Hermes: A Multi-Scale Spatial-Temporal Hypergraph Network for Stock Time Series Forecasting

Xiangfei Qiu[1], **Liu Yang**[1], **Xiangyu Xu**[1], **Hanyin Cheng**[1], **Xingjian Wu**[1], **Rongjia Wu**[2], **Zhigang Zhang**[2], **Ding Tu**[2], **Chenjuan Guo**[1], **Bin Yang**[1], **Christian S. Jensen**[3], **Jilin Hu**[1*]
[1]*East China Normal University, Shanghai, China*
[2]*CFETS Information Technology (Shanghai) Co., Ltd, Shanghai, China*
[3]*Aalborg University, Aalborg, Denmark*

Abstract

Time series forecasting occurs in a range of financial applications providing essential decision-making support to investors, regulatory institutions, and analysts. Unlike multivariate time series from other domains, stock time series exhibit industry correlation. Exploiting this kind of correlation can improve forecasting accuracy. However, existing methods based on hypergraphs can only capture industry correlation relatively superficially. These methods face two key limitations: they do not fully consider inter-industry lead-lag interactions, and they do not model multi-scale information within and among industries. This study proposes the **Hermes** framework for stock time series forecasting that aims to improve the exploitation of industry correlation by addressing these limitations. The framework integrates moving aggregation and multi-scale fusion modules in a hypergraph network. Specifically, to more flexibly capture the lead-lag relationships among industries, Hermes proposes a hyperedge-based moving aggregation module. This module incorporates a sliding window and utilizes dynamic temporal aggregation operations to consider lead-lag dependencies among industries. Additionally, to effectively model multi-scale information, Hermes employs cross-scale, edge-to-edge message passing to integrate information from different scales while maintaining the consistency of each scale. Experimental results on multiple real-world stock datasets show that Hermes outperforms existing state-of-the-art methods.

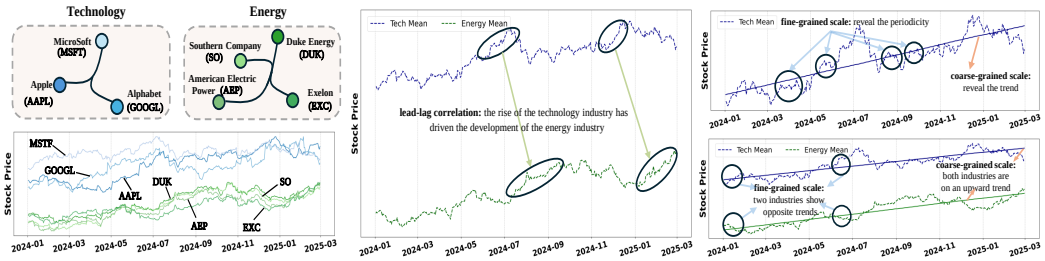
1 Introduction

Stock time series are typically composed of data from multiple stocks, each of which includes multiple key indicators such as stock opening prices, closing prices, etc. Fan and Shen (2024); Li et al. (2024). Stock time series forecasting (STSF) is important in financial applications, providing crucial decision-making support for investors, regulatory institutions, and analysts, helping them make data-driven decisions in complex and ever-changing market environments Rather et al. (2017); Faloutsos et al. (2018); Jacob and Shasha (1999); Zheng et al. (2023).

Most existing studies treat stocks in stock time series independently, thus disregarding the interconnected nature of market dynamics and contradicting the intrinsic interdependencies observed in real-world financial systems Diebold and Yılmaz (2014). In reality, stocks are often interrelated, and there are rich signals in the relationships between them Nobi et al. (2014). For example, with the breakthroughs of Artificial Intelligence technology, companies like Apple, Microsoft, and Alphabet that actively invest and innovate in this domain are expected to lead in the market due to their technological advantages. It is one of the important factors for their stock prices to show a steady upward trend in the overall stock market[1] —see Figure 1a. At the same time, the soaring power

[1] www.morningstar.co.uk/uk/news/258865/tech-and-communication-stocks-drove-us-market-gains-in-2024.aspx

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Intra-industry correlation (b) Lead-lag correlation (c) Multi-scale correlation

Figure 1: (a): Intra-industry correlation. Relations between stocks within an industry (hyperedge) can be represented by a hypergraph. (b): Lead-lag correlation. (c): Multi-scale correlation.

demand, driven by electrification, decarbonization, and the advent of artificial intelligence, led to a concurrent rise in the stock prices of companies in the energy sector[2]—see Figure 1a. Such industry correlations have gradually attracted more attention. Especially within the same industry, different stocks often exhibit correlations, reflecting the common economic environment, market fluctuations, policy changes, and other factors that impact the stocks.

Hypergraphs can connect multiple graph nodes (representing, e.g., stocks) through hyperedges. By connecting stocks within the same industry by hyperedges to construct a hypergraph network, it is possible to capture more comprehensively the interrelationships and dynamics among stocks within an industry. However, existing hypergraph-based STSF methods that consider industry correlations still have the following challenges.

Challenge 1: Current methods fail to fully consider inter-industry lead-lag interactions. In financial markets, there are time dependencies among industries, and these relationships are highly diverse and complex Qiu et al. (2025a, 2026a). As shown in Figure 1b, the technology industry has flourished with breakthroughs in key technologies and market innovations, achieving significant progress. Technological innovations from technology companies have injected new vitality into the development of renewable energy, and energy companies have leveraged technologies such as big data, AI, and the Internet of Things to enhance the efficiency of energy production and management[2], thereby driving up stock prices Abid et al. (2025); Olanrewaju et al. (2024); Rafiei et al. (2025). Therefore, in the modeling process, it is important to capture lead-lag correlations among industries more accurately, so as to better reflect the dynamic interactions and dependencies in financial markets, ultimately improving the accuracy and practicality of the forecasts.

Challenge 2: Current methods do not model multi-scale information within and among industries. Stock time series inherently possess multi-scale characteristics, and markets exhibit different features and patterns at different time scales. For example, in Figure 1c, we can observe a clear overall trend at a coarse-grained scale, and a cyclical pattern at a fine-grained scale. Moreover, two different industries, may exhibit correlations at a coarse-grained scale. However, delving deeper into a fine-grained analysis might find that two industries show opposite trends. Single-scale analysis methods cannot fully capture such characteristics, so there is a pressing need to adopt multi-scale analysis to reveal the dynamic changes in financial markets across different time scales, thereby providing richer and more accurate information for stock time series forecasting.

In this study, we address the two challenges by proposing a general stock time series forecasting framework, **Hermes**, which integrates *hyperedge-based moving aggregation* and *hyperedge-based multi-scale fusion* modules into a spatial-temporal hypergraph network. These integrations enhance the precision and stability of stock forecasting. First, to more flexibly model lead-lag relationships among industries, we integrate a *hyperedge-based moving aggregation module* into the hypergraph network. This module incorporates a sliding window and utilizes dynamic temporal aggregation operations to consider lead-lag dependencies among industries. This approach aims to capture a market's complex structure more precisely. Second, to address the issue that existing methods fail to effectively model multi-scale information, we decompose raw data at multiple scales and integrate a *multi-scale fusion module* into the hypergraph network. This module employs cross-scale,

[2] https://finance.yahoo.com/news/american-electric-power-aep-among-121752028.html?

edge-to-edge message passing to integrate information from different scales while preserving the consistency of each scale, thereby boosting both the accuracy and stability of forecasts. Experimental results on multiple real-world stock datasets show that Hermes outperforms existing state-of-the-art methods.

Our contributions are summarized as follows.

- To solve the STSF problem, we propose a general spatial-temporal hypergraph network framework, **Hermes**, which learns more accurate and adaptive forecasting models by considering lead-lag and multi-scale industry correlations.
- We propose a hyperedge-based moving aggregation module that introduces a sliding window and utilizes dynamic temporal aggregation operations to consider lead-lag dependencies among industries.
- We design a hyperedge-based multi-scale fusion module. This module utilizes a cross-scale, edge-to-edge message-passing to integrate information from different scales while maintaining the consistency of each scale.
- We report on extensive experiments on public stock datasets, finding that Hermes is capable of outperforming state-of-the-art baselines.

2 Related Work

2.1 Deep Learning Stock Time Series Forecasting

With the rise of deep learning, notable progress has occurred in stock forecasting. This is mainly because deep neural networks can model complex nonlinear patterns Wu et al. (2025a); Li et al. (2026). Some studies use recurrent neural networks (RNNs) Nelson et al. (2017); Qin et al. (2017); Akita et al. (2016); Rahman et al. (2019) to capture the historical development in individual stock prices and predict their short-term trends. In addition, due to the significant progress of Multi-Layer Perceptrons (MLPs) architecture in the general time series field Zeng et al. (2023); Lin et al. (2025, 2024). Increasingly many proposals target stock forecasting performance by improving the MLPs architecture. For example, a series of studies Fan and Shen (2024); Lazcano et al. (2024); Tashakkori et al. (2024) enhance the learning ability of simple MLPs structures by utilizing the MLP-Mixer backbone, thereby improving forecasting accuracy. Next, the Transformer model has become increasingly popular due to the ability of its self-attention mechanism to capture long-term dependencies. For example, Master Li et al. (2024) proposes a novel stock transformer for stock price forecasting to effectively capture stock correlation. While these deep learning methods have advanced stock forecasting, the complexity of stock data and the need to account for intricate relationships among stocks have led researchers to explore new methods based on hypergraphs.

2.2 Hypergraph-based Stock Time Series Forecasting

Traditional graphs have limitations when it comes to modeling higher-order multivariate relationships, as they can only represent relations between pairs of nodes. To address this issue, Hypergraph Neural Networks (HGNNs) Feng et al. (2019a) were introduced. Hypergraphs extend the concept of simple graphs to enable the capture of relationships among multiple stocks Chen et al. (2020). Hypergraphs have gained increasing attention and application across various fields. Hyperedge in a hypergraph represents a set of vertices, making them suitable for modeling non-pairwise relationships Luo et al. (2014); Sawhney et al. (2020); Huynh et al. (2023). For example, HGAM Li et al. (2022) is a hypergraph-based reinforcement learning method for stock portfolio selection. THGNN Xiang et al. (2022) is a temporal and heterogeneous graph neural network model that aims to learn dynamic relationships by using two-stage attention mechanisms. STHAN-SR Sawhney et al. (2021) enhances corporate relevance based on Wiki data and uses hypergraph convolution to propagate information from higher-order neighbors. ESTIMATE Huynh et al. (2023), employs hypergraphs to capture non-pairwise correlations, utilizing temporal generative filters to identify individual patterns for each stock. Unlike the above methods, we propose a new hypergraph-based stock time series forecasting model that specifically takes into account the multi-scale nature of stock data, enabling it to capture the complexity of market dynamics more comprehensively. Furthermore, our approach places particular emphasis on the lead-lag relationships between financial industries, where the changes in one industry

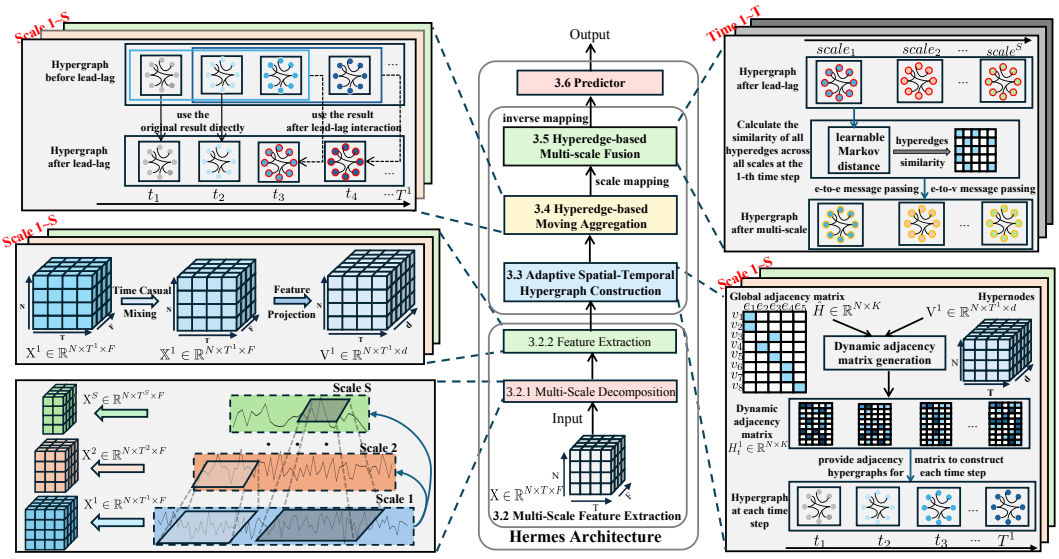


Figure 2: The architecture of Hermes.

may precede or lag behind those in another. By capturing these key relationships more fully, we aim to improve forecasting accuracy.

3 Methodology

In financial markets, a stock time series $\mathbf{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$ records the historical data of N stocks, where $\boldsymbol{X}_i \in \mathbb{R}^{T \times F}$ is data of one stock with a lookback window of T time steps and with F technical indicators at each time step. In stock time series, the indicators may include opening price, closing price, highest price, lowest price, and trading volume. Following existing studies Huynh et al. (2023); Fan and Shen (2024), we input a stock time series with multiple indicators, denoted as $\mathbf{X} \in \mathbb{R}^{N \times T \times F}$, and aim to predict the closing price p_i^t at time step t. We define the 1-day return ratio as $r_i^t = \frac{p_i^t - p_i^{t-1}}{p_i^{t-1}}$. With θ representing the model parameters, the process can be expressed as follows.

$$\mathbf{X} \in \mathbb{R}^{N \times T \times F} \xrightarrow{\theta} p \in \mathbb{R}^{N \times 1} \rightarrow r \in \mathbb{R}^{N \times 1}. \tag{1}$$

3.1 Framework Overview

In this section, we introduce the framework of our proposed spatiotemporal hypergraph neural network that considers multiple time scales and models the lead-lag relationships between industries—see Figure 2. Our framework takes historical financial sequences as input, first samples the time series into several sequences of different granularities according to different time scales, and maps the feature dimension to the hidden dimension to consider the high-dimensional correlations between time series segments under different scales. Subsequently, we consider constructing relationships between stocks at each scale, specifically considering constructing hyperedges for stocks in the same industry, and modeling the lead-lag relationships between industries in the corresponding hypergraphs. Then, we model the interactions between different scales, construct hyperedges for industries of different scales, and adaptively learn the correlations between them under the corresponding multi-scale hypergraph, thereby playing a role in fusing scales. Finally, we predict the corresponding indicators for each stock for the next day through a prediction head based on a multilayer perceptron.

3.2 Multi-Scale Feature Extraction

Multi-Scale Decomposition. Time series of different scales exhibit different characteristics, among which fine scales primarily describe detailed patterns and contain more periodic information, while coarse scales highlight macro changes and cover the overall trend changes. This multi-scale perspective can essentially unravel the complex variations within multiple components, thereby facilitating

time variation modeling. It is worth noting that, especially for prediction tasks, due to their different dominant time patterns, multi-scale time series show different predictive capabilities.

Specifically, for the financial input time series $\mathbf{X} \in \mathbb{R}^{N \times T \times F}$, which denotes the records of F key indicators for N stocks of T timestamps in the past. To decouple the intricate multi-resolution information, we use convolutional layers of different sizes to adaptively down-sample the time series into sequences of different scales:

$$\mathbf{X}^i = \text{1D-Conv}^i(\mathbf{X}), \tag{2}$$

where $\mathbf{X}^i \in \mathbb{R}^{N \times T^i \times F}$ denotes the i-th scale of financial time series, T^i is the down-sampled length. We then obtain the multi-scale time series sequences $\{\mathbf{X}^1, \mathbf{X}^2, \cdots, \mathbf{X}^S\}$ of S kinds of scales through S convolutional layers with different kernel sizes and stride lengths.

Feature Extraction. Subsequently, for time series of different scales, we enhance their representations at two aspects. First, we adopt the Causal-Mixing technique to process the temporal dimension. The Causal-Mixing technique ensures causal relationships rather than full connectivity, and ensure that time series points can only see previous points. It consists of a group of MLP layers:

$$\text{Causal-Mixing}^i = \{\text{MLP}^{i,1}, \text{MLP}^{i,2}, \cdots, \text{MLP}^{i,T^i}\}, \tag{3}$$

$$\text{MLP}^{i,j} = \mathbb{R}^j \to \mathbb{R}, \mathbb{X}^i = \text{Causal-Mixing}^i(\mathbf{X}^i). \tag{4}$$

The Causal-Mixing is applied along the temporal dimension, which maps $X^i_{1:j}$ through $\text{MLP}^{i,j}$ to $\mathbb{X}^i_j$. The new representation $\mathbb{X} \in \mathbb{R}^{N \times T^i \times F}$ keeps the original shape.

Then their feature dimensions are mapped into the high-dimension hidden spaces to enrich their representation capabilities, facilitating the extraction of more complex hidden relationships:

$$\mathcal{X}^i = \mathbb{X}^i \cdot W^i, \tag{5}$$

where $\mathcal{X}^i \in \mathbb{R}^{N \times T^i \times d}$, d is the dimension of the latent space. $W^i \in \mathbb{R}^{F \times d}$, constructing a linear projection to enhance representational capability.

3.3 Adaptive Spatial-Temporal Hypergraph Construction

For the representation $\mathcal{X}^i \in \mathbb{R}^{N \times T^i \times d}$ of i-th scale, the hypergraph of i-th scale $\mathcal{X}^i$ can be constructed through a global-shared matrix $H \in \mathbb{R}^{N \times K}$, where N represents the number of stocks, K represents the number of industries, and $H_{l,m} = 1$ denotes that the l-th stock belongs to the m-th industry. The i-th Spatial-Temporal hypergraph $\mathcal{G}^i$ is formulated as:

$$\mathcal{G}^i = (H, \mathcal{X}^i, E^i), \tag{6}$$

where $\mathcal{X}^i$ is the representation of i-th scale and works as the nodes of the $\mathcal{G}^i$, $E^i \in \mathbb{R}^{K \times T^i \times d}$ is the hyperedge constructed through the stock-industry relationship matrix H:

$$\text{scores}^i = \text{Linear}^i(\mathcal{X}^i) \in \mathbb{R}^{N \times 1}, \text{Linear}^i = \mathbb{R}^{T^i \times d} \to \mathbb{R}, \tag{7}$$

$$H^i = H \odot \text{scores}^i + (1 - H) \odot (-\infty), \tag{8}$$

$$\mathcal{H}^i = \text{SoftMax}(H^i, \text{dim} = 0), E^i = (\mathcal{H}^i)^T \cdot \mathcal{X}^i, \tag{9}$$

where the matrix H is first processed with a learnable renovation to adapt to the current representation $\mathcal{X}^i$. H^i is the intermediate result obtained through endowing the $H_{l,m} = 1$ parts with learnable scores^i and $H_{l,m} = 0$ parts with $-\infty$ to ensure zeros in the SoftMax step. Then the $\mathcal{H}^i \in \mathbb{R}^{N \times K}$ with Continuous Weights is obtained, and the hyperedges $E^i \in \mathbb{R}^{K \times T^i \times d}$ of industries are generated through information integration from all the related stocks. We obtain all the Spatial-Temporal hypergraphs $\{\mathcal{G}^1, \mathcal{G}^2, \cdots \mathcal{G}^S\}$ for each scale as above mentioned.

3.4 Hyperedge-based Moving Aggregation

After delineating different time series scales with Spatial-Temporal hypergraphs $\{\mathcal{G}^1, \mathcal{G}^2, \cdots \mathcal{G}^S\}$, we further consider the lead-lag relationships between industries under each scale to analyze the complex correlations in financial markets—see Figure 5 in Appendix E.

We construct a message passing mechanism on the hyperedges to capture the lead-lag relationships between industries. First, we employ a fixed-size sliding window as the range for examining the lead-lag relationships. For each hyperedge, through the sliding window, we can delineate different patches, which serve as the smallest units for constructing the lead-lag relationships:

$$\mathcal{P}^i = \text{SlidingWindow}(E^i, \text{size} = k^i, \text{stride} = 1), \tag{10}$$

where $\mathcal{P}^i \in \mathbb{R}^{K\times(T^i-k^i+1)\times k^i\times d}$ denotes the patches of E^i, containing local information with the size of k^i. From the perspective of causality, considering the relationship between the entire window and its last point, we can construct the relationship matrix of the leading part and the lagging part. We then mine the lead-lag relationships inside each patch by considering the correlations between $\mathcal{P}^i_{before}$ and $\mathcal{P}^i_{last}$:

$$\mathcal{P}^i_{before} = \text{Reshape}(\mathcal{P})^i \in \mathbb{R}^{(T^i-k^i+1)\times[K\times k^i]\times d}, \tag{11}$$

$$\mathcal{P}^i_{last} = \text{Reshape}(\mathcal{P})^i_{-1} \in \mathbb{R}^{(T^i-k^i+1)\times[K\times 1]\times d}. \tag{12}$$

Subsequently, by using the message passing method to propagate through the leading part, we obtain the feature fusion among hyperedges:

$$\tilde{A}^i = \mathcal{P}^i_{before} \otimes \mathcal{P}^i_{last} \in \mathbb{R}^{(T^i-k^i+1)\times[K\times k^i]\times K}, A^i = SoftMax(\tilde{A}^i, \text{dim} = 1), \tag{13}$$

$$\tilde{E}^i = A^i \otimes \mathcal{P}^i_{before} \in \mathbb{R}^{K\times(T^i-k^i+1)\times d}, \hat{E}^i \in \mathbb{R}^{K\times T^i\times d} = \begin{cases} E^i, & t < k^i \\ \tilde{E}^i, & k^i \le t \le T^i \end{cases}, \tag{14}$$

where $\tilde{A}^i$ considers the correlations among the leading parts and lagging parts, A^i probabilityizes $\tilde{A}^i$ to control weightsuming, $\hat{E}^i \in \mathbb{R}^{K\times T^i\times d}$ denotes the embedding of hyperedges integrated with lead-lag relationships among different industries. It is particularly important to note that when $t < k^i$, no transformation is performed; when $k^i \le t \le T^i$, the result incorporating lead-lag interactions is used as the new hyperedges.

This is the method of considering the leading and lagging relationships between industries under each scale using Spatial-Temporal hypergraphs. After performing the above operation for each scale, we obtain a multi-scale representation of hyperedges $\{\hat{E}^1, \hat{E}^2, \cdots, \hat{E}^S\}$. Subsequently, fusion between scales will be carried out to integrate information of different granularities.

3.5 Hyperedge-based Multi-scale Fusion

For multi-scale representation of hyperedges $\{\hat{E}^1, \hat{E}^2, \cdots, \hat{E}^S\}$ which contains the industry information, we further extract the potential correlations among different scales to reveal the latent interactions among different industries. First, we up-sample different scales back to the original length to ensure uniform representation size:

$$\hat{\mathcal{E}}^i = \text{Linear}^i(\hat{E}^i) \in \mathbb{R}^{K\times T\times d}, \text{Linear}^i = \mathbb{R}^{T^i} \to \mathbb{R}^T, \tag{15}$$

$$\hat{\mathcal{E}} = \{\hat{\mathcal{E}}^1, \hat{\mathcal{E}}^2, \cdots, \hat{\mathcal{E}}^S\} \in \mathbb{R}^{S\times K\times T\times d}. \tag{16}$$

Then, we construct adaptive adjacency matrices for these hyperedges of different scales to measure the correlations between different industries at various information granularities. We employ metric learning for this task and use a learnable Mahalanobis distance Goldberger et al. (2004); Qiu et al. (2025b) to automatically build relationships between different industries in the hidden space. Since the operations across the time dimension T are independent and can be executed in parallel, we omit this dimension in the following formulations for notational simplicity.

$$\tilde{\mathcal{E}} = \text{Flatten}(\hat{\mathcal{E}}) \in \mathbb{R}^{(S\times K)\times d}, D_{ij} = (\tilde{\mathcal{E}}_i - \tilde{\mathcal{E}}_j) \cdot Q \cdot (\tilde{\mathcal{E}}_i - \tilde{\mathcal{E}}_j)^T, \tag{17}$$

$$C_{ij} = \begin{cases} \frac{1}{D_{ij}} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}, \quad B = \text{SoftMax}(C, \text{dim} = 0), \tag{18}$$

where $D, C, B \in \mathbb{R}^{(S\times K)\times(S\times K)}$. $Q \in \mathbb{R}^{d\times d}$ is a learnable semi-positive definite matrix, and can be practically constructed by $Q = A^T \cdot A$, where A is also a learnable matrix. Subsequently, we

construct interactions between hyperedges at different scales using message passing and further update the representation of hyperedges.

$$\mathcal{E}' = B \cdot \tilde{\mathcal{E}} \cdot W^e \in \mathbb{R}^{(S\times K)\times d}, \tag{19}$$

where $W^e \in \mathbb{R}^{d\times d}$ is the learnable parameters in the graph message passing mechanism. After integrating the lead-lag relationships between industries on the hyperedges and fusing the hidden relationships between different scales, the representation of the hyperedges $\mathcal{E}'$ is adaptively updated. We then continue to complete the message passing from hyperedges to nodes—see Figure 6 in Appendix E, thereby feeding back the complex correlations within the industry to specific stocks and completing the update of the representation of stock nodes. Finally, we restore the omitted T dimension:

$$\mathcal{V} = \mathcal{E}' \otimes \mathcal{H} \in \mathbb{R}^{(S\times N)\times T\times d}. \tag{20}$$

3.6 Predictor

After completing the complex representation extraction using a hypergraph-based structure, we employ an MLP-based predictor and incorporate residual connection modules to forecast financial indicators across each scale i.

$$\hat{\mathcal{X}}^i = \text{MLP}_1^i(\mathcal{X}^i) \in \mathbb{R}^{N\times 1}, \tag{21}$$

$$\tilde{\mathcal{V}}^i = \text{MLP}_2^i(\mathcal{V}^i) \in \mathbb{R}^{N\times 1}, \tag{22}$$

$$\hat{Y}^i = \hat{\mathcal{X}}^i + \tilde{\mathcal{V}}^i, \;\; \hat{Y} = \sum_{i=1}^{S} \hat{Y}^i, \tag{23}$$

where $\mathcal{X}^i \in \mathbb{R}^{(N\times T^i\times d)}$ represents the initial node embedding in Equation 5 that does not consider the lead-lag relationship and multi-scale interaction effects. However, $\mathcal{V}^i \in \mathbb{R}^{(N\times T\times d)}$ is the spatio-temporal hypernode embedding in Equation 20 that incorporates lead-lag correlation and multi-scale information. Finally, we combine the prediction results at different scales to obtain the final prediction result.

3.7 Optimization Objectives

Following StockMixer Fan and Shen (2024), We use the 1-day return ratio of a stock as the ground-truth rather than the normalized price used in previous work using a combination of a pointwise regression and pairwise ranking-aware loss to minimize the MSE between the predicted and actual return ratios while maintaining the relative order of top ranked stocks with higher expected return for investment as:

$$L = L_{\text{MSE}} + \alpha \sum_{i=1}^{N}\sum_{j=1}^{N} \max(0, -(\hat{r}_i^t - \hat{r}_j^t)(r_i^t - r_j^t)), \tag{24}$$

where $\hat{r}^t$ and r^t are the predicted and actual ranking scores, respectively, and α is a weighting parameter.

4 Experiments

4.1 Experimental Setup

Datasets and Baselines. We evaluate our approach using three real-world datasets from the US stock market. These datasets contain relatively complete stock-industry relationships or Wiki-based company relations. Besides, we conduct a comparative analysis between our proposed Hermes and a range of baseline methods, which include prominent approaches in both machine learning and deep learning methods for STSF. Given that existing research has demonstrated the limited effectiveness of traditional statistical methods Fan and Shen (2024); Li et al. (2024), we exclude them from our comparative evaluation to focus on more competitive approaches.

Metrics. To evaluate model performance, we employ three widely used metrics: IC and RIC, which are correlation-based evaluation metrics, and SR, which is a risk-adjusted return metric.

Table 1: Comparison results on public stock market datasets (measured by t-test with p-value < 0.01). The methods for comparison are mainly divided into five types: HGNN (HyperGraph Neural Network), Transformer, MLP (Multi-Layer Perceptron), GNN (Graph Neural Network), and RNN (Recurrent Neural Network). Bold indicates the best (SOTA) results.

Models		HGNN				Transformer			MLP			GNN			RNN	
		Hermes	EST.	STHAN.	ASHyper	iTrans.	PatchTST	DUET	SMixer.	DLin.	TMixer.	RSR-I	GAT	RGCN	ALSTM	LSTM
NASDAQ	IC	**0.044**	0.037	0.039	0.041	0.038	0.039	0.039	0.043	0.031	0.041	0.038	0.035	0.034	0.035	0.032
	RIC	**0.538**	0.444	0.451	0.508	0.420	0.496	0.494	0.501	0.453	0.532	0.398	0.377	0.382	0.371	0.354
	SR	**2.161**	1.307	1.416	1.157	1.877	1.261	1.751	1.465	1.778	1.177	1.238	1.233	1.054	0.941	0.892
NYSE	IC	**0.032**	0.030	0.029	0.025	0.023	0.029	0.024	0.029	0.026	0.031	0.026	0.025	0.025	0.023	0.024
	RIC	**0.466**	0.327	0.344	0.250	0.303	0.284	0.327	0.351	0.277	0.314	0.284	0.297	0.275	0.276	0.256
	SR	**1.655**	1.115	1.228	1.208	1.520	0.898	1.361	1.454	1.526	1.556	1.098	1.070	0.932	0.764	0.857
S&P500	IC	**0.050**	0.035	0.037	0.046	0.048	0.046	0.049	0.041	0.043	0.040	0.033	0.034	0.028	0.029	0.031
	RIC	**0.334**	0.241	0.227	0.268	0.242	0.249	0.275	0.262	0.214	0.253	0.200	0.191	0.175	0.181	0.186
	SR	**2.247**	1.547	1.533	2.148	1.326	1.402	1.982	1.586	1.557	2.170	1.437	1.484	1.359	1.298	1.332

Model abbreviations: EST. (ESTIMATE), STHAN. (STHAN-SR), iTrans. (iTransformer), SMX (StockMixer), DLin. (DLinear), and TMX (TimeMixer).

Table 2: Ablation study results on public datasets NASDAQ, NYSE, and S&P500.

Ablation	NASDAQ			NYSE			S&P500		
Model Component	IC	ICIR	SR	IC	ICIR	SR	IC	ICIR	SR
w/o Multi-Scale Fusion	0.038	0.464	1.374	0.028	0.421	1.354	0.031	0.248	1.517
w/o Total Multi-Scale	0.030	0.422	0.975	0.023	0.347	1.199	0.032	0.213	1.075
w/o Lead-Lag	0.039	0.472	1.451	0.028	0.420	1.225	0.037	0.201	1.628
w/o Skip-Connection	0.041	0.441	1.655	0.029	0.345	1.449	0.043	0.252	1.653
Only Initial Node Embedding	0.019	0.174	0.695	0.016	0.113	0.449	0.018	0.122	0.659
Hermes (Ours)	**0.044**	**0.538**	**2.161**	**0.032**	**0.466**	**1.655**	**0.050**	**0.334**	**2.247**

Implementation Details. For fair comparison, all samples are generated by moving a 16 time steps lookback window. All experiments with Hermes are conducted using PyTorch Paszke et al. (2019) in Python 3.9, and are executed on a server featuring an Intel(R) Xeon(R) Platinum 8358 CPU and an NVIDIA Tesla-A800 GPU. We use ADAM Kingma and Ba (2015) with an initial learning rate of 5e-3. We do not apply the "Drop Last" trick Qiu et al. (2024, 2025c) to ensure a fair comparison. Detailed descriptions of the experimental datasets, baselines, and evaluation metrics are provided in the Appendix A.1.

4.2 Overall Performance

We conduct a comprehensive comparison of Hermes against five categories of baseline methods (HGNN, Transformer, MLP, GNN, and RNN) across three representative stock market datasets: NASDAQ, NYSE, and S&P 500. As demonstrated in Table 1. We have the following observations: 1) Hermes consistently achieves State-of-the-Art (SOTA) performance across all datasets and evaluation metrics. 2) For univariate methods, whether it is LSTM or ALSTM, their performance is inferior to that of other hybrid architectures. This strongly highlights the necessity and importance of capturing the complex industry and stock relationships in financial markets. 3) For GNNs algorithms such as RGCN, GAT, and RSR-I, their performance shows a significant improvement over RNNs, indicating that GNNs have strong stock relationship modeling capabilities. However, compared to HGNNs algorithms like STHAN-SR, ESTIMATE, and ASHyper, GNNs perform slightly worse. This suggests that in the context of the financial market industry, using hypergraph models to unify the relationships of stocks within the industry plays a crucial role in enhancing forecasting performance. 4) Compared to other hypergraph algorithms that consider stock relationships within industries (such as STHAN-SR, ESTIMATE, and ASHyper), the Hermes algorithm we propose shows significant performance improvement. This clearly demonstrates that, based on the consideration of stock relationships within industries, incorporating inter-industry lead-lag relationships and multi-scale relationships plays a crucial role in enhancing forecasting performance.

4.3 Ablation Studies

Table 2 illustrates the unique impact of each module. We have the following observations: 1) When the **multi-scale fusion module** in Section 3.5 and the **total multi-scale module** (Remove the Total Multi-Scale module, including Decomposition in Section 3.2 and Fusion in Section 3.5,

Table 3: Efficiency analysis on public stock datasets, concerning the training time, inference time (measured in milliseconds per batch under the condition of a batch size being 1), and memory usage (quantified in GB) of different HGNN-based (HyperGraph Neural Network) models. Model abbreviations: EST. (ESTIMATE) and STHAN. (STHAN-SR).

Dataset	NASDAQ				NYSE				S&P500			
Property	STHAN.	EST.	ASHyper	Hermes	STHAN.	EST.	ASHyper	Hermes	STHAN.	EST.	ASHyper	Hermes
Training Time	257 ms	1,267 ms	288 ms	215 ms	384 ms	1,462 ms	378 ms	282 ms	297 ms	1,317 ms	314 ms	211 ms
Inference Time	52 ms	163 ms	58 ms	42 ms	73 ms	241 ms	71 ms	61 ms	32 ms	109 ms	36 ms	25 ms
Memory Usage	2.15 GB	9.62 GB	1.99 GB	0.98 GB	2.82 GB	10.25 GB	2.17 GB	1.14 GB	1.23 GB	4.52 GB	1.51 GB	0.54 GB

respectively) are removed, the performance of the model decreases in both cases. This underscores the effectiveness of our multi-scale approach, confirming that both multi-scale decomposition and fusion are essential to the model's predictive power. 2) The ablation of the **lead–lag module** in Section 3.4 causes a substantial decline in performance, demonstrating that explicitly modeling inter-industry lead–lag relationships is essential for strong predictive accuracy. 3) Removing the **skip-connection module** in Section 3.6 also reduces performance, confirming its crucial role in facilitating information transmission and gradient flow, thereby improving the model's feature extraction capabilities. 4) When the Predictor relies **solely on the initial node embedding** from Equation 21 without integrating the updated features from Equation 22, the model performance undergoes a significant degradation. This is expected given the inherent complexity of stock forecasting, where primitive encodings are insufficient to capture intricate market dynamics. Such a sharp decline underscores the necessity and effectiveness of our proposed hypergraph framework.

4.4 Efficiency Analysis

In order to ensure good applicability of the model in practical applications, the runtime and memory usage of the model are crucial. We conducted an efficiency comparison analysis of Hermes and other hypergraph-based methods, STHAN-SR, ESTIMATE, and ASHyper—see Table 3.

Runtime Analysis: In practical application scenarios, fast response is one of the key factors. For example, in the field of financial market forecasting, timely obtaining forecasting results can provide valuable decision-making basis for investors. If the model takes too long to operate, even if the prediction accuracy is high, it will be difficult to meet the actual needs. Therefore, we evaluate the computational speed by recording the time each model takes to train and infer a sample (that is, the batch size is uniformly set to 1) with the same dataset. The results show that the Hermes model performs excellently. Compared with the other two hypergraph-based methods, its computational time is reduced. This is mainly attributed to the fact that some modules of the Hermes model adopt an efficient MLP structure in the algorithm design process. This structure can significantly improve the computational efficiency while ensuring the accuracy of forecasting results.

Memory Analysis: Memory usage is also an important factor to measure the efficiency of the model. Excessive memory usage will not only limit the application range of the model in different hardware environments but also may cause the system to run slowly or even crash. In this experiment, we monitored the memory usage of each model in detail during operation. The results show that the Hermes has a lower memory usage rate. This advantage primarily stems from its unique architectural design, which enables superior performance without requiring large hidden dimensions, as further validated in Section F through sensitivity analysis of the Latent Space Dimension parameter. Consequently, Hermes effectively reduces memory usage while maintaining high performance, enhancing both the practicality and deployment flexibility of the model.

4.5 Portfolio Performance Evaluation

We evaluate the portfolio performance of model predictions using a unified Top-5 equal-weighted long-only strategy on the NASDAQ test set. On each test day, valid stocks are ranked by their model-predicted scores, and the five highest-ranked stocks are selected, each receiving a portfolio weight of 20%. The daily portfolio return is the average realized return of the selected stocks, and cumulative returns are obtained by compounding daily portfolio returns. All models use the same evaluation period, realized returns, and validity masks to ensure a consistent comparison. The test period covers 237 trading days, with initial portfolio wealth normalized to one. Returns are reported

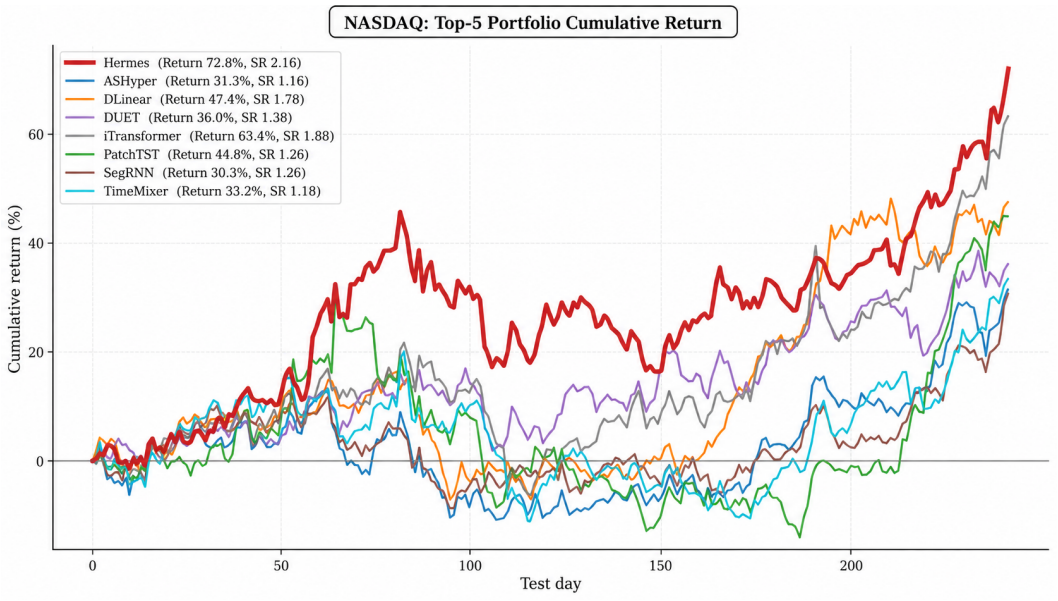


Figure 3: Cumulative returns of Top-5 equal-weighted long-only portfolios on the NASDAQ test set. Each portfolio selects the five valid stocks with the highest model-predicted scores on each test day and assigns them equal weights. Returns are compounded daily and reported before transaction costs. The legend reports the terminal cumulative return over 237 trading days and the annualized Sharpe ratio (SR). Hermes is highlighted by the thick red curve.

before transaction costs. We also report the annualized Sharpe ratio to assess returns relative to their volatility, assuming a zero risk-free rate and 252 trading days per year.

As shown in Figure 3, Hermes achieves the highest terminal cumulative return of 72.79% on the NASDAQ test set, outperforming the strongest baseline, iTransformer (63.40%), by 9.39 percentage points. Hermes also achieves the highest annualized Sharpe ratio of 2.16, compared with 1.88 for iTransformer, indicating that its return advantage is accompanied by stronger risk-adjusted performance. The cumulative return trajectories show that Hermes gradually establishes an advantage during the earlier part of the test period and maintains a relatively high cumulative return through the subsequent declines. Although DLinear temporarily takes the lead during part of the later period, Hermes regains its advantage and finishes with the highest cumulative return. Its advantage therefore reflects cumulative-return leadership on most test days and superior terminal performance, rather than uninterrupted dominance. These results support the stock-selection value of Hermes predictions under the evaluated test period and Top-5 portfolio strategy.

5 Conclusions

Stock time series forecasting plays a pivotal role in the modern economy, serving as a key tool for investors, financial institutions, and policymakers to make accurate decisions. However, existing forecasting methods often fall short in fully considering the lead-lag between industries and multi-scale information, resulting in limited predictive performance. To address these challenges, this study introduces the Hermes framework. By ingeniously fusing multi-scale analysis with lead-lag modules within a hypergraph network, the Hermes framework significantly boosts the performance of stock forecasting. Specifically, the hyperedge-based moving aggregation module incorporates a sliding window and utilizes dynamic temporal aggregation operations to consider lead-lag dependencies among industries. Meanwhile, the hyperedge-based multi-scale fusion module decomposes raw data at multiple scales, and then employs cross-scale, edge-to-edge message passing to integrate information from different scales while preserving the consistency of each scale.

Acknowledgments and Disclosure of Funding

This work was partially supported by National Natural Science Foundation of China (62472174), the ECNU Multifunctional Platform for Innovation (001), and the Fundamental Research Funds for the Central Universities (YBNLTS2026010). Jilin Hu is the corresponding author of the work.

References

Jinyong Fan and Yanyan Shen. Stockmixer: A simple yet strong mlp-based architecture for stock price forecasting. In *AAAI*, pages 8389–8397, 2024.

Tong Li, Zhaoyang Liu, Yanyan Shen, Xue Wang, Haokun Chen, and Sen Huang. Master: Market-guided stock transformer for stock price forecasting. In *AAAI*, pages 162–170, 2024.

Akhter Mohiuddin Rather, VN Sastry, and Arun Agarwal. Stock market prediction and portfolio selection models: a survey. *Opsearch*, 54:558–579, 2017.

Christos Faloutsos, Jan Gasthaus, Tim Januschowski, and Yuyang Wang. Forecasting big time series: old and new. In *Proc. VLDB Endow.*, pages 2102–2105, 2018.

Kaippallimalil J Jacob and Dennis Shasha. Fintime: a financial time series benchmark. *ACM SIGMOD Record*, 28(4):42–48, 1999.

Zetao Zheng, Jie Shao, Jia Zhu, and Heng Tao Shen. Relational temporal graph convolutional networks for ranking-based stock prediction. In *ICDE*, pages 123–136, 2023.

Francis X Diebold and Kamil Yılmaz. On the network topology of variance decompositions: Measuring the connectedness of financial firms. *Journal of econometrics*, 182(1):119–134, 2014.

Ashadun Nobi, Seong Eun Maeng, Gyeong Gyun Ha, and Jae Woo Lee. Effects of global financial crisis on network structure in a local stock market. *Physica A: Statistical Mechanics and its Applications*, 407:135–143, 2014.

Xiangfei Qiu, Xingjian Wu, Hanyin Cheng, Xvyuan Liu, Chenjuan Guo, Jilin Hu, and Bin Yang. DBLoss: Decomposition-based loss function for time series forecasting. In *NeurIPS*, 2025a.

Xiangfei Qiu, Yuhan Zhu, Zhengyu Li, Xingjian Wu, Bin Yang, and Jilin Hu. DAG: A dual correlation network for time series forecasting with exogenous variables. In *ICML*, 2026a.

Md Shadman Abid, Razzaqul Ahshan, Mohammed Al-Abri, and Rashid Al Abri. Robust deep learning model with attention framework for spatiotemporal forecasting of solar and wind energy production. *Energy Conversion and Management: X*, page 100919, 2025.

Omowonuola Ireoluwapo Kehinde Olanrewaju, Gideon Oluseyi Daramola, and Darlington Eze Ekechukwu. Strategic financial decision-making in sustainable energy investments: Leveraging big data for maximum impact. *World Journal of Advanced Research and Reviews*, 22(3):564–573, 2024.

S Saba Rafiei, Mahdi S Naderi, and Mehrdad Abedi. A comprehensive energy management application method considering smart home occupant behavior using iot and real big data. *arXiv preprint arXiv:2502.06052*, 2025.

Xingjian Wu, Xiangfei Qiu, Hongfan Gao, Jilin Hu, Bin Yang, and Chenjuan Guo. K^2VAE: A koopman-kalman enhanced variational autoencoder for probabilistic time series forecasting. In *ICML*, 2025a.

Zhengyu Li, Xiangfei Qiu, Yuhan Zhu, Xingjian Wu, Jilin Hu, Chenjuan Guo, and Bin Yang. GCGNet: Graph-consistent generative network for time series forecasting with exogenous variables. In *ICLR*, 2026.

David MQ Nelson, Adriano CM Pereira, and Renato A De Oliveira. Stock market's price movement prediction with lstm neural networks. In *IJCNN*, pages 1419–1426, 2017.

Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*, 2017.

Ryo Akita, Akira Yoshihara, Takashi Matsubara, and Kuniaki Uehara. Deep learning for stock prediction using numerical and textual information. In *ICIS*, pages 1–6, 2016.

Mohammad Obaidur Rahman, Md Sabir Hossain, Ta-Seen Junaid, Md Shafiul Alam Forhad, and Muhammad Kamal Hossen. Predicting prices of stock market using gated recurrent units (grus) neural networks. *International Journal of Computer Science and Network Security*, 19(1):213–222, 2019.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI*, pages 11121–11128, 2023.

Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. Cyclenet: enhancing time series forecasting through modeling periodic patterns. pages 106315–106345, 2025.

Shengsheng Lin, Weiwei Lin, Wentai Wu, Haojun Chen, and Junjie Yang. Sparsetsf: Modeling long-term time series forecasting with *1k* parameters. In *ICML*, pages 30211–30226, 2024.

Ana Lazcano, Miguel A Jaramillo-Morán, and Julio E Sandubete. Back to basics: The power of the multilayer perceptron in financial time series forecasting. *Mathematics*, 12(12):1920, 2024.

Arash Tashakkori, Mohammad Talebzadeh, Fatemeh Salboukh, and Lochan Deshmukh. Forecasting gold prices with mlp neural networks: A machine learning approach. *IJSEA*, pages 13–20, 2024.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *AAAI*, pages 3558–3565, 2019a.

Chaofan Chen, Zelei Cheng, Zuotian Li, and Manyi Wang. Hypergraph attention networks. In *TrustCom*, pages 1560–1565, 2020.

Yongen Luo, Jicheng Hu, Xiaofeng Wei, Dongjian Fang, and Heng Shao. Stock trends prediction based on hypergraph modeling clustering algorithm. In *PIC*, pages 27–31, 2014.

Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, and Rajiv Ratn Shah. Spatiotemporal hypergraph convolution network for stock movement forecasting. In *ICDM*, pages 482–491, 2020.

Thanh Trung Huynh, Minh Hieu Nguyen, Thanh Tam Nguyen, Phi Le Nguyen, Matthias Weidlich, Quoc Viet Hung Nguyen, and Karl Aberer. Efficient integration of multi-order dynamics and internal dynamics in stock movement prediction. In *WSDM*, pages 850–858, 2023.

Xiaojie Li, Chaoran Cui, Donglin Cao, Juan Du, and Chunyun Zhang. Hypergraph-based reinforcement learning for stock portfolio selection. In *ICASSP*, pages 4028–4032, 2022.

Sheng Xiang, Dawei Cheng, Chencheng Shang, Ying Zhang, and Yuqi Liang. Temporal and heterogeneous graph neural network for financial time series prediction. In *CIKM*, pages 3584–3593, 2022.

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 *AAAI*, pages 497–504, 2021.

Jacob Goldberger, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov. Neighbourhood components analysis. In *NeurIPS*, volume 17, 2004.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. DUET: Dual clustering enhanced multivariate time series forecasting. In *SIGKDD*, pages 1185–1196, 2025b.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. TFB: Towards comprehensive and fair benchmarking of time series forecasting methods. In *Proc. VLDB Endow.*, pages 2363–2377, 2024.

Xiangfei Qiu, Zhe Li, Wanghui Qiu, Shiyan Hu, Lekui Zhou, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Aoying Zhou, Zhenli Sheng, Jilin Hu, Christian S. Jensen, and Bin Yang. TAB: Unified benchmarking of time series anomaly detection methods. In *Proc. VLDB Endow.*, pages 2775–2789, 2025c.

Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. Temporal relational ranking for stock prediction. *TOIS*, 37(2):1–30, 2019b.

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

Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. Enhancing stock movement prediction with adversarial training. In *IJCAI*, pages 5843–5849, 2019c.

Wei Li, Ruihan Bao, Keiko Harimoto, Deli Chen, Jingjing Xu, and Qi Su. Modeling the stock relation with graph network for overnight stock movement prediction. In *IJCAI*, pages 4541–4547, 2021.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

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 *ICLR*, 2024.

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 *ICLR*, 2024.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *ICLR*, 2023.

Zongjiang Shang, Ling Chen, Binqing Wu, and Dongliang Cui. Ada-mshyper: Adaptive multi-scale hypergraph transformer for time series forecasting. In *NeurIPS*, pages 33310–33337, 2024.

Furong Ye, Liming Zhang, Defu Zhang, Hamido Fujita, and Zhiguo Gong. A novel forecasting method based on multi-order fuzzy time series and technical analysis. *Information Sciences*, 367: 41–57, 2016.

Adebiyi Ariyo Ariyo, Aderemi Oluyinka Adewumi, and Charles K. Ayo. Stock price prediction using the ARIMA model. In *ICCMS*, pages 106–112, 2014.

Aditya Gupta and Bhuwan Dhingra. Stock market prediction using hidden markov models. In *SCES*, pages 1–4, 2012.

Edwin J Elton, Martin J Gruber, Stephen J Brown, and William N Goetzmann. *Modern portfolio theory and investment analysis*. 2009.

Khalid Alkhatib, Hassan Najadat, Ismail Hmeidi, and Mohammed K Ali Shatnawi. Stock price prediction using k-nearest neighbor (knn) algorithm. *IJBHT*, 3(3):32–44, 2013.

FX Satriyo D Nugroho, Teguh Bharata Adji, and Silmi Fauziati. Decision support system for stock trading using multiple indicators decision tree. In *ICITACEE*, pages 291–296, 2014.

Kyoung-jae Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1-2):307–319, 2003.

Jingru Fei, Kun Yi, Alex Xing Wang, Qingsong Wen, Xiangxiang Zhu, and Wei Fan. Olivia: Harmonizing time series foundation models with power spectral density. *arXiv preprint arXiv:2605.17340*, 2026.

Yue Ma, Xinyu Wang, Qianli Ma, Qinghe Wang, Mingzhe Zheng, Xiangpeng Yang, Hao Li, Chongbo Zhao, Jixuan Ying, Harry Yang, et al. Group editing: Edit multiple images in one go. *arXiv preprint arXiv:2603.22883*, 2026a.

Xiangfei Qiu, Kangjia Yan, Xvyuan Liu, Xingjian Wu, and Jilin Hu. Bridging time and frequency: A joint modeling framework for irregular multivariate time series forecasting. In *ICML*, 2026b.

Yue Ma, Yingqing He, Xiaodong Cun, Xintao Wang, Siran Chen, Xiu Li, and Qifeng Chen. Follow your pose: Pose-guided text-to-video generation using pose-free videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 4117–4125, 2024.

Xvyuan Liu, Xiangfei Qiu, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Jilin Hu, and Bin Yang. Rethinking irregular time series forecasting: A simple yet effective baseline. In *AAAI*, 2026a.

Yue Ma, Kunyu Feng, Xinhua Zhang, Hongyu Liu, David Junhao Zhang, Jinbo Xing, Yinhan Zhang, Ayden Yang, Zeyu Wang, and Qifeng Chen. Follow-your-creation: Empowering 4d creation through video inpainting. *arXiv preprint arXiv:2506.04590*, 2025a.

Xiangfei Qiu, Xvyuan Liu, Tianen Shen, Xingjian Wu, Hanyin Cheng, Bin Yang, and Jilin Hu. SEER: Transformer-based robust time series forecasting via automated patch enhancement and replacement. In *ICML*, 2026c.

Yue Ma, Zhikai Wang, Tianhao Ren, Mingzhe Zheng, Hongyu Liu, Jiayi Guo, Mark Fong, Yuxuan Xue, Zixiang Zhao, Konrad Schindler, et al. Fastvmt: Eliminating redundancy in video motion transfer. *arXiv preprint arXiv:2602.05551*, 2026b.

Xvyuan Liu, Xiangfei Qiu, Hanyin Cheng, Xingjian Wu, Chenjuan Guo, Bin Yang, and Jilin Hu. ASTGI: Adaptive spatio-temporal graph interactions for irregular multivariate time series forecasting. In *ICLR*, 2026b.

Yue Ma, Yulong Liu, Qiyuan Zhu, Ayden Yang, Kunyu Feng, Xinhua Zhang, Zhifeng Li, Sirui Han, Chenyang Qi, and Qifeng Chen. Follow-your-motion: Video motion transfer via efficient spatial-temporal decoupled finetuning. *arXiv preprint arXiv:2506.05207*, 2025b.

Xingjian Wu, Xiangfei Qiu, Zhengyu Li, Yihang Wang, Jilin Hu, Chenjuan Guo, Hui Xiong, and Bin Yang. CATCH: Channel-aware multivariate time series anomaly detection via frequency patching. In *ICLR*, 2025b.

Xingjian Wu, Xiangfei Qiu, Hanyin Cheng, Zhengyu Li, Jilin Hu, Chenjuan Guo, and Bin Yang. Enhancing time series forecasting through selective representation spaces: A patch perspective. In *NeurIPS*, 2025c.

Sihao Liao, Liang Xie, Yuanchuang Du, Shengshuang Chen, Hongyang Wan, and Haijiao Xu. Stock trend prediction based on dynamic hypergraph spatio-temporal network. *Applied Soft Computing*, 154:111329, 2024.

A Appendix

A.1 Experimental Setup

A.1.1 Datasets

To validate the performance of Hermes, we use three publicly available stock market datasets, with the dataset statistics provided in Table 4. These datasets provide a diverse set of stock time series, allowing us to comprehensively assess the model's effectiveness across different financial scenarios. We evaluate our approach using three real-world datasets from the US stock market. These datasets contain relatively complete stock-industry relationships or Wiki-based company relations. The NASDAQ and NYSE datasets Feng et al. (2019b) include EOD data from 01/02/2013 to 12/08/2017, filtered from the respective markets. Abnormal patterns and penny stocks were removed, while preserving their representative properties, with NASDAQ being more volatile and NYSE more stable. The S&P500 dataset Huynh et al. (2023) gathers historical price data and industry information for the companies listed in the S&P500 index from the Yahoo Finance database, covering the period from 01/04/2016 to 05/25/2022.

Table 4: Statistics of datasets.

Property \ Dataset	NASDAQ	NYSE	S&P500
# Stocks	1026	1737	474
# Industries	113	130	11
# Technical Indicators	5	5	5
Start Time	13-01-02	13-01-02	16-01-04
End Time	17-12-08	17-12-08	22-05-25
Total Time Steps	1281	1281	1611
Train Time Steps	756	756	1006
Validate Time Steps	252	252	253
Test Time Steps	273	273	352

A.1.2 Metrics

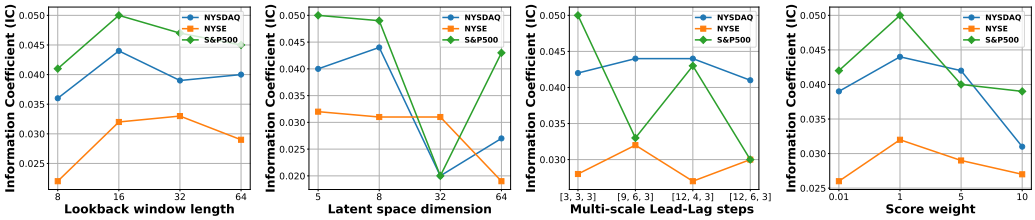
Previous studies adopt different evaluation metrics, making it challenging to comprehensively compare various methods. To provide a comprehensive assessment of model performance, we employ three widely used metrics: IC and RIC, which are correlation-based evaluation metrics, and SR, which is a risk-adjusted return metric.

A.1.3 Baselines

We conduct a comparative analysis between our proposed Hermes and a range of baseline methods, which include prominent approaches in both machine learning and deep learning methods for STSF. Given that existing research has demonstrated the limited effectiveness of traditional statistical methods Fan and Shen (2024); Li et al. (2024), we exclude them from our comparative evaluation to focus on more competitive approaches.

- **LSTM** Hochreiter and Schmidhuber (1997) refers to the standard LSTM model, which operates on sequential data, including closing prices and moving averages over 5, 10, 20, and 30 days, to generate a sequential embedding. A fully connected (FC) layer is then employed to predict the return ratio.
- **ALSTM** Feng et al. (2019c) integrates adversarial training and stochastic simulation into an enhanced LSTM model, allowing it to more effectively capture market dynamics.
- **RGCN** Li et al. (2021) introduces a model that captures both positive and negative correlations among stocks using a correlation matrix computed from historical market data. The LSTM mechanism in RGCN layers helps mitigate over-smoothing issues when predicting overnight stock price movements.

- **GAT** Veličković et al. (2017) employs Graph Attention Networks (GAT) to aggregate stock embeddings, which are encoded by a GRU, on the stock graph. By combining the advantages of Graph Neural Networks (GNN) and attention mechanisms, it enhances performance in large-scale graph settings.
- **RSR** Feng et al. (2019b) introduces a novel neural network-based framework called Relational Stock Ranking, which addresses the stock forecasting problem in a learning-to-rank approach. Additionally, it proposes a new component in neural network modeling, called Temporal Graph Convolution, which is designed to explicitly capture the domain knowledge of stock relationships in a time-sensitive manner.
- **STHAN-SR** Sawhney et al. (2021) introduces a novel Spatio-Temporal Hypergraph Attention Network that models inter-stock relationships of different types and strengths as a hypergraph for stock ranking. It combines temporal Hawkes attention with spatial hypergraph convolutions through hypergraph attention to capture both the correlations in the movements of related stocks and the temporal evolution of their historical features.
- **ESTIMATE** Huynh et al. (2023) integrates temporal generative filters within a memory-based shared parameter LSTM network, enhancing the model's ability to learn temporal patterns for each stock. Furthermore, it introduces attention hypergraph convolutional layers that utilize the wavelet basis—a convolution approach that leverages the polynomial wavelet basis to streamline message passing and focus on localized convolutions.
- **Linear** Zeng et al. (2023) employs a straightforward approach by utilizing only fully connected layers to predict the final price. This method does not incorporate any advanced architectures or temporal dependencies, relying solely on basic linear transformations to generate the forecasting. Despite its simplicity, it serves as a baseline model for comparison with more complex models.
- **StockMixer** Fan and Shen (2024) introduces a lightweight yet effective MLP-based architecture for stock price forecasting. The model incorporates three key components—indicator mixing, time mixing, and stock mixing—which work together to capture the intricate correlations within the stock data.
- **DLinear** Zeng et al. (2023) employs a straightforward linear regression approach that challenges complex Transformer-based architectures. It decomposes the time series and directly regresses historical data for future prediction using a weighted sum operation, demonstrating that simple linear formulations can effectively capture long-term temporal trends.
- **iTransformer** Liu et al. (2024) applies attention and feed-forward networks on inverted dimensions to better capture multivariate correlations. Specifically, it embeds the time series of each variable into variate tokens to model dependencies among variables via attention, while utilizing feed-forward networks to learn nonlinear representations for each token.
- **TimeMixer** Wang et al. (2024) utilizes a fully MLP-based architecture designed to leverage disentangled multiscale time series. It incorporates Past-Decomposable-Mixing (PDM) blocks to mix seasonal and trend components across different scales, and Future-Multipredictor-Mixing (FMM) blocks to aggregate predictions, thereby capturing both microscopic and macroscopic information effectively.
- **PatchTST** Nie et al. (2023) presents an efficient Transformer-based design centered on patching and channel independence. It segments time series into subseries-level patches to serve as input tokens, allowing the model to attend to longer histories with reduced computational costs while retaining local semantic information through independent channel processing.
- **ASHyper** Shang et al. (2024) proposes an Adaptive Multi-Scale Hypergraph Transformer to model group-wise interactions in time series. It features an adaptive hypergraph learning module and a multi-scale interaction module to capture comprehensive patterns. Additionally, it employs node and hyperedge constraints to cluster nodes with similar semantics and differentiate temporal variations.
- **DUET** Qiu et al. (2025b) introduces a dual clustering framework that operates on both temporal and channel dimensions to address pattern heterogeneity. It employs a Temporal Clustering Module (TCM) to group time series into fine-grained distributions and a Channel Clustering Module (CCM) to capture inter-channel relationships in the frequency domain, effectively mitigating noise and modeling complex dependencies.



(a) Lookback length (b) Latent space dimension (c) Lead-Lag steps (d) Score weight

Figure 4: Parameter sensitivity studies of main hyper-parameters in Hermes.

B Limitations

Despite the strong performance of Hermes on multiple real-world stock forecasting benchmarks, several limitations remain. Hermes relies on predefined industry relationships to construct hyperedges, which may not fully capture rapidly evolving or latent market structures in real financial systems. And, although the proposed multi-scale fusion and lead-lag aggregation modules improve forecasting accuracy, the model may still face scalability challenges when applied to extremely large-scale financial markets with thousands of assets and dense inter-industry interactions.

C Impact Statement

This work proposes Hermes, a spatial-temporal hypergraph framework for stock time series forecasting that explicitly models inter-industry lead-lag relationships and multi-scale market dynamics. The proposed framework may benefit financial forecasting systems, quantitative investment research, and market analysis by improving the modeling of complex stock correlations. This research is conducted for academic purposes only. All datasets used in this work are publicly available, and no private or sensitive personal information is involved. And we only use LLMs for grammatical polishing during writing, with all research content being original and ethically compliant.

D Related Work

D.1 Traditional Stock Time Series Forecasting

Early stock forecasting methods rely mainly on statistical learning techniques to capture patterns and relationships in time series Ye et al. (2016); Ariyo et al. (2014); Gupta and Dhingra (2012); Elton et al. (2009). One of the most widely used techniques is the Autoregressive Integrated Moving Average (ARIMA) Ariyo et al. (2014) model, which stabilizes data by combining Autoregressive components, Moving Average, and differencing, thereby modeling time series. Another commonly used traditional method is the Hidden Markov Model (HMM) Gupta and Dhingra (2012), which is used to describe unobservable states in the market and can detect changes in market conditions, making it suitable for analyzing market transitions. Although these methods provide predictive capabilities, they exhibit significant limitations when dealing with nonlinear, dynamic changes, and sudden events in financial markets. As a market environment becomes more complex, traditional statistical methods increasingly fail to provide sufficient predictive accuracy and robustness.

With the rapid development in machine learning, machine learning methods for stock time series forecasting have emerged Alkhatib et al. (2013); Nugroho et al. (2014); Kim (2003). For example, Support Vector Machines (SVMs) Kim (2003) map data to high-dimensional spaces using kernel functions and find optimal separating hyperplanes in those spaces, which allows them to capture complex relationships and dynamics in financial markets. However, SVMs are highly sensitive to parameter selection and often struggles to capture long-term dependencies. Overall, although machine learning methods show considerable potential in stock forecasting, they still rely on manual feature engineering and model design. Moreover, these traditional stock forecasting methods have not delved into the exploration of lead-lag relationships and multi-scale modeling.

D.2 Deep Learning Stock Time Series Forecasting

With the rise of deep learning Fei et al. (2026); Ma et al. (2026a); Qiu et al. (2026b); Ma et al. (2024); Liu et al. (2026a); Ma et al. (2025a), notable progress has occurred in stock forecasting. This is mainly because deep neural networks can model complex nonlinear patterns Qiu et al. (2026c); Ma et al. (2026b); Liu et al. (2026b); Ma et al. (2025b); Wu et al. (2025b). Some studies use recurrent neural networks (RNNs) Nelson et al. (2017); Qin et al. (2017); Akita et al. (2016); Rahman et al. (2019); Wu et al. (2025c) to capture the historical development in individual stock prices and predict their short-term trends. For example, LSTM networks combined with historical stock data and text information are used to predict stock prices Akita et al. (2016). In addition, due to the significant progress of Multi-Layer Perceptrons (MLPs) architecture in the general time series field, the DLinear Zeng et al. (2023), CycleNet Lin et al. (2025), and SparseTSF Lin et al. (2024) show the benefits of this simple architecture for time series forecasting. Increasingly many proposals target stock forecasting performance by improving the MLPs architecture. For example, a series of studies Fan and Shen (2024); Lazcano et al. (2024); Tashakkori et al. (2024) enhance the learning ability of simple MLPs structures by utilizing the MLP-Mixer backbone, thereby improving forecasting accuracy. Next, the Transformer model has become increasingly popular due to the ability of its self-attention mechanism to capture long-term dependencies. For example, Master Li et al. (2024) proposes a novel stock transformer for stock price forecasting to effectively capture stock correlation.

Some deep learning models have begun to explore lead-lag relationships and multi-scale modeling. For instance, StockMixer Fan and Shen (2024) engages in multi-scale modeling by segmenting the original time series into subsequence-level patches and mixing features at different scales. However, this approach lacks effective integration of information across scales; it merely concatenates information from different scales without thorough fusion. On the other hand, ESTIMATE Huynh et al. (2023) takes into account lead-lag relationships through data-driven detection of leading and lagging time series clusters. Nonetheless, this method falls short in capturing fine-grained lead-lag dynamics, as clustering can only reveal global lead-lag relationships. Distinctive from the aforementioned approaches, Hermes employs a sliding window technique and utilizes dynamic temporal aggregation operations to consider fine-grained lead-lag dependencies among stock industries. Moreover, it features a multi-scale fusion module, which utilizes a cross-scale, edge-to-edge message-passing to integrate information from different scales while maintaining the consistency of each scale.

While these deep learning methods have advanced stock forecasting, the complexity of stock data and the need to account for intricate relationships among stocks have led researchers to explore new methods based on hypergraphs.

D.3 Hypergraph-based Stock Time Series Forecasting

Traditional graphs have limitations when it comes to modeling higher-order multivariate relationships, as they can only represent relations between pairs of nodes. To address this issue, Hypergraph Neural Networks (HGNNs) Feng et al. (2019a) were introduced. Hypergraphs extend the concept of simple graphs to enable the capture of relationships among multiple stocks Chen et al. (2020). Hypergraphs have gained increasing attention and application across various fields. Hyperedge in a hypergraph represents a set of vertices, making them suitable for modeling non-pairwise relationships Luo et al. (2014); Sawhney et al. (2020). For example, HGAM Li et al. (2022) is a hypergraph-based reinforcement learning method for stock portfolio selection. THGNN Xiang et al. (2022) is a temporal and heterogeneous graph neural network model that aims to learn dynamic relationships by using two-stage attention mechanisms. STHAN-SR Sawhney et al. (2021) enhances corporate relevance based on Wiki data and uses hypergraph convolution to propagate information from higher-order neighbors. DHSTN Liao et al. (2024) is a dynamic hypergraph network for learning spatio-temporal relations of stocks. Dynamic hypergraphs are generated adaptively using (Graph Attention Networks) GATs to capture time-varying higher-order spatial relationships among stocks. The latest method, ESTIMATE Huynh et al. (2023), employs hypergraphs to capture non-pairwise correlations, utilizing temporal generative filters to identify individual patterns for each stock.

Unlike the above methods, we propose a new hypergraph-based stock forecasting model that specifically takes into account the multi-scale nature of stock data, enabling it to capture the complexity of market dynamics more comprehensively. Furthermore, our approach places particular emphasis on the lead-lag relationships between financial industries, where the changes in one industry may

precede or lag behind those in another. By capturing these key relationships more fully, we aim to improve forecasting accuracy.

E Illustration Figures

Figure 5 illustrates the specific lead-lag interaction process within the window, with a lead-lag step of 3 as an example. Figure 6 depicts three types of message passing: hypernodes for stocks and hyperedges for industries.

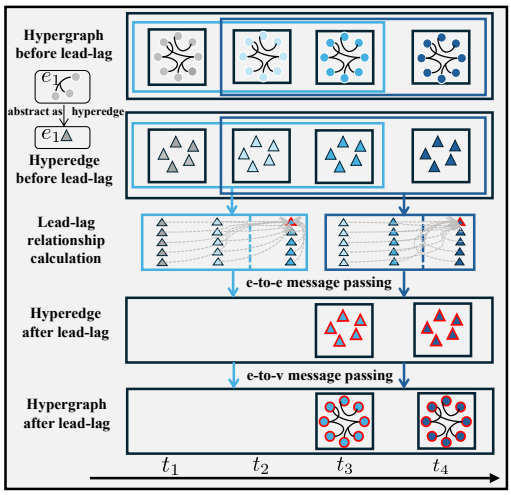


Figure 5: The specific lead-lag interaction process within the window, with a lead-lag step of 3 as an example.

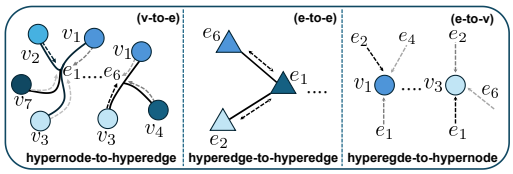


Figure 6: Three types of message passing: hypernodes for stocks and hyperedges for industries.

F Hyperparameter Sensitivity

We also study the parameter sensitivity of the Hermes—see Figure 4.

Lookback window length T: We examine the predictive performance of Hermes as we adjust the lookback window length, T, illustrated in Figure 4a. Consistently across all datasets, a balanced window length emerges as the most effective. Window lengths that are too short result in a swift decline in performance due to inadequate information, whereas excessively lengthy sequences also underperform because of the absence of timely information benefits and the heightened learning expenses for stocks.

Latent space dimension d: In Section 3.2, we enhance the representational capacity of indicators by mapping them into a high-dimensional hidden space d, thereby facilitating the extraction of more intricate hidden relationships. In Figure 4b, we explore the impact of different hidden dimensions d and observe that datasets exhibit optimal performance at varying m values. Notably, when d is set to smaller values such as 5 or 8, models tend to achieve good performance; however, as d increases, models become prone to overfitting, leading to a decline in performance. Furthermore, selecting smaller d not only improves model performance but also effectively reduces the number of model parameters.

Multi-scale Lead-Lag steps $w^1, w^2, \cdots, w^S$: In Section 3.4, for time series at each scale, we study the lead-lag correlations within w^s time steps to better extract complex dependencies in the time series, thereby improving forecasting performance. Here, we select three scales and specify the lead-lag time steps as w^1, w^2, and w^3, respectively. As shown in Figure 4c, we find that different datasets have variations in their multi-scale lead-lag time steps when achieving optimal performance. This is because different datasets possess unique characteristics, and their cyclicity (fine-grained scale) and trend (coarse-grained scale) may change due to external factors. Therefore, we recommend adjusting this parameter specifically when dealing with new datasets.

Score weight α: Besides, we adopt the score weight in section 3.7 to trade off the MSE loss and the ranking-aware loss—see Figure 4d. Experimental results show that all datasets achieve optimal performance and exhibit high stability when the score weight is set to 1. Therefore, we recommend setting this parameter to 1 in future research to obtain more optimized results.

NeurIPS Paper Checklist

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 paper's contributions and scope.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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 it in Appendix B.

   Guidelines:

   - The answer [N/A] 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: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] 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 fully disclose all the information needed to reproduce the main experimental results of the paper.

   Guidelines:

   - The answer [N/A] 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.

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: We provide open access to the data and code in `https://anonymous.4open.science/r/Hermes-E150`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: We have discussed it in Appendix A.1.

   Guidelines:

   - The answer [N/A] 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: [N/A]

   Justification: The paper does not include statistical significance experiments.

   Guidelines:

   - The answer [N/A] 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 discussed it in Table 3.

   Guidelines:

   - The answer [N/A] 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: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] 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]

    Justification: We have discussed it in Appendix C.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] 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: All external resources, including open-source code, public datasets, and the models adopted in this work, are properly cited in the paper.

    Guidelines:

    - The answer [N/A] 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: The new assets introduced in the paper are well-documented, and the documentation is provided alongside the assets.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] 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.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We have discussed it in Appendix C.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.