DSFormer: Dimension-Segment Transformer with Cross-Stock Correlation Modeling for Stock Prediction

Wenyun Xiao[1,*], Kai-Ye Hu[1,*], Ziyuan Liu[1], Hongnian Wang[2,†]
[1]College of Information Science and Technology, Jinan University, Guangzhou, China
{xwyxwy, hu634565644, lzy2002}@stu2024.jnu.edu.cn
[2] Key Laboratory of Digital-Intelligent Disease Surveillance and Health Governance,
North Sichuan Medical College, Nanchong, China
hongnianwang@gmail.com

***Abstract*—Stock return prediction requires extracting predictive signals from high-dimensional and noisy financial time series. Existing methods typically project heterogeneous factors into a shared representation space, which can entangle semantically distinct features and hinder dynamic cross-stock correlation modeling. To address this issue, we propose the Dimension-Segment Transformer (DSFormer), a framework that combines dimension-wise temporal encoding with cross-stock correlation modeling. DSFormer first applies dimension-segment embedding to learn multi-scale intra-stock representations while preserving factor-specific semantics. It then uses data-driven inter-stock attention to capture market-wide co-movements from these disentangled features. Experiments on the CSI300 and CSI500 benchmarks show that DSFormer achieves the best IC and Rank IC among the compared methods, showing the benefit of combining dimension-wise temporal modeling with cross-stock dependency learning for stock return prediction.**

***Index Terms*—stock return prediction, dimension-wise modeling, multi-scale fusion, inter-stock attention, transformer**

I. Introduction

Stock return prediction is important for investment decision-making and risk management [1]. However, financial markets are highly volatile and complex, and the observed signals contain substantial short-term noise that obscures predictive structure. This makes forecasting inherently challenging.

Recent methods such as MASTER [2] typically adopt unified embedding strategies. These methods project diverse financial factors into a shared representation space to model cross-sectional interactions. While this helps capture market-wide trends, it also mixes features with different semantics, such as high-frequency trading volume and smoother moving-average signals. As a result, noisy factors can overwhelm more persistent patterns.

To address this limitation, dimension-wise approaches such as Crossformer [3] model each feature dimension independently and preserve the semantics of heterogeneous factors. However, these models are primarily designed for multivariate time-series samples and usually process stock sequences in isolation. As a result, they focus on intra-stock patterns while overlooking systemic effects such as sector co-movements, where multiple stocks move together. More importantly, when cross-stock dependencies are estimated from mixed embeddings, noisy factors may distort the inferred relationships. Existing solutions therefore tend to prioritize either feature semantics (Crossformer) or market correlations (MASTER), but rarely both.

To bridge this gap, we propose the Dimension-Segment Transformer (DSFormer), a unified framework that integrates dimension-wise temporal encoding with cross-stock correlation modeling. DSFormer first uses dimension-segment embedding to separate high-dimensional factors into independent streams and extract multi-scale patterns. A data-driven correlation module then captures market dynamics from these representations. This design preserves feature semantics while modeling cross-stock dependencies.

Our main contributions are as follows:

1) We propose DSFormer, a unified framework combining dimension-segment embedding with data-driven cross-stock correlation modeling to address the limitations of both unified embedding and purely dimension-wise approaches.
2) We design a two-stage attention encoder that separately models temporal dependencies within each factor dimension, enabling multi-scale pattern extraction while preserving factor-specific semantics.
3) Extensive experiments on the CSI300 and CSI500 benchmarks show that DSFormer achieves the best IC and Rank IC among the compared methods.

II. Related Work

A. Intra-Stock Feature Modeling

Deep learning for stock prediction has evolved from RNN variants [4], [5] to Transformer architectures [6], including

* Wenyun Xiao and Kai-Ye Hu contributed equally to this work.
† Corresponding author: Hongnian Wang.
This work was supported by the Sichuan Science and Technology Program (Grant No. 2026NSFSC1446).

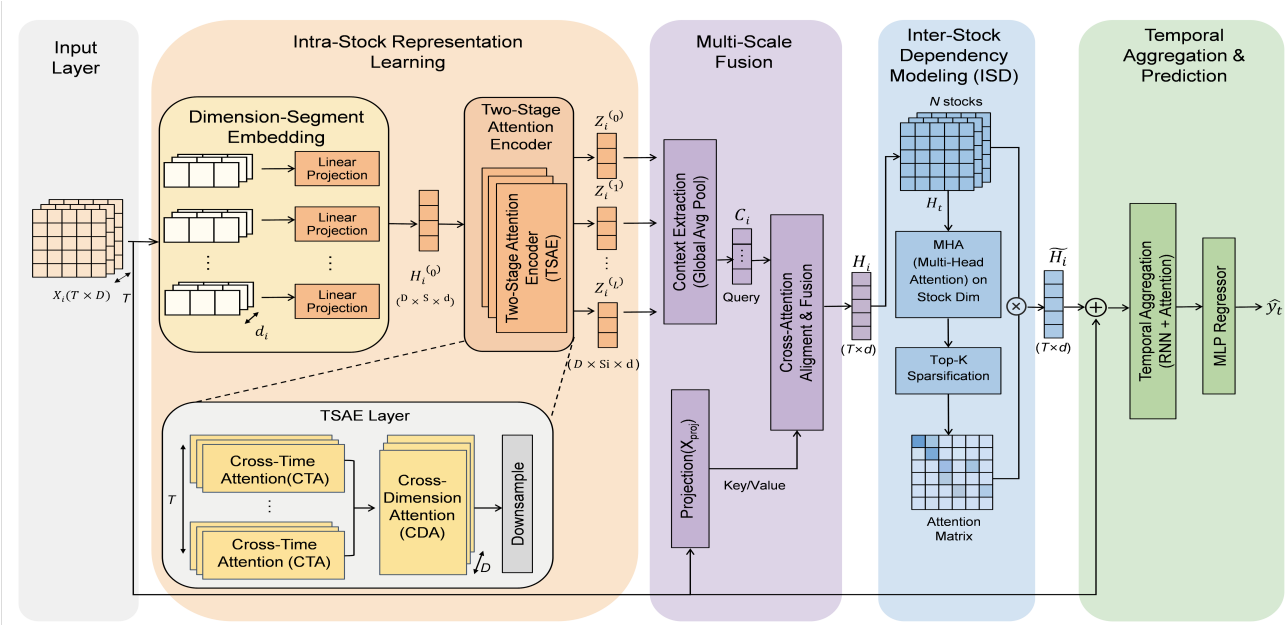

Fig. 1. Overview of the DSFormer framework.

recent advances such as Informer [7] for long-term dependencies. However, dominant models like MASTER [2] and DTML [8] typically adopt a unified embedding strategy. When diverse factors are projected into a single vector, features with distinct characteristics may become entangled, potentially allowing high-frequency noise to overshadow more persistent trend signals. Recent works such as MATCC [9] and CFC-GMixer [10] have explored richer temporal and multi-scale feature modeling, yet these methods still rely on unified embeddings that mix heterogeneous features.

To address this limitation, dimension-wise strategies like Crossformer [3] and alternative architectures such as structured state spaces [11] segment data along feature dimensions to preserve semantic integrity. While effective for individual series, these methods typically process stock sequences in isolation, neglecting the cross-sectional dependencies essential for assessing systematic market risk.

B. Cross-Stock Correlation Modeling

Modeling stock interactions has shifted from static graphs based on industry priors [12], [13] or shareholder links [14] to dynamic correlation mining. Hierarchical methods such as HATS [15] and HATR [16] capture multi-level relationships, while hypergraph-based methods such as ESTIMATE [17] model multi-order dynamics. More recently, MASTER [2] employs market-guided attention to aggregate real-time signals, COGRASP [18] constructs dynamic co-occurrence graphs to capture rapid market shifts, and ALSGCN [19] models long- and short-term stock dependencies through graph convolution. Frequency-domain variants such as CFGAN [20] further explore inter-series dependency modeling in the spectral domain.

Despite these algorithmic improvements, a fundamental limitation persists in most time-domain dynamic models: they typically compute correlations based on mixed representations. Whether deriving correlations from hypergraphs [17] or attention mechanisms [2], they generally project diverse factors into unified embeddings. The learned relationships may therefore be influenced by noise from mixed feature representations, particularly in volatile market conditions. In contrast, our approach models dynamic correlations on top of separated feature representations, ensuring that relationships are learned from clean, semantic-aware signals.

III. Methodology

A. Problem Formulation

Assume a stock universe containing N stocks. For the i-th stock, we are given its historical multivariate time series:

$$X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,T}\},$$

where T is the lookback window length. Each observation $x_{i,t} \in \mathbb{R}^D$ contains D feature values, instantiated in our experiments as daily Alpha360 factors from Qlib.

All stocks' historical sequences are jointly represented as:

$$X = \{X_1, X_2, \ldots, X_N\}.$$

The task is to learn a prediction function:

$$f : X \to Y,$$

where $Y = \{y_1, y_2, \ldots, y_N\}$, and y_i denotes the return of the i-th stock over a future prediction horizon.

The model must simultaneously capture: (1) temporal and feature-wise dependencies within each stock; (2) cross-sectional correlations among different stocks; and (3) complementary information across multiple temporal scales.

B. Overview

Figure 1 shows the overall architecture of DSFormer. The model has four main components. **Intra-stock representation learning** extracts multi-scale temporal patterns within each stock via dimension-segment embedding and two-stage attention. **Multi-scale fusion** integrates information across temporal scales through cross-attention. **Inter-stock dependency modeling** captures dynamic correlations among stocks through time-step-wise multi-head attention. **Temporal aggregation and prediction** then maps the enhanced sequence to the return prediction. Specifically, $X_i \in \mathbb{R}^{T\times D}$ is converted into $H_i^{(0)} \in \mathbb{R}^{D\times S\times d}$, encoded into $\{Z_i^{(0)}, \ldots, Z_i^{(M)}\}$, fused into $H_i \in \mathbb{R}^{T\times d}$, enhanced to $\tilde{H}_i \in \mathbb{R}^{T\times d}$, and finally mapped to $\hat{y}_i$.

C. Intra-Stock Representation Learning

Dimension-Segment Embedding. For each stock sequence $X_i \in \mathbb{R}^{T\times D}$, we partition the temporal axis into non-overlapping segments of length L_{seg}. If T is not divisible by L_{seg}, we pad the earliest time steps with zeros. This yields

$$S = \left\lceil \frac{T}{L_{\text{seg}}} \right\rceil$$

segments:

$$X_i = \{X_{i,1}, X_{i,2}, \ldots, X_{i,S}\}, \quad X_{i,s} \in \mathbb{R}^{L_{\text{seg}}\times D}.$$

To preserve semantic independence, we process each feature dimension separately. For feature dimension d', the values within each segment are projected independently into a segment token:

$$e_{i,s,d'} = \text{Linear}\left(X_{i,s}[:, d']\right) \in \mathbb{R}^d,$$

where the linear layer maps an L_{seg}-dimensional segment into a d-dimensional embedding. Stacking the S segment tokens for feature d' gives

$$H_{i,d'}^{(0)} = [e_{i,1,d'}, e_{i,2,d'}, \ldots, e_{i,S,d'}] \in \mathbb{R}^{S\times d}.$$

Stacking embeddings from all D dimensions yields the initial representation:

$$H_i^{(0)} = [H_{i,1}^{(0)}, \ldots, H_{i,D}^{(0)}] \in \mathbb{R}^{D\times S\times d}.$$

Two-Stage Attention Encoder (TSAE). To preserve dimension-wise disentanglement, we adopt a two-stage attention encoder that separately models temporal dependencies within each feature dimension and cross-dimension interactions within each temporal segment. Stacking such layers with temporal downsampling yields multi-scale intra-stock representations.

Cross-Time Attention (CTA). CTA captures long-range dependencies across temporal segments within the same feature dimension, modeling the independent temporal evolution of each factor. For each dimension $d' \in \{1, \ldots, D\}$, the sequence $H_{i,d'}^{(0)} \in \mathbb{R}^{S\times d}$ is processed independently:

$$\hat{H}_{\text{time}}^{i,d'} = \text{LN}\left(H_{i,d'}^{(0)} + \text{MHSA}(H_{i,d'}^{(0)})\right),$$

$$H_{\text{time}}^{i,d'} = \text{LN}\left(\hat{H}_{\text{time}}^{i,d'} + \text{FFN}(\hat{H}_{\text{time}}^{i,d'})\right).$$

Outputs from all D dimensions are concatenated to form $H_{\text{time}}^i \in \mathbb{R}^{D\times S\times d}$.

Cross-Dimension Attention (CDA). CDA then models interactions among different features within each temporal segment, capturing complementary relationships such as price-volume coordination. For each segment $s \in \{1, \ldots, S\}$, let $Z_s \in \mathbb{R}^{D\times d}$ denote the feature matrix at that segment, obtained by slicing H_{time}^i along the temporal axis. MHSA is applied along the dimension axis:

$$\hat{Z}_s = \text{LN}\left(Z_s + \text{MHSA}(Z_s)\right),$$

$$Z_{\text{dim}}^s = \text{LN}\left(\hat{Z}_s + \text{FFN}(\hat{Z}_s)\right).$$

Concatenating outputs across all segments gives the initial multi-scale representation:

$$Z_i^{(0)} = [Z_{\text{dim}}^1, \ldots, Z_{\text{dim}}^S] \in \mathbb{R}^{D\times S\times d}.$$

This serves as the input to the subsequent multi-scale encoder.

Multi-Scale Encoding with Downsampling. By stacking M additional encoder layers with 2× temporal downsampling after each layer, we obtain $M+1$ scales in total. Each layer $l \in \{1, \ldots, M\}$ applies CTA and CDA, then downsamples the temporal dimension:

$$Z_i^{(l)} = \text{Downsample}(\text{CDA}(\text{CTA}(Z_i^{(l-1)}))), \quad S_l = S_{l-1}/2.$$

Finally, the encoder outputs $M+1$ scales:

$$\{Z_i^{(0)}, Z_i^{(1)}, \ldots, Z_i^{(M)}\}, \quad Z_i^{(l)} \in \mathbb{R}^{D\times S_l\times d},$$

where $S_0 = S$ and $S_l = S/2^l$ for $l \geq 1$. $Z_i^{(0)}$ corresponds to the finest scale and $Z_i^{(M)}$ captures long-term trends.

D. Multi-Scale Fusion

Market signals operate at multiple frequencies. We therefore use an adaptive fusion module that extracts a global context vector from all encoder outputs and aligns it with the original features through cross-attention.

Multi-Scale Context Extraction. This module aligns multi-scale intra-stock representations from the encoder. Each scale representation $Z_i^{(l)} \in \mathbb{R}^{D\times S_l\times d}$ is first reshaped as:

$$F_i^{(l)} = \text{Reshape}(Z_i^{(l)}) \in \mathbb{R}^{S_l\times(D\cdot d)}.$$

Global average pooling then extracts scale-specific context vectors:

$$c_i^{(l)} = \frac{1}{S_l}\sum_{s=1}^{S_l} F_i^{(l)}(s, :).$$

All scale contexts are concatenated:

$$C_i = \text{Concat}\left(c_i^{(0)}, \ldots, c_i^{(M)}\right).$$

Cross-Attention Alignment and Fusion. The concatenated multi-scale context C_i is projected by a feed-forward network to obtain a unified context representation $\tilde{C}_i \in \mathbb{R}^{1\times d_h}$, where

d_h is the hidden dimension. The original features are projected as

$$X^i_{\text{proj}} = X_i W_x + \mathbf{1} b_x^\top \in \mathbb{R}^{T \times d_h},$$

where $W_x \in \mathbb{R}^{D \times d_h}$, $b_x \in \mathbb{R}^{d_h}$, and $\mathbf{1} \in \mathbb{R}^T$ is an all-ones vector.

Cross-attention is applied with $\tilde{C}_i$ as query and X^i_{proj} as keys and values. We first compute

$$q_i = \tilde{C}_i W_{qc}, \quad K_i = X^i_{\text{proj}} W_{kc}, \quad V_i = X^i_{\text{proj}} W_{vc},$$

where $W_{qc}, W_{kc}, W_{vc} \in \mathbb{R}^{d_h \times d_h}$. The aligned context is then

$$H^i_{\text{align}} = \text{Softmax}\left(\frac{q_i K_i^T}{\sqrt{d_h}}\right) V_i \in \mathbb{R}^{1 \times d_h}.$$

The aligned context is broadcast along the temporal dimension and fused via residual connection:

$$H^i_{\text{fusion}} = \text{LN}\left(X^i_{\text{proj}} + \text{Repeat}(H^i_{\text{align}}, T)\right) \in \mathbb{R}^{T \times d_h}.$$

Finally, a linear projection restores dimensionality:

$$H_i = H^i_{\text{fusion}} W_r + \mathbf{1} b_r^\top \in \mathbb{R}^{T \times d},$$

where $W_r \in \mathbb{R}^{d_h \times d}$ and $b_r \in \mathbb{R}^d$.

E. Inter-Stock Dependency Modeling (ISD)

After multi-scale fusion, each stock i has a fused temporal representation $H_i \in \mathbb{R}^{T \times d}$. We denote each time step's hidden state as $h_{i,t} \in \mathbb{R}^d$, forming the sequence $H_i = \{h_{i,1}, h_{i,2}, \ldots, h_{i,T}\}$.

Stock price movements are not independent but are linked through time-varying relationships. To explicitly model these cross-sectional dynamics, we introduce the ISD module. Unlike static sector-based graphs, which miss transient dependencies, ISD re-estimates relationships at *every* time step in a data-driven manner.

At each time step t, we stack the representations of all N stocks in the current batch:

$$H_t = \text{Stack}(h_{1,t}, h_{2,t}, \ldots, h_{N,t}) \in \mathbb{R}^{N \times d}.$$

In practice, samples are grouped by trading date, so ISD is computed over stocks from the same day.

A multi-head attention mechanism is then applied along the stock dimension to learn interactions. Queries, keys, and values are first obtained by linear projections of the stock representations:

$$Q_t = H_t W_q, \quad K_t = H_t W_k, \quad V_t = H_t W_v.$$

Here, $W_q, W_k, W_v \in \mathbb{R}^{d \times d}$ are learnable projection matrices.

For each head $m \in \{1, \ldots, N_h\}$, we define the attention matrix

$$A^{(t)}_m = \text{Softmax}\left(\frac{Q_{t,m} K^T_{t,m}}{\sqrt{d/N_h}}\right) \in \mathbb{R}^{N \times N},$$

where $Q_{t,m}$ and $K_{t,m}$ denote the head-specific projections. The matrix $A^{(t)}_m$ represents directed influence among stocks. A high weight from stock j to i implies that j's current state provides useful context for predicting i. For sparsification, we aggregate attention weights across heads by averaging:

$$A^{(t)} = \frac{1}{N_h} \sum_{m=1}^{N_h} A^{(t)}_m \in \mathbb{R}^{N \times N}.$$

To suppress spurious correlations in a noisy market, we apply ratio-based Top-K_n sparsification to the attention matrix, retaining only the top $K_n = \lceil \rho N \rceil$ weights for each stock:

$$\tilde{A}^{(t)}_{i,j} = \begin{cases} A^{(t)}_{i,j}, & \text{if } j \in \text{Top-}K_n(A^{(t)}_{i,:}), \\ 0, & \text{otherwise,} \end{cases}$$

where $\rho \in (0, 1]$ is the sparsification ratio and $K_n = \lceil \rho N \rceil$ is the number of retained neighbors for each stock. This keeps the model focused on the strongest inter-stock connections and reduces the effect of noisy correlations.

The aggregated inter-stock message is then computed as:

$$M_t = \tilde{A}^{(t)} V_t \in \mathbb{R}^{N \times d}.$$

Finally, a residual connection and layer normalization fuse this market context with the original stock representation:

$$\tilde{H}_t = \text{LN}\left(H_t + \text{Dropout}(M_t)\right).$$

This process is repeated for all time steps $t = 1 \ldots T$, yielding the inter-stock enhanced representations:

$$\tilde{H}_i = \{\tilde{h}_{i,1}, \tilde{h}_{i,2}, \ldots, \tilde{h}_{i,T}\}, \quad \tilde{h}_{i,t} \in \mathbb{R}^d.$$

F. Temporal Aggregation and Prediction

After inter-stock dependency modeling, each stock is represented by a market-aware temporal sequence:

$$\tilde{H}_i = \{\tilde{h}_{i,1}, \tilde{h}_{i,2}, \ldots, \tilde{h}_{i,T}\}, \quad \tilde{h}_{i,t} \in \mathbb{R}^d,$$

This sequence is then passed to the prediction head.

Since $\tilde{h}_{i,t}$ lies in a latent feature space different from the original input, a feature adapter is employed to project it back to the original feature dimension:

$$\hat{x}_{i,t} = \phi(\tilde{h}_{i,t}), \quad \hat{x}_{i,t} \in \mathbb{R}^D,$$

where $\phi(\cdot)$ is a learnable feed-forward projection.

The adapted contextual signal is then injected into the original time-series input via a residual connection:

$$\bar{x}_{i,t} = x_{i,t} + \text{Dropout}(\hat{x}_{i,t}),$$

yielding a market-aware temporal sequence

$$\bar{X}_i = \{\bar{x}_{i,1}, \ldots, \bar{x}_{i,T}\}.$$

To aggregate temporal information, we employ an RNN followed by temporal attention:

$$o_{i,t} = \text{RNN}(\bar{x}_{i,t}, o_{i,t-1}), \qquad s_{i,t} = \text{MLP}(o_{i,t}),$$

$$\alpha_{i,t} = \frac{\exp(s_{i,t})}{\sum_{\tau=1}^{T} \exp(s_{i,\tau})},$$

where the softmax normalization is taken over all time steps of stock i. The final representation combines the last hidden state and attention-weighted sum:

$$z_i = \text{Concat}\left(o_{i,T}, \sum_{t=1}^{T} \alpha_{i,t} o_{i,t}\right).$$

An MLP regressor produces the predicted return $\hat{y}_i = \text{MLP}(z_i)$. The model is trained via MSE loss: $\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$.

G. Complexity Analysis

The computational complexity of DSFormer is

$$O\big(M(DS^2d + SD^2d) + TN^2d + TNK_nd\big),$$

where M, D, S, N, T, K_n, and d denote the number of encoder layers, features, segments, stocks, time steps, retained neighbors, and hidden dimension, respectively, with $K_n = \lceil \rho N \rceil$. Temporal segmentation reduces the intra-stock attention cost from $O(T^2)$ to $O(S^2)$, where $S = \lceil T/L_{\text{seg}} \rceil \ll T$. In the ISD module, the TN^2d term comes from computing pairwise attention scores over stocks at each time step, while the TNK_nd term corresponds to sparse aggregation after Top-K_n pruning. Therefore, ratio-based sparsification reduces the aggregation cost from dense $O(TN^2d)$ to sparse $O(TNK_nd)$, but does not change the quadratic cost of score computation. For stock universes of the size used here and daily-frequency inference, the overall cost remains manageable.

IV. Experiments

A. Experimental Setup

Datasets. We evaluate DSFormer on two benchmark datasets, CSI300 and CSI500, covering 2009–2024 with daily Alpha360 factors from Qlib. Following standard practice, we adopt a chronological split: training (2009–2019), validation (2020–2021), and test (2022–2024). The lookback window is 60 days, and the prediction target is the next-day return.

Baselines. We compare DSFormer with baselines from three categories. *Traditional sequential models* include MLP (3-layer multilayer perceptron), LSTM [4], GRU [21], and ALSTM [4] (Attention-based LSTM, a simplified implementation of DA-RNN [4] that applies temporal attention over recurrent hidden states). *Graph-based methods* include GAT [22], which models stock relationships through predefined sector-based graphs. *Attention-based methods* include Transformer [6], Crossformer [3] (with dimension-segment embedding for multivariate time-series forecasting), and MASTER [2] (a recent inter-stock attention model with market-guided gating).

Evaluation Metrics. Primary metrics are Information Coefficient (IC) and Rank Information Coefficient (Rank IC), which are standard in quantitative finance literature [2], [23]. IC measures the Pearson correlation between predicted returns and realized returns, reflecting prediction accuracy. Rank IC measures the Spearman rank correlation, reflecting ranking consistency critical for portfolio construction. Both metrics are computed daily and averaged over the test period. Higher values indicate better prediction performance.

Implementation Details. All models are implemented in PyTorch within the Qlib framework [23]. We use the Alpha360 benchmark factors with standard preprocessing: training-set robust z-score normalization and zero filling for missing values, without additional per-stock or rolling normalization. For DSFormer, we set segment length $L_{\text{seg}} = 8$, model dimension $d = 64$, 2 encoder layers, 2 TSAE heads, 8 ISD heads, and sparsification ratio $\rho = 0.1$. Because the lookback window is 60, we pad each sequence to length 64 before segmentation, resulting in $S = 8$ segments. For baselines, we tune layers $\{1, 2, 3\}$, hidden dimensions $\{64, 128, 256\}$, and learning rates $\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$, selecting the best configuration based on validation IC. All methods are trained for at most 150 epochs with early stopping on NVIDIA Tesla V100 GPUs. Each experiment is repeated 10 times; we report mean $\pm$ std.

B. Main Results

Table I shows that DSFormer achieves the best IC and Rank IC among the compared methods on both CSI300 and CSI500.

Compared with MASTER, DSFormer improves on both datasets (e.g., +7.3% IC on CSI300 and +2.9% on CSI500). This matches our design: dimension-wise modeling preserves feature semantics and reduces noise from unified embeddings. In DSFormer, CTA models temporal evolution within each factor, while CDA captures interactions across factors.

Compared with Crossformer, DSFormer shows clear improvements. This suggests that dimension-wise modeling alone is not enough without explicit inter-stock dependency modeling. In financial markets influenced by systematic factors such as sector rotation and liquidity contagion [24], treating stocks independently can discard useful cross-sectional information. DSFormer addresses this limitation by combining dimension-wise modeling with data-driven inter-stock attention. It captures both factor-specific semantics and market-wide co-movements. DSFormer also outperforms the graph-based baseline GAT (CSI300 IC: 0.0710 vs. 0.0633), whose predefined sector graph may be less suitable for rapidly changing correlations.

To further assess the contribution of each component, we conduct a two-level ablation study. At the main-component level, removing ISD degrades CSI300 IC from 0.0710 to 0.0640 (-9.9%), indicating that cross-sectional correlation modeling adds useful predictive information. Removing TSAE reduces CSI300 IC to 0.0665 (-6.3%) and CSI500 IC to 0.0744 (-1.7%), showing the value of dimension-wise disentanglement. The two modules play different roles: TSAE preserves factor-specific semantics, while ISD injects market-wide context. The full DSFormer model outperforms both single-removal variants and the external ALSTM baseline (0.0710 vs. 0.0615 on CSI300). At the sub-component level, removing Cross-Dimension Attention (CDA) causes a larger performance drop (CSI300 IC: 0.0666, -6.2%) than removing Cross-Time Attention (CTA) (CSI300 IC: 0.0691, -2.7%). This

TABLE I
PERFORMANCE COMPARISON ON CSI300 AND CSI500 (BEST RESULTS IN BOLD).

Model	CSI300		CSI500	
	IC	Rank IC	IC	Rank IC
MLP	0.0513 ± 0.0016	0.0482 ± 0.0014	0.0637 ± 0.0014	0.0564 ± 0.0016
LSTM	0.0611 ± 0.0014	0.0582 ± 0.0011	0.0724 ± 0.0012	0.0654 ± 0.0015
GRU	0.0615 ± 0.0019	0.0578 ± 0.0017	0.0741 ± 0.0008	0.0672 ± 0.0008
ALSTM	0.0615 ± 0.0021	0.0580 ± 0.0022	0.0739 ± 0.0011	0.0669 ± 0.0010
Transformer	0.0577 ± 0.0018	0.0554 ± 0.0019	0.0694 ± 0.0010	0.0626 ± 0.0009
GAT	0.0633 ± 0.0014	0.0592 ± 0.0015	0.0734 ± 0.0027	0.0662 ± 0.0026
Crossformer	0.0474 ± 0.0009	0.0443 ± 0.0010	0.0642 ± 0.0008	0.0570 ± 0.0007
MASTER	0.0662 ± 0.0030	0.0633 ± 0.0032	0.0736 ± 0.0021	0.0682 ± 0.0017
-wo ISD	0.0640 ± 0.0016	0.0599 ± 0.0017	0.0727 ± 0.0024	0.0659 ± 0.0027
-wo TSAE	0.0665 ± 0.0017	0.0624 ± 0.0020	0.0744 ± 0.0029	0.0678 ± 0.0032
-wo CTA	0.0691 ± 0.0014	0.0654 ± 0.0012	0.0734 ± 0.0015	0.0665 ± 0.0011
-wo CDA	0.0666 ± 0.0012	0.0626 ± 0.0011	0.0744 ± 0.0020	0.0679 ± 0.0022
DSFormer	**0.0710 ± 0.0022**	**0.0672 ± 0.0022**	**0.0757 ± 0.0018**	**0.0690 ± 0.0018**

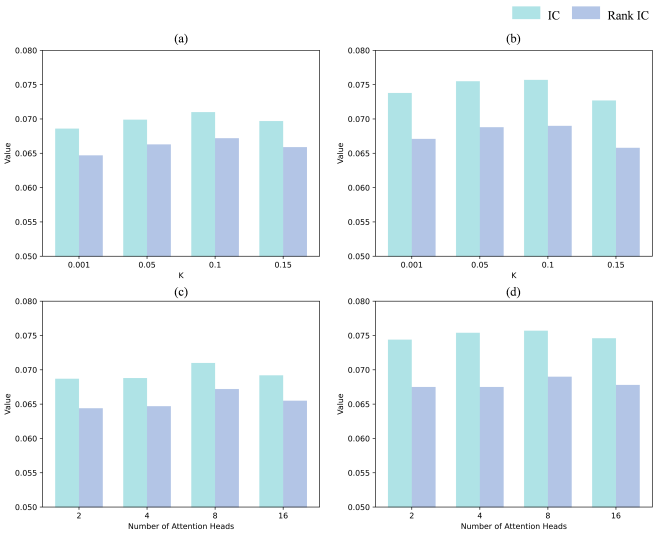

Fig. 2. Hyperparameter sensitivity analysis of DSFormer's ISD module. The model achieves optimal performance at sparsification ratio $\rho = 0.1$ and 8 attention heads; both smaller and larger values degrade performance.

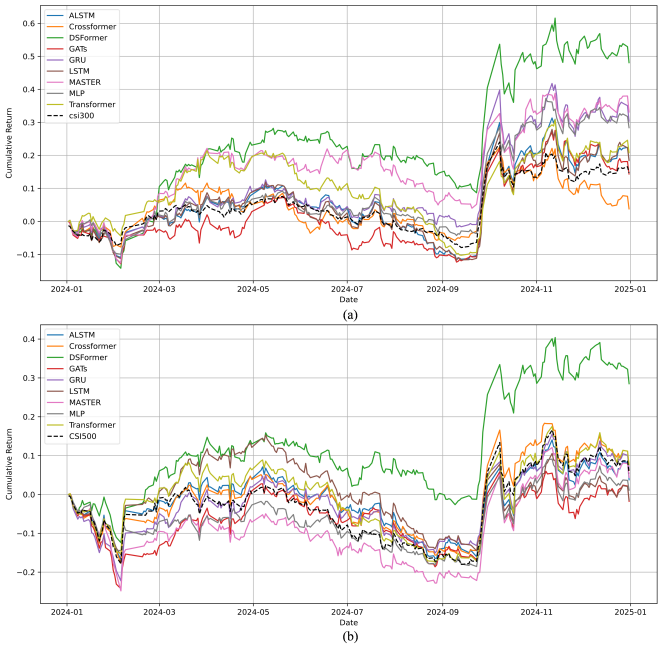

Fig. 3. Cumulative returns of DSFormer and baselines on CSI300 and CSI500 in 2024. Portfolios are constructed by selecting the top-15 stocks based on predicted returns. The dashed line denotes the market index benchmark.

suggests that cross-dimension interactions matter more than temporal attention in this setting.

C. Hyperparameter Analysis

We analyze DSFormer's sensitivity to two key hyperparameters in the ISD module. Results are shown in Figure 2.

Sparsification Ratio. As the sparsification ratio ρ increases from 0.001 to 0.1, both IC and Rank IC improve on both

datasets, suggesting that including more correlated stocks helps prediction. However, further increasing ρ to 0.15 leads to saturation, indicating that weak correlations introduce noise. Top-K_n filtering retains the most relevant inter-stock dependencies while keeping the downstream aggregation cost manageable in practice. Based on this analysis, we set $\rho = 0.1$.

Number of Attention Heads. Increasing the number of attention heads from 2 to 8 improves performance on both datasets. Multi-head attention captures different inter-stock interaction patterns through parallel attention distributions. Beyond 8 heads, the gains plateau, likely because additional heads add redundancy. Overall, DSFormer remains stable across a reasonable hyperparameter range.

D. Investment Simulation

We conduct backtesting experiments to evaluate the model in a simple trading setting.

Setup. We adopt daily rebalancing. At each trading day, stocks in the CSI300/CSI500 universe are ranked by predicted scores, and an equally-weighted long-only portfolio is constructed using the top-K_{port} stocks, with $K_{\text{port}} = 15$. All trades are assumed to be executed at the next-day opening price to avoid look-ahead bias. For simplicity, this backtest does not explicitly model transaction costs, slippage, or turnover constraints.

Results. Figure 3 presents cumulative returns in 2024. Under this simplified, transaction-cost-free protocol, DSFormer achieves the highest cumulative returns on both datasets, outperforming all baselines and the market benchmark. The gap is larger on CSI500, which is consistent with the higher volatility of mid-cap stocks. During market drawdowns, DSFormer shows smaller losses and a steadier return curve. These results suggest that stronger IC and Rank IC can translate to better portfolio-level performance under this setting.

V. Conclusion

We propose DSFormer, a unified framework that combines dimension-wise modeling with inter-stock dependency learning for stock return prediction. Experiments on CSI300 and CSI500 show that it achieves the best IC and Rank IC among the compared methods. The results support jointly modeling feature-level semantics and cross-stock correlations. Future work will explore alternative data sources, more adaptive cross-stock structures, and broader financial prediction tasks.

References

[1] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019," *Applied Soft Computing*, vol. 90, p. 106181, 2020.

[2] T. Li, Z. Liu, Y. Shen, X. Wang, H. Chen, and S. Huang, "MASTER: Market-guided stock transformer for stock price forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 1, 2024, pp. 162–170.

[3] Y. Zhang and J. Yan, "Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting," in *The Eleventh International Conference on Learning Representations*, 2023.

[4] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2017, pp. 2627–2633.

[5] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, "Deep learning with long short-term memory for time series prediction," *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114–119, 2019.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[7] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, May 2021, pp. 11 106–11 115.

[8] J. Yoo, Y. Soun, Y.-c. Park, and U. Kang, "Accurate multivariate stock movement prediction via data-axis transformer with multi-level contexts," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 2037–2045.

[9] Z. Cao, J. Xu, C. Dong, P. Yu, and T. Bai, "MATCC: A novel approach for robust stock price prediction incorporating market trends and cross-time correlations," in *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 2024, pp. 187–196.

[10] L. Gao, Z. Li, J. Yu, J. Zhang, W. Yao, and H. Wang, "CFC-GMixer: Closed-form continuous-time graph networks with multi-scale feature mixer for stock predictions," in *International Joint Conference on Neural Networks*, Rome, Italy, 2025, pp. 1–8.

[11] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," in *International Conference on Learning Representations*, 2022.

[12] Y. Chen, Z. Wei, and X. Huang, "Incorporating corporation relationship via graph convolutional neural networks for stock price prediction," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1655–1658.

[13] F. Feng, X. He, X. Wang, C. Luo, Y. Liu, and T.-S. Chua, "Temporal relational ranking for stock prediction," *ACM Transactions on Information Systems*, vol. 37, no. 2, pp. 1–30, 2019.

[14] R. Sawhney, S. Agarwal, A. Wadhwa, and R. R. Shah, "Spatiotemporal hypergraph convolution network for stock movement forecasting," in *IEEE International Conference on Data Mining*, 2020, pp. 482–491.

[15] R. Kim, C. H. So, M. Jeong, S. Lee, J. Kim, and J. Kang, "HATS: A hierarchical graph attention network for stock movement prediction," *arXiv preprint arXiv:1908.07999*, 2019.

[16] H. Wang, S. Li, T. Wang, and J. Zheng, "Hierarchical adaptive temporal-relational modeling for stock trend prediction," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021, pp. 3691–3698.

[17] T. T. Huynh, M. H. Nguyen, T. T. Nguyen, P. L. Nguyen, M. Weidlich, Q. V. H. Nguyen, and K. Aberer, "Efficient integration of multi-order dynamics and internal dynamics in stock movement prediction," in *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 2023, pp. 850–858.

[18] Z. Li, Z. Song, T. Yuan, and X. Fu, "COGRASP: Co-occurrence graph based stock price forecasting," in *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 2025, pp. 7527–7535.

[19] J. Yu, W. Yao, Z. Li, L. Gao, W. Xiao, and H. Wang, "ALSGCN: An attention-based long- and short-term graph convolutional network for stock recommendation," in *IEEE International Conference on Systems, Man, and Cybernetics*, Vienna, Austria, 2025, pp. 3186–3191.

[20] Z. Li, L. Gao, W. Qiu, W. Xiao, and H. Wang, "CFGAN: Complex frequency-domain graph attention network for time series forecasting," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2026, pp. 1231–1235.

[21] R. Zhao, D. Wang, R. Yan, K. Mao, F. Shen, and J. Wang, "Machine health monitoring using local feature-based gated recurrent unit networks," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1539–1548, 2017.

[22] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, Feb. 2018.

[23] X. Yang, W. Liu, D. Zhou, J. Bian, and T.-Y. Liu, "Qlib: An AI-oriented quantitative investment platform," *arXiv preprint arXiv:2009.11189*, 2020.

[24] A. Nobi, S. E. Maeng, G. G. Ha, and J. W. Lee, "Effects of global financial crisis on network structure in a local stock market," *Physica A: Statistical Mechanics and its Applications*, vol. 407, pp. 135–143, 2014.