.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our paper poses no such risks, as we focus on theoretical analysis and general research areas, and conduct our experiments on public datasets.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We make sure to cite the original papers (or URLs) of the code packages or datasets that are used in our paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our codes are provided on the project homepage at anonymous Github.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No crowdsourcing nor research with human subjects is involved.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No crowdsourcing nor research with human subjects is involved.

    Guidelines:

https://doi.org/10.52202/079017-1050

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.