# HiMarket: A HIERARCHICAL MARKET WORLD MODEL FOR STOCK SELECTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Stock selection plays a pivotal role in quantitative investment, requiring models to identify stocks likely to outperform under changing market conditions. Recently, representation learning and pretraining methods have shown promise for this task. However, market states are implicit, hierarchical, and time-varying, making it difficult to separate shared dynamics from stock-specific variation. Moreover, predicting shared dynamics alone does not identify which stocks will outperform. To address these challenges, we propose **HiMarket**, a **Hi**erarchical **Market** world model for stock selection. Specifically, a hierarchical market-state construction module decomposes encoded stock representations into global, dynamic-group, and residual components. Then, during pretraining, a market-conditioned residual prediction objective uses forecast shared market states to predict future stock residual representations. Finally, a stock-conditioned factor-query readout combines encoded stock representations with their residual components to form stock-specific queries that retrieve context from historical global and group memories. Experiments on three real-world stock-market datasets demonstrate that HiMarket achieves state-of-the-art (SOTA) performance.

## 1 INTRODUCTION

Stock selection is a core task in quantitative investment, supporting portfolio construction. It ranks stocks by their expected future returns using available information (Feng et al., 2019b). Effective selection requires understanding both shared market conditions and differences among individual stocks.

Existing stock-selection methods model temporal patterns and relationships among stocks to predict future returns or rankings (Feng et al., 2019b; Li et al., 2024; Fan & Shen, 2024). Self-supervised learning further supports representation learning through stock-specific auxiliary tasks (Wang et al., 2025). More recently, Mahns et al. (2026) investigate joint-embedding predictive architectures (JEPA) in U.S. equity markets. Their study shows that the learned market-state representations capture persistent volatility and correlation structure and provide information about some gradual recovery dynamics. These findings provide an empirical foundation for learning market structure and its evolution through latent prediction.

Although recent work has shown that JEPA can capture persistent market structure and aspects of its evolution (Mahns et al., 2026), the usefulness of these representations for cross-sectional stock selection remains to be established. Specifically, shared market conditions provide context for interpreting individual stock signals, while stock selection requires distinguishing which stocks are likely to earn higher future returns. Building on these findings, we seek to connect predictive market-state learning with the needs of stock selection. However, learning representations that capture evolving market context and support stock-level ranking is non-trivial due to two primary challenges:

**(1) The market state is implicit, hierarchical, and time-varying.** Price–volume observations entangle market-wide movements, behavior shared by groups of stocks, and stock-specific variation. The underlying shared components must be inferred from the observed panel, while relationships among stocks can change with market conditions (Li et al., 2024). A useful market-state representation must therefore capture both the latent hierarchy and its temporal evolution.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

**(2) Shared market states must be adapted to stock-level ranking.** Representations learned for market prediction are not necessarily aligned with cross-sectional stock ranking. The same market conditions can have different implications for different stocks, depending on their individual characteristics. A stock-selection model therefore needs to interpret shared context together with stock-level information. The challenge is to learn stock-level variation under evolving market conditions and determine which aspects of the shared context are relevant to each stock's ranking.

To address these challenges, we propose **HiMarket**, a hierarchical market world model for stock selection. Specifically, a hierarchical market-state construction module decomposes encoded stock representations into global, dynamic-group, and residual components. During pretraining, a market-conditioned residual prediction objective uses forecast shared market states to predict future stock residual representations. For downstream stock selection, a stock-conditioned factor-query readout pools encoded stock representations and their residual components over time, then projects their concatenation into stock-specific queries. These queries retrieve context from historical global and group memories. The pretrained encoder and hierarchy modules are jointly fine-tuned with this newly initialized readout under ranking supervision.

The contributions of this paper are summarized as follows:

- We introduce a hierarchical predictive framework with adaptive stock grouping that learns global, group-level, and residual representations through residual prediction conditioned on forecast shared market states.
- We design a stock-conditioned factor-query readout that forms queries from both encoded stock representations and their residual components to retrieve stock-relevant context from historical global and group memories, adapting the pretrained hierarchy to supervised stock ranking.
- We evaluate HiMarket on three real-world stock-market datasets. HiMarket achieves the highest investment return ratio on all three datasets and the highest Sharpe ratio on NASDAQ and NYSE among the evaluated methods.

Our implementation is available in an anonymous repository.

## 2 Related Work

### 2.1 Stock Selection

Stock-selection research models temporal patterns and relationships among stocks to predict future returns or rankings. Relational and hypergraph methods incorporate dependencies among stocks into ranking models (Feng et al., 2019b; Sawhney et al., 2021). More recent approaches integrate market context and evolving stock interactions. MASTER uses market information to guide feature selection and models temporal and cross-stock dependencies (Li et al., 2024). StockMixer exchanges information across indicator, temporal, and stock dimensions (Fan & Shen, 2024), while MDGNN models multiple types of stock relations and their temporal evolution through dynamic graphs (Qian et al., 2024). These studies highlight the importance of interpreting individual stock signals through their relationships with other stocks and the broader market.

### 2.2 Market State Modeling

Market-state modeling characterizes shared market conditions and their changes over time. Regime-discovery approaches identify recurring patterns across financial time series and model transitions between market regimes (Xu et al., 2024). Graph-based approaches describe market structure through stock-correlation networks. For example, graph auto-encoders use changes in network reconstruction quality to characterize market instability and support volatility forecasting (Gorduza et al., 2025). These approaches provide descriptions of market conditions beyond individual price trajectories.

Shared market structure can also be represented through latent factors and groups of related stocks. FactorVAE separates common latent factors from stock-specific components within a probabilistic model for return prediction (Duan et al., 2022). HIST distinguishes information shared through

predefined concepts, information shared through dynamically discovered concepts, and individual residual information (Xu et al., 2021). HireVAE connects market conditions with stock-level latent factors through a hierarchical model that adapts to changing market regimes (Wei et al., 2023). Together, these studies establish that shared structure and individual variation can be modeled jointly, and that market conditions can inform stock-level prediction.

### 2.3 Self-Supervised Learning

Self-supervised learning provides a way to learn representations beyond direct supervision from future stock returns. In stock selection, CI-STHPAN combines self-supervised time-series pretraining with dynamic hypergraph modeling (Xia et al., 2024). SSPT develops auxiliary tasks tailored to stock data, including stock code classification, sector classification, and moving-average prediction (Wang et al., 2025). These methods demonstrate the value of designing pretraining objectives around the characteristics of financial data.

Predictive learning differs in the targets used to train representations. Financial generative models predict market observations: Kronos models tokenized candlestick sequences (Shi et al., 2025), while MarS generates order-level activity for market simulation (Li et al., 2025a). Joint-embedding predictive architectures instead predict target embeddings rather than reconstructing raw observations. I-JEPA studies this approach for images (Assran et al., 2023), with temporal and video applications explored by TS-JEPA and V-JEPA 2 (Ennadir et al., 2024; Assran et al., 2025).

Most closely related to our work, Mahns et al. (2026) apply JEPA to daily equity cross-sections and learn market-state embeddings from sequences of factor tokens. Their evaluations show that these embeddings capture volatility and correlation structure and provide predictive information about some gradual market transitions. The study examines the content and dynamics of market representations, but does not evaluate cross-sectional stock selection.

HiMarket develops predictive pretraining for this downstream task. It predicts a hierarchical shared market state and uses the predicted state to condition the prediction of stock-specific residual representations. This objective directs stock-level learning toward differences beyond shared market movements while retaining the context needed to interpret them. The pretrained encoder is subsequently adapted to supervised stock ranking.

## 3 Preliminaries

### 3.1 Problem Definition

At decision time $t$, after the observations for trading day $t$ are available, let $\mathcal{U}_t$ denote the set of $N_t$ stocks with eligible historical observations. The input price–volume feature panel is

$$X_t = [x_{t-L+1}, \ldots, x_t] \in \mathbb{R}^{L \times N_t \times F}, \tag{1}$$

where $L$ is the lookback length and $F$ is the feature dimension. A stock-selection model produces scores $s_t = f_\omega(X_t) \in \mathbb{R}^{N_t}$, with higher scores indicating higher expected future returns. The return label for stock $i$ is

$$y_{t,i}^{(H)} = \frac{p_{t+H,i} - p_{t,i}}{p_{t,i}}, \tag{2}$$

where $p_{t,i}$ denotes the closing price and $H$ is the return horizon in trading days. We consider one-trading-day returns, with $H = 1$. The task is to rank eligible stocks by their future returns using only information available through day $t$. Stock eligibility depends on observed history; unavailable future labels are excluded from supervised losses and evaluation. These returns define prediction labels. Portfolio construction and execution assumptions are specified in the evaluation protocol.

### 3.2 Joint-Embedding Predictive Learning

Joint-embedding predictive architectures learn representations by predicting target embeddings from an observed context (Assran et al., 2023). In equity markets, Mahns et al. (2026) apply this principle to sequences of learned market-factor tokens.

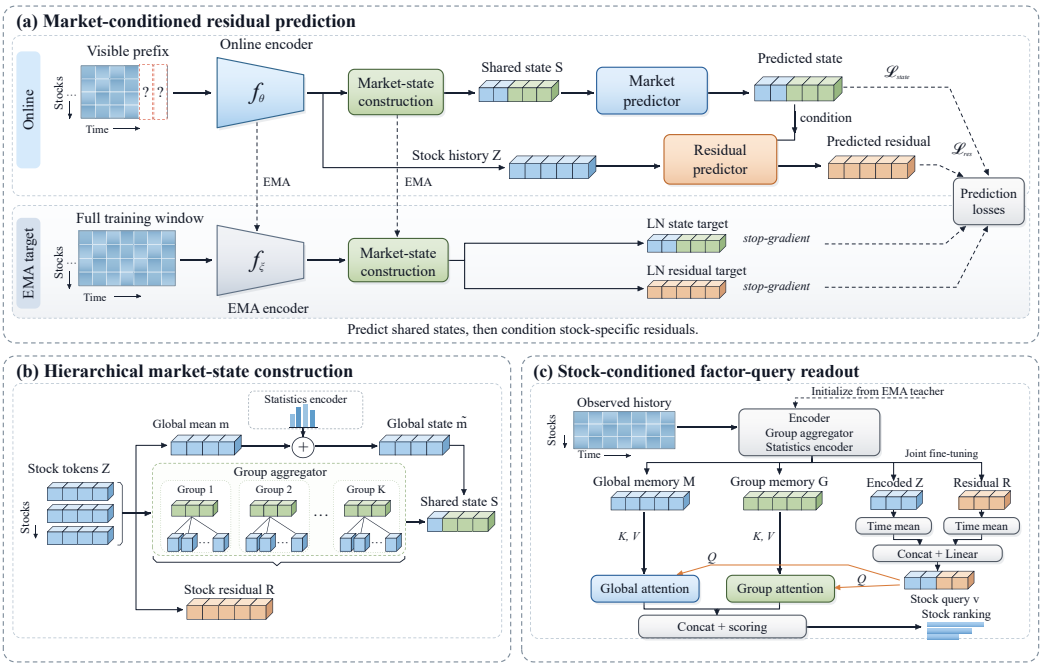



Figure 1: Overview of HiMarket. (a) Market-conditioned residual prediction first forecasts shared market states and then predicts future stock residuals conditioned on these forecasts, using targets from an EMA teacher. (b) Hierarchical market-state construction decomposes encoded stock representations into global, dynamic-group, and residual components. (c) Stock-conditioned factor-query readout forms stock queries by linearly projecting the concatenation of temporally pooled encoded representations and residuals. These queries retrieve historical global and group context for stock ranking.

Given a training window $X$, let $X_{\mathcal{C}}$ denote its visible context and $\mathcal{T}$ the hidden target positions. For the temporal setting considered here, the context is a prefix and the targets occupy a subsequent suffix. An online encoder $f_\theta$ and predictor $P_\psi$ produce target predictions, while a teacher encoder $f_{\bar{\theta}}$ provides training targets:

$$\begin{aligned} \widehat{Z}_{\mathcal{T}} &= P_\psi\big(f_\theta(X_{\mathcal{C}}), \mathcal{T}\big), \\ Z^*_{\mathcal{T}} &= \mathrm{sg}([f_{\bar{\theta}}(X)]_{\mathcal{T}}), \end{aligned} \tag{3}$$

where $[\cdot]_{\mathcal{T}}$ selects target positions and sg denotes stop-gradient. Training minimizes a discrepancy between predicted and target embeddings, and the teacher parameters follow an exponential moving average of the online parameters. The full window is used only to construct teacher targets; the online predictor receives only the visible context. HiMarket's target transformations and hierarchical prediction objectives are specified in the next section.

## 4 Method

HiMarket connects predictive market-state learning with stock selection through three components: hierarchical market-state construction, market-conditioned residual prediction, and a stock-conditioned factor-query readout (Figure 1). During pretraining, an online model predicts hidden suffix representations from a visible prefix, using an exponential-moving-average (EMA) teacher to construct targets. During supervised adaptation, the teacher's encoder and hierarchy initialize a trainable model whose new readout retrieves historical market context for each stock. Implementation details are provided in Appendix A.

### 4.1 HIERARCHICAL MARKET-STATE CONSTRUCTION

Given a price–volume panel, we divide each stock's $L$-day history into $P = L/\delta$ patches. An axial encoder alternates temporal attention within stocks and cross-sectional attention within patches to produce $\mathbf{z}_{q,i} \in \mathbb{R}^d$ for patch $q$ and stock $i$. For pretraining, $\mathcal{C} = \{1, \dots, c\}$ is the visible prefix and $\mathcal{T} = \{c+1, \dots, P\}$ is the hidden suffix. Online normalization and attention use only visible observations; attention is bidirectional within this view, with unavailable tokens masked.

**Global, group, and residual components.** For the valid stock set $\mathcal{V}_q$, the global mean and centered tokens are

$$\mathbf{m}_q = \frac{1}{|\mathcal{V}_q|} \sum_{i \in \mathcal{V}_q} \mathbf{z}_{q,i}, \qquad \mathbf{c}_{q,i} = \mathbf{z}_{q,i} - \mathbf{m}_q. \tag{4}$$

The $K$ learned group queries $\mathbf{u}_k$ score centered tokens by projected cosine similarity:

$$\ell_{q,k,i} = \cos(W_Q \mathbf{u}_k, W_K \mathbf{c}_{q,i})/\tau, \qquad \tau > 0, \tag{5}$$

where $\tau$ controls assignment sharpness. Alternating normalization produces soft assignments $a_{q,i,k}$ to $K$ groups, with $\sum_k a_{q,i,k} = 1$ and approximately balanced group usage. Assignments depend on the encoded panel and can change across patches and windows, without fixed sector labels. The group tokens and stock residuals are

$$\begin{aligned} \mathbf{g}_{q,k} &= \frac{\sum_{i \in \mathcal{V}_q} a_{q,i,k} \mathbf{c}_{q,i}}{\sum_{i \in \mathcal{V}_q} a_{q,i,k}}, \\ \mathbf{b}_{q,i} &= \sum_{k=1}^{K} a_{q,i,k} \mathbf{g}_{q,k}, \qquad \mathbf{r}_{q,i} = \mathbf{z}_{q,i} - \mathbf{m}_q - \mathbf{b}_{q,i}. \end{aligned} \tag{6}$$

Thus $\mathbf{z}_{q,i} = \mathbf{m}_q + \mathbf{b}_{q,i} + \mathbf{r}_{q,i}$ is an additive decomposition; it does not imply statistical independence or identified economic factors.

**Shared market state.** A statistics encoder maps observable return, breadth, and volume statistics to $\boldsymbol{\phi}_q \in \mathbb{R}^d$. The shared state is

$$\widetilde{\mathbf{m}}_q = \mathbf{m}_q + \boldsymbol{\phi}_q, \qquad \mathbf{S}_q = [\widetilde{\mathbf{m}}_q, \mathbf{g}_{q,1}, \dots, \mathbf{g}_{q,K}] \in \mathbb{R}^{(K+1) \times d}. \tag{7}$$

Statistics augment the global conditioning state; the residual in Equation 6 subtracts the unaugmented mean $\mathbf{m}_q$. Statistics use only observations available in the corresponding view (Appendix A.3).

### 4.2 MARKET-CONDITIONED RESIDUAL PREDICTION

We first forecast shared market states and then use them to condition stock-level residual prediction:

$$\begin{aligned} \widehat{\mathbf{S}}_{c+1:P} &= P_\psi^{\text{market}}(\mathbf{S}_{1:c}), \\ \widehat{\mathbf{r}}_{q,i} &= P_\psi^{\text{res}}(\mathbf{z}_{1:c,i}, \widehat{\mathbf{S}}_q, q), \qquad q \in \mathcal{T}. \end{aligned} \tag{8}$$

The market predictor uses time-slot queries to attend to visible shared states and predicts all suffix states in parallel. For stock $i$ and target patch $q \in \mathcal{T}$, the residual predictor constructs a query $\mathbf{q}_{q,i} \in \mathbb{R}^{d_p}$ from the encoded stock history, retrieves context from the predicted state at the same patch, and fuses the two:

$$\begin{aligned} \mathbf{q}_{q,i} &= P_\psi^{\text{local}}(\mathbf{z}_{1:c,i}, q), \\ \mathbf{a}_{q,i} &= \text{MHA}_{\text{cond}}(\mathbf{q}_{q,i}, W_M \widehat{\mathbf{S}}_q, W_M \widehat{\mathbf{S}}_q), \\ \widehat{\mathbf{r}}_{q,i} &= W_O(\mathbf{q}_{q,i} + \text{MLP}_{\text{fuse}}([\mathbf{q}_{q,i}; \mathbf{a}_{q,i}])) + \mathbf{b}_O. \end{aligned} \tag{9}$$

Here $\text{MHA}(Q, K, V)$ denotes multi-head attention, and $[\cdot;\cdot]$ denotes concatenation. The local query uses encoded stock tokens, whereas the prediction target is a residual. This separates shared-state evolution and stock-specific variation into distinct prediction targets while allowing residual dynamics to depend on market context. Residual-prediction gradients also update the market predictor, coupling shared-state learning to stock-level prediction. Additional predictor details appear in Appendix A.4.

**Pretraining objective.** Following JEPA (Assran et al., 2023), the EMA teacher contains the encoder, group aggregator, and statistics encoder. It encodes the full training window, normalizes suffix stock tokens before constructing their hierarchy, and normalizes the resulting residual and state tokens to form detached targets $\mathbf{r}^*_{q,i}$ and $\mathbf{S}^*_{q,k}$. The full window is used only for teacher targets. With token-wise Smooth-$L_1$ distance $D$, the prediction losses are

$$\begin{aligned}\mathcal{L}_{\text{res}} &= \left\langle D(\widehat{\mathbf{r}}_{q,i}, \mathbf{r}^*_{q,i})\right\rangle_{q\in\mathcal{T},\, i\in\mathcal{V}^*_q},\\ \mathcal{L}_{\text{state}} &= \left\langle D(\widehat{\mathbf{S}}_{q,k}, \mathbf{S}^*_{q,k})\right\rangle_{q\in\mathcal{T},\, k=0:K}.\end{aligned} \tag{10}$$

Here $\mathcal{V}^*_q$ includes stocks with valid target observations and visible history, $k = 0$ is the global slot, and averages include valid batch elements. The full objective is

$$\mathcal{L}_{\text{pre}} = \mathcal{L}_{\text{res}} + \lambda_{\text{state}}\mathcal{L}_{\text{state}} + \lambda_{\text{rel}}\mathcal{L}_{\text{rel}} + \lambda_{\text{div}}\mathcal{L}_{\text{div}}. \tag{11}$$

The relational term matches pairwise teacher residual similarities, while the diversity term discourages redundant online context group directions. Assignment balancing uses normalization rather than an additional loss. Teacher targets, auxiliary losses, and EMA updates are detailed in Appendices A.5–A.6.

### 4.3 Stock-Conditioned Factor-Query Readout

We initialize the encoder, group aggregator, and statistics encoder from the same EMA checkpoint selected by minimum validation pretraining loss. At decision date $t$, all $P$ input patches are observed history. We pool the encoded tokens and residuals over time and project their concatenation to form a stock-specific query:

$$\begin{aligned}\mathbf{z}^{\text{pool}}_i = \frac{1}{P}\sum_{q=1}^{P}\mathbf{z}_{q,i}, \qquad \mathbf{r}^{\text{pool}}_i = \frac{1}{P}\sum_{q=1}^{P}\mathbf{r}_{q,i},\\ \mathbf{v}_i = W_{\text{mix}}[\mathbf{z}^{\text{pool}}_i; \mathbf{r}^{\text{pool}}_i] + \mathbf{b}_{\text{mix}}, \qquad W_{\text{mix}} \in \mathbb{R}^{d\times 2d}.\end{aligned} \tag{12}$$

This query combines the full stock representation with its explicit residual component. The shared memories are

$$\mathbf{G} = [\mathbf{g}_{q,k}]_{q=1:P,\, k=1:K}, \qquad \mathbf{M} = [\widetilde{\mathbf{m}}_q]_{q=1:P}. \tag{13}$$

The group memory $\mathbf{G} \in \mathbb{R}^{PK\times d}$ and global memory $\mathbf{M} \in \mathbb{R}^{P\times d}$ retain historical shared context. Two separate attention branches enrich the combined stock query:

$$\mathbf{h}^B_i = \mathrm{LN}_B(\mathbf{v}_i + \mathrm{Attn}_B(\mathbf{v}_i, \mathbf{B})), \qquad B \in \{G, M\}. \tag{14}$$

Each $\mathrm{Attn}_B$ uses $\mathbf{v}_i$ as the query and $\mathbf{B}$ as both keys and values. The branches share an affine query normalization and have separate attention parameters and memory normalizations (Appendix A.7). The two outputs are concatenated and scored:

$$s_{t,i} = \mathbf{w}^\top \mathrm{LN}_{\text{out}}([\mathbf{h}^G_i; \mathbf{h}^M_i]) + b, \qquad \mathbf{w} \in \mathbb{R}^{2d}. \tag{15}$$

Stock-specific queries make retrieval depend on both encoded stock information and residual variation. This readout uses observed historical representations and states; it is distinct from the pretraining predictor, which uses encoded histories and forecast states.

**Supervised adaptation.** We jointly fine-tune the transferred modules and newly initialized readout using regression to centered within-date return ranks and a pairwise ordering loss:

$$\mathcal{L}_{\text{sel}}(t) = \mathrm{MSE}(\mathbf{s}_t, \widetilde{\mathbf{y}}_t) + \lambda_{\text{rank}}\mathcal{L}_{\text{pair}}(t). \tag{16}$$

Both terms use valid labels. The pairwise term penalizes reversed orderings in proportion to target rank separation; exact targets and reductions are given in Appendix A.8. Validation RankIC selects the downstream checkpoint. Downstream training uses neither EMA updates nor pretraining losses, and neither pretraining predictor is transferred or used at inference. All scores are computed from the observed historical panel.

Table 1: Dataset statistics and chronological splits.

| Dataset | Stocks | Training | Validation | Test |
|---|---|---|---|---|
| NASDAQ | 1,026 | 2013-02-19 –2015-12-31 | 2016-01-04 –2016-12-30 | 2017-01-03 –2017-12-11 |
| NYSE | 1,737 | 2013-02-19 –2015-12-31 | 2016-01-04 –2016-12-30 | 2017-01-03 –2017-12-08 |
| CSMD300 | 300 | 2021-04-07 –2024-03-14 | 2024-03-15 –2024-08-07 | 2024-08-08 –2024-12-31 |

# 5 EXPERIMENTS

Our evaluation examines stock-selection performance across markets, the contribution of the proposed designs.

## 5.1 EXPERIMENTAL SETUP

**Datasets.** Following SSPT (Wang et al., 2025), we evaluate stock selection using a daily Top-5 portfolio and report investment return ratio (IRR) and Sharpe ratio (SR) on the test split. We evaluate HiMarket on three publicly available stock-market datasets: NASDAQ, NYSE, and CSMD300. NASDAQ and NYSE are U.S. stock-selection benchmarks introduced by Feng et al. (2019b) and subsequently used by SSPT (Wang et al., 2025). CSMD300 comprises 300 constituent stocks of the CSI 300 index and is released as part of the CSMD benchmark (Liu et al., 2025). For all three datasets, we use daily open, high, low, and close prices, together with trading volume. Although CSMD300 also includes financial news, we use only its price–volume component.

The NASDAQ and NYSE data panels contain 1,026 and 1,737 stocks, respectively. Both start on November 19, 2012, and end on December 11, 2017 for NASDAQ and December 8, 2017 for NYSE. The CSMD300 panel covers January 4, 2021 to December 31, 2024. We partition each dataset chronologically into training, validation, and test sets, as summarized in Table 1. The reported split dates refer to target return dates; each input window contains only observations preceding its target date. Initial observations provide the history required to construct rolling features and input windows. The reported stock counts denote the total panel sizes, while availability masks determine the eligible stocks on each date.

**Baselines.** The baseline suite comprises 13 methods spanning four architectural families: (1) RNN-based models: SFM (Zhang et al., 2017), DA-RNN (Qin et al., 2017), and Adv-ALSTM (Feng et al., 2019a); (2) MLP-based models: TimeMixer (Wang et al., 2024) and StockMixer (Fan & Shen, 2024); (3) graph-/hypergraph-based models: RSR-I (Feng et al., 2019b), STHAN-SR (Sawhney et al., 2021), and CI-STHPAN (Xia et al., 2024); and (4) Transformer-based models: DTML (Yoo et al., 2021), ALSP-TF (Wang et al., 2022), MASTER (Li et al., 2024), TSPRank (Li et al., 2025b), and SSPT (Wang et al., 2025). SSPT is evaluated in its individual-task (SSPT-ind) and combined-task (SSPT-comb) configurations, yielding 14 baseline configurations in total. Hybrid models are grouped by their primary architectural components. We also include an equal-weight market portfolio as a reference.

**Evaluation Protocol.** We evaluate stock rankings using a daily rebalanced, equal-weight, long-only portfolio with a one-trading-day holding horizon ($H = 1$). For each target trading day $d$, predictions use observations only through the preceding trading day $d-1$. The evaluation universe $\mathcal{E}_d$ contains stocks with eligible histories, finite predictions, and valid one-day return labels. We select the $k_d = \min(5, |\mathcal{E}_d|)$ highest-scoring stocks, denoted by $\mathcal{A}_d$, and compute the portfolio return as

$$R_d = \frac{1}{k_d} \sum_{i \in \mathcal{A}_d} \left( \frac{p_{d,i}}{p_{d-1,i}} - 1 \right), \tag{17}$$

where $p_{d,i}$ is the closing price of stock $i$. Dates with fewer than two evaluable stocks are excluded. This is an idealized close-to-close benchmark with zero transaction costs; execution delays and slippage are not modeled.

We report Investment Return Ratio (IRR) and annualized Sharpe Ratio (SR). Over $T_{\text{eval}}$ evaluated trading days, they are defined as

$$\text{IRR} = \prod_{d=1}^{T_{\text{eval}}} (1 + R_d) - 1, \qquad \text{SR} = \sqrt{252}\,\frac{\overline{R}}{\sigma_R}, \tag{18}$$

where

$$\overline{R} = \frac{1}{T_{\text{eval}}} \sum_{d=1}^{T_{\text{eval}}} R_d, \qquad \sigma_R = \sqrt{\frac{1}{T_{\text{eval}}} \sum_{d=1}^{T_{\text{eval}}} (R_d - \overline{R})^2}. \tag{19}$$

IRR measures cumulative test-period return and is not annualized. SR assumes a zero risk-free return and uses the population standard deviation. Higher values indicate better performance for both metrics.

Downstream checkpoints are selected by validation RankIC, defined as the mean daily Spearman correlation between predicted scores and realized returns over evaluable stocks. Tied values receive average ranks, and undefined daily correlations are excluded from the average.

Table 2: Comparison of stock selection performance on NASDAQ, NYSE, and CSMD300. Higher IRR and SR indicate better performance. Best results are in **bold** and second best are underlined.

| Architecture | Method | NASDAQ | | NYSE | | CSMD300 | |
|---|---|---|---|---|---|---|---|
| | | IRR ↑ | SR ↑ | IRR ↑ | SR ↑ | IRR ↑ | SR ↑ |
| – | Market | 0.15 | 1.53 | 0.10 | 1.49 | 0.17 | 1.48 |
| RNN | SFM | 0.09 | 0.16 | 0.11 | 0.19 | 0.15 | 0.93 |
| | DA-RNN | 0.14 | 0.71 | 0.13 | 0.66 | 0.21 | 1.34 |
| | Adv-ALSTM | 0.23 | 0.97 | 0.14 | 0.81 | 0.26 | 1.64 |
| MLP | TimeMixer | 0.42 | 1.64 | 0.23 | 1.23 | -0.01 | -0.05 |
| | StockMixer | 0.20 | 1.40 | 0.54 | 1.57 | -0.11 | -0.37 |
| Graph | RSR-I | 0.39 | 1.34 | 0.21 | 0.95 | 0.16 | 1.53 |
| | STHAN-SR | 0.44 | 1.42 | 0.33 | 1.12 | 0.25 | 1.58 |
| | CI-STHPAN | 0.66 | 2.01 | <u>0.79</u> | 2.14 | 0.07 | 0.62 |
| Transformer | DTML | 0.41 | 1.35 | 0.45 | 1.17 | 0.21 | 1.32 |
| | ALSP-TF | 0.53 | 1.55 | 0.41 | 1.24 | 0.16 | 1.43 |
| | MASTER | 0.24 | 1.20 | 0.23 | 1.27 | <u>0.40</u> | **2.97** |
| | TSPRank | 0.29 | 1.43 | 0.28 | 1.74 | 0.24 | 1.75 |
| | SSPT-ind | 0.74 | <u>2.32</u> | 0.41 | 2.11 | 0.31 | 1.66 |
| | SSPT-comb | <u>0.82</u> | 2.25 | 0.60 | <u>2.35</u> | 0.20 | 1.07 |
| **HiMarket(ours)** | | **0.83** | **2.63** | **0.83** | **2.77** | **0.45** | <u>2.39</u> |

**Implementation Details.** HiMarket is implemented in PyTorch and trained on NVIDIA RTX 3080 Ti, RTX 3090, and RTX 5090 GPUs. We use 32-day input windows with four-day patches and mask the final two patches during pretraining. The encoder has two blocks, hidden dimension 128, and four attention heads; both predictors have depth two and width 64. We use three latent groups with assignment temperature 0.2. The market-state, relational, and diversity loss weights are 0.1, 0.1, and 0.05, respectively. Complete input-feature definitions, preprocessing procedures, and HiMarket training configurations are provided in Appendix B.

Both training stages use AdamW (Loshchilov & Hutter, 2019) with weight decay $10^{-4}$ and gradient clipping at 1.0. Pretraining runs for 100 epochs at learning rate $3 \times 10^{-4}$, with EMA momentum increasing linearly from 0.996 to 1.0. We initialize from the selected EMA teacher and jointly fine-tune the encoder, grouping module, statistics encoder, and readout. Fine-tuning uses learning rate $10^{-4}$ for NYSE and $3 \times 10^{-4}$ for NASDAQ and CSMD300, with early-stopping patience of 25 epochs after a minimum of 40 epochs. Pretraining and downstream checkpoints are selected by minimum validation pretraining loss and maximum validation RankIC, respectively.

### 5.2 Main Results

Table 2 reports stock-selection performance on NASDAQ, NYSE, and CSMD300. HiMarket achieves the highest IRR on all three datasets and the highest SR on NASDAQ and NYSE, ranking first in five of the six comparisons. On CSMD300, it achieves an IRR of 0.45, while its SR of 2.39 ranks second to MASTER's 2.97. HiMarket also outperforms the market portfolio on both metrics across all three datasets. On NASDAQ, the IRR improvement over SSPT-comb is modest (0.83 versus 0.82), whereas its SR exceeds the strongest baseline (2.63 versus 2.32).

These results show competitive performance across the three markets, although HiMarket does not dominate every metric. The ablation studies further examine the contributions of hierarchical market decomposition and the factor-query head.

Table 3: Ablation results. Each cell reports IRR / SR.

| Variant | NASDAQ | NYSE | CSMD300 |
|---|---|---|---|
| w/o Pre-training | 0.46 / 1.58 | 0.29 / 1.17 | 0.36 / 2.10 |
| Frozen Pretrained | 0.38 / 1.71 | -0.22 / -0.90 | 0.16 / 0.99 |
| Plain JEPA + Linear Readout | 0.38 / 1.29 | 0.16 / 0.71 | -0.10 / -0.76 |
| w/o Factor-Query Head | 0.55 / 1.71 | 0.62 / 1.85 | 0.00 / 0.15 |
| Full HiMarket | **0.83 / 2.63** | **0.83 / 2.77** | **0.45 / 2.39** |

**Ablation Analysis.** Table 3 shows that Full HiMarket achieves the highest IRR and SR across all three datasets. Removing pretraining consistently reduces both metrics, supporting the benefit of pretrained initialization. Freezing the pretrained modules also degrades performance, with NYSE IRR falling from 0.83 to -0.22, highlighting the importance of adapting the representations to downstream stock selection.

Removing the factor-query head lowers both metrics on every dataset. The effect is particularly pronounced on CSMD300, where IRR drops from 0.45 to 0.00 and SR from 2.39 to 0.15. This supports the role of the readout in adapting hierarchical representations for stock ranking. Plain JEPA with a linear readout also underperforms Full HiMarket across all datasets. Since this variant changes both pretraining and readout, the comparison assesses their combined contribution rather than isolating market decomposition.

## 6 Conclusion

We proposed HiMarket, a hierarchical market world model that connects predictive market-state learning with cross-sectional stock selection. The framework decomposes stock representations into global, dynamic-group, and residual components, and predicts future residual representations conditioned on forecast shared market states. A stock-conditioned factor-query readout retrieves relevant context from historical global and group memories, while joint fine-tuning adapts the pretrained representations to stock ranking. Experiments on NASDAQ, NYSE, and CSMD300 show that HiMarket achieves the highest IRR on all three datasets and the highest Sharpe ratio on NASDAQ and NYSE among the evaluated methods. Ablation studies further support the benefits of pretraining, joint fine-tuning, and the factor-query readout. These findings support hierarchical predictive learning as an effective approach to connecting shared market dynamics with the stock-specific distinctions required for selection.

## AI Use Statement

We used generative AI tools to assist with manuscript drafting and revision, LaTeX formatting, literature search, and checks of mathematical notation, tables, and cross-references. The authors reviewed the AI-assisted content and take full responsibility for the final manuscript, including its methods, experimental results, and conclusions.

## REFERENCES


Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023. URL https://arxiv.org/abs/2301.08243.

Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Yitong Duan, Lei Wang, Qizhong Zhang, and Jian Li. FactorVAE: A probabilistic dynamic factor model based on variational autoencoder for predicting cross-sectional stock returns. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4):4468–4476, 2022. doi: 10.1609/aaai.v36i4.20369.

Sofiane Ennadir, Siavash Golkar, and Leopoldo Sarra. Joint embeddings go temporal. In *NeurIPS 2024 Workshop on Time Series in the Age of Large Models*, 2024. URL https://openreview.net/forum?id=FIdbozebmy.

Jinyong Fan and Yanyan Shen. StockMixer: A simple yet strong MLP-based architecture for stock price forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8):8389–8397, 2024. doi: 10.1609/aaai.v38i8.28681.

Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. Enhancing stock movement prediction with adversarial training. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 5843–5849, 2019a. doi: 10.24963/ijcai.2019/810.

Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. Temporal relational ranking for stock prediction. *ACM Transactions on Information Systems*, 37(2):1–30, 2019b. doi: 10.1145/3309547.

Dragos Gorduza, Stefan Zohren, and Xiaowen Dong. Understanding stock market instability via graph auto-encoders. *EPJ Data Science*, 14:13, 2025. doi: 10.1140/epjds/s13688-025-00523-3.

Junjie Li, Yang Liu, Weiqing Liu, Shikai Fang, Lewen Wang, Chang Xu, and Jiang Bian. MarS: A financial market simulation engine powered by generative foundation model. In *International Conference on Learning Representations*, 2025a. URL https://arxiv.org/abs/2409.07486.

Tong Li, Zhaoyang Liu, Yanyan Shen, Xue Wang, Haokun Chen, and Sen Huang. MASTER: Market-guided stock transformer for stock price forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(1):162–170, 2024. doi: 10.1609/aaai.v38i1.27767.

Weixian Waylon Li, Yftah Ziser, Yifei Xie, Shay B. Cohen, and Tiejun Ma. TSPRank: Bridging pairwise and listwise methods with a bilinear travelling salesman model. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, pp. 707–718. ACM, 2025b. doi: 10.1145/3690624.3709234.

Yu Liu, Zhuoying Li, Ruifeng Yang, Fengran Mo, and Cen Chen. CSMD: Curated multimodal dataset for chinese stock analysis. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 6471–6475. ACM, 2025. doi: 10.1145/3746252.3761636. URL https://doi.org/10.1145/3746252.3761636.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Simon Mahns, Randall Balestriero, and Mahmoud Assran. Joint-embedding predictive learning of latent market states in u.s. equities. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306. PMLR, 2026. URL https://openreview.net/forum?id=BZfkxSasd3.

Hao Qian, Hongting Zhou, Qian Zhao, Hao Chen, Hongxiang Yao, Jingwei Wang, Ziqi Liu, Fei Yu, Zhiqiang Zhang, and Jun Zhou. MDGNN: Multi-relational dynamic graph neural network for comprehensive and dynamic stock investment prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 14642–14650, 2024. doi: 10.1609/aaai.v38i13.29381. URL https://ojs.aaai.org/index.php/AAAI/article/view/29381.

Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W. 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*, pp. 2627–2633, 2017. doi: 10.24963/ijcai.2017/366.

Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, Tyler Derr, and Rajiv Ratn Shah. Stock selection via spatiotemporal hypergraph attention network: A learning to rank approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1):497–504, 2021. doi: 10.1609/aaai.v35i1.16127.

Yu Shi, Zongliang Fu, Shuo Chen, Bohan Zhao, Wei Xu, Changshui Zhang, and Jian Li. Kronos: A foundation model for the language of financial markets. *arXiv preprint arXiv:2508.02739*, 2025. URL https://arxiv.org/abs/2508.02739.

Heyuan Wang, Tengjiao Wang, Shun Li, Jiayi Zheng, Shijie Guan, and Wei Chen. Adaptive long-short pattern transformer for stock investment selection. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pp. 3970–3977, 2022. doi: 10.24963/ijcai.2022/551.

Mengyu Wang, Tiejun Ma, and Shay B. Cohen. Pre-training time series models with stock data customization. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 3019–3030. ACM, 2025. doi: 10.1145/3711896.3737005.

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 *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7oLshfEIC2.

Zikai Wei, Anyi Rao, Bo Dai, and Dahua Lin. HireVAE: An online and adaptive factor model based on hierarchical and regime-switch VAE. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pp. 4903–4911, 2023. doi: 10.24963/ijcai.2023/545.

Hongjie Xia, Huijie Ao, Long Li, Yu Liu, Sen Liu, Guangnan Ye, and Hongfeng Chai. CI-STHPAN: Pre-trained attention network for stock selection with channel-independent spatio-temporal hypergraph. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8):9187–9195, 2024. doi: 10.1609/aaai.v38i8.28770.

Kunpeng Xu, Lifei Chen, Jean-Marc Patenaude, and Shengrui Wang. RHINE: A regime-switching model with nonlinear representation for discovering and forecasting regimes in financial markets. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*, pp. 526–534, 2024. doi: 10.1137/1.9781611978032.61.

Wentao Xu, Weiqing Liu, Lewen Wang, Yingce Xia, Jiang Bian, Jian Yin, and Tie-Yan Liu. HIST: A graph-based framework for stock trend forecasting via mining concept-oriented shared information. *arXiv preprint arXiv:2110.13716*, 2021. URL https://arxiv.org/abs/2110.13716.

Jaemin Yoo, Yejun Soun, Yong-chan 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*, pp. 2037–2045. ACM, 2021. doi: 10.1145/3447548.3467297.

Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. Stock price prediction via discovering multi-frequency trading patterns. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2141–2149. ACM, 2017. doi: 10.1145/3097983.3098117.

# A Additional Method Details

This appendix specifies the encoding, assignment, prediction, and optimization operations used by HiMarket. Pretraining predicts hidden suffix representations, whereas downstream stock selection reads the observed historical panel. The downstream model transfers the teacher's encoder, group aggregator, and statistic projection, and learns a new factor-query readout.

## A.1 Panel Encoding and Available Views

Each stock's history of length $L$ is partitioned into $P = L/\delta$ non-overlapping patches of $\delta$ days. A shared linear map embeds each patch, and a sinusoidal temporal position is added:

$$\mathbf{h}_{q,i}^{(0)} = W_{\text{emb}} \operatorname{vec}(\mathbf{X}_{q,i}) + \mathbf{b}_{\text{emb}} + \mathbf{e}_q \in \mathbb{R}^d. \tag{20}$$

An axial encoder alternates temporal self-attention within each stock and cross-sectional self-attention among stocks at each patch, producing encoded tokens $\mathbf{z}_{q,i}$. Attention blocks include residual feed-forward layers. No stock-index embedding is introduced. Hidden and unavailable tokens are masked; attention is bidirectional within the available view.

During pretraining, the online encoder observes the prefix $\mathcal{C} = \{1, \dots, c\}$, and the hidden suffix is $\mathcal{T} = \{c+1, \dots, P\}$. Online feature normalization uses only visible observations, so suffix values cannot enter the context through preprocessing or attention. The teacher encodes the full sampled training window solely to construct targets. During downstream stock selection, all patches in the input panel are observed by the decision date. The hidden pretraining suffix is future relative to the pretraining context boundary; it does not provide observations beyond a downstream decision date.

## A.2 Adaptive Group Assignment

Let $\mathcal{V}_q$ be the valid stock set, $\mathbf{m}_q$ its mean encoded token, and $\mathbf{c}_{q,i} = \mathbf{z}_{q,i} - \mathbf{m}_q$ the centered token. The $K$ learned group queries $\mathbf{u}_k$ score centered tokens by projected cosine similarity:

$$\ell_{q,k,i} = \cos(W_Q \mathbf{u}_k, W_K \mathbf{c}_{q,i})/\tau, \qquad \tau > 0. \tag{21}$$

Starting from exponentiated scores, alternating normalization rescales each group's total assignment toward $|\mathcal{V}_q|/K$ and each stock's total assignment to one. After a fixed number of iterations, the resulting weights $a_{q,i,k}$ satisfy $\sum_k a_{q,i,k} = 1$, with approximately balanced group usage. Assignment balancing is implemented by normalization; it does not add a separate loss term. Membership depends on the encoded panel and can change across windows and patches without fixed sector labels.

These weights define group tokens and stock-specific group contributions:

$$\mathbf{g}_{q,k} = \frac{\sum_{i \in \mathcal{V}_q} a_{q,i,k} \mathbf{c}_{q,i}}{\sum_{i \in \mathcal{V}_q} a_{q,i,k}}, \qquad \mathbf{b}_{q,i} = \sum_{k=1}^{K} a_{q,i,k} \mathbf{g}_{q,k}. \tag{22}$$

The stock residual is $\mathbf{r}_{q,i} = \mathbf{z}_{q,i} - \mathbf{m}_q - \mathbf{b}_{q,i}$. This is a learned additive decomposition and does not imply statistical independence or identification of economic factors.

## A.3 Observable Market Statistics

For each patch, we compute four statistics: the cross-sectional mean and population standard deviation of summed daily close-return features, signed breadth (the mean sign of these sums), and mean absolute daily volume change. All four use only available stock-days in the corresponding view. A two-layer GELU MLP followed by non-affine layer normalization maps the statistics, rescaled by fixed constants, to $\boldsymbol{\phi}_q \in \mathbb{R}^d$. The statistics augment the global conditioning token:

$$\widetilde{\mathbf{m}}_q = \mathbf{m}_q + \boldsymbol{\phi}_q, \qquad \mathbf{S}_q = [\widetilde{\mathbf{m}}_q, \mathbf{g}_{q,1}, \dots, \mathbf{g}_{q,K}] \in \mathbb{R}^{(K+1)\times d}. \tag{23}$$

The residual subtracts the unaugmented mean $\mathbf{m}_q$; $\boldsymbol{\phi}_q$ augments the conditioning state only.

### A.4 Pretraining Predictors

**Shared-state predictor.** The market predictor projects the visible states $\mathbf{S}_{1:c}$ into a $d_p$-dimensional memory with temporal positions and learned slot embeddings. Each target time-slot pair receives a learned query combined with its future position and slot embedding. Stacked cross-attention blocks read the visible memory, followed by normalization and an output map:

$$\widehat{\mathbf{S}}_{c+1:P} = P_\psi^{\text{market}}(\mathbf{S}_{1:c}) \in \mathbb{R}^{(P-c)\times(K+1)\times d}. \tag{24}$$

All suffix states are predicted in parallel. The outputs are latent representations rather than reconstructed prices or action-conditioned outcomes.

**Market-conditioned residual predictor.** For stock $i$ and target patch $q \in \mathcal{T}$, a future-position query attends to the stock's visible encoded history, producing $\mathbf{q}_{q,i} \in \mathbb{R}^{d_p}$. It then retrieves the predicted shared state at the same future patch and fuses that context:

$$\begin{aligned}
\mathbf{q}_{q,i} &= P_\psi^{\text{local}}(\mathbf{z}_{1:c,i}, q), \\
\mathbf{a}_{q,i} &= \text{MHA}_{\text{cond}}(\mathbf{q}_{q,i}, W_M\widehat{\mathbf{S}}_q, W_M\widehat{\mathbf{S}}_q), \\
\widehat{\mathbf{r}}_{q,i} &= W_O\big(\mathbf{q}_{q,i} + \text{MLP}_{\text{fuse}}([\mathbf{q}_{q,i}; \mathbf{a}_{q,i}])\big) + \mathbf{b}_O.
\end{aligned} \tag{25}$$

Here $\text{MHA}(Q, K, V)$ is multi-head attention and $[\cdot;\cdot]$ denotes concatenation. Local queries use encoded stock tokens, which already include cross-stock information from the axial encoder; their prediction targets are residuals. Residual-prediction gradients also update the market predictor.

### A.5 Teacher Targets and EMA Updates

The exponential-moving-average (EMA) teacher contains copies of the encoder, group aggregator, and statistic projection. Following joint-embedding predictive learning (Assran et al., 2023), it supplies stop-gradient targets. First, it encodes the full sampled training window. Second, it applies non-affine layer normalization to encoded suffix tokens and constructs the target hierarchy from these normalized tokens. Third, its residuals and shared-state tokens are normalized again:

$$\mathbf{r}^*_{q,i} = \text{sg}[\text{LN}_0(\overline{\mathbf{r}}_{q,i})], \qquad \mathbf{S}^*_{q,k} = \text{sg}[\text{LN}_0(\overline{\mathbf{S}}_{q,k})]. \tag{26}$$

Bars denote teacher quantities, sg stops gradients, and $\text{LN}_0$ normalizes each token without learned affine parameters. The full view is restricted to target construction and is not supplied to the online predictor.

Let $\theta$ collect the online encoder, aggregator, and statistic-projection parameters. Teacher parameters $\overline{\theta}$ begin as identical copies of the randomly initialized online parameters. After each optimizer step, they are updated as

$$\overline{\theta} \leftarrow \mu\overline{\theta} + (1-\mu)\theta, \tag{27}$$

with momentum $\mu$ increased toward one. Only online modules and predictors receive gradients, and the teacher remains in evaluation mode. The teacher's encoder, group aggregator, and statistic projection are retained together from the checkpoint with the lowest validation pretraining loss.

### A.6 Pretraining Losses and Validity Masks

The residual and shared-state prediction losses use the Smooth-$L_1$ distance

$$D(\mathbf{a}, \mathbf{b}) = \frac{1}{d}\sum_{j=1}^{d}\rho(a_j - b_j), \qquad \rho(u) = \begin{cases} u^2/2, & |u| < 1, \\ |u| - 1/2, & \text{otherwise.} \end{cases} \tag{28}$$

Specifically,

$$\begin{aligned}
\mathcal{L}_{\text{res}} &= \big\langle D(\widehat{\mathbf{r}}_{q,i}, \mathbf{r}^*_{q,i})\big\rangle_{q\in\mathcal{T},\, i\in\mathcal{V}^*_q}, \\
\mathcal{L}_{\text{state}} &= \big\langle D(\widehat{\mathbf{S}}_{q,k}, \mathbf{S}^*_{q,k})\big\rangle_{q\in\mathcal{T},\, k=0:K}.
\end{aligned} \tag{29}$$

Angle brackets average over valid entries, including batch elements. $\mathcal{V}_q^*$ contains stocks with both valid target observations and visible history. Shared-state targets require a nonempty valid stock set, and slot $k = 0$ denotes the global state.

Two auxiliary terms preserve teacher residual geometry over sampled valid stock pairs $\mathcal{P}_q$ and discourage redundant group directions:

$$\mathcal{L}_{\mathrm{rel}} = \Big\langle \rho\big(\cos(\widehat{\mathbf{r}}_{q,i}, \widehat{\mathbf{r}}_{q,j}) - \cos(\mathbf{r}^*_{q,i}, \mathbf{r}^*_{q,j})\big) \Big\rangle_{q\in\mathcal{T},\,(i,j)\in\mathcal{P}_q}, \tag{30}$$

$$\mathcal{L}_{\mathrm{div}} = \big\langle \cos^2(\mathbf{g}_{q,k}, \mathbf{g}_{q,l}) \big\rangle_{q\in\mathcal{C},\,k\neq l}. \tag{31}$$

Relation losses are averaged within each target patch and then across valid patches and batch elements. They supervise latent relationships rather than realized return ordering. Group diversity is evaluated on the online context hierarchy. The complete objective is

$$\mathcal{L}_{\mathrm{pre}} = \mathcal{L}_{\mathrm{res}} + \lambda_{\mathrm{state}}\mathcal{L}_{\mathrm{state}} + \lambda_{\mathrm{rel}}\mathcal{L}_{\mathrm{rel}} + \lambda_{\mathrm{div}}\mathcal{L}_{\mathrm{div}}. \tag{32}$$

### A.7 Factor-Query Readout

The downstream encoder, group aggregator, and statistic projection are initialized from the same teacher checkpoint. For decision date $t$, all $P$ patches of $\mathbf{X}_t$ are observable. Applying the transferred hierarchy gives encoded stock tokens and residuals, which are averaged over time patches and combined into a stock query:

$$\mathbf{z}_i^{\mathrm{pool}} = \frac{1}{P}\sum_{q=1}^{P}\mathbf{z}_{q,i}, \qquad \mathbf{r}_i^{\mathrm{pool}} = \frac{1}{P}\sum_{q=1}^{P}\mathbf{r}_{q,i},$$
$$\mathbf{v}_i = W_{\mathrm{mix}}[\mathbf{z}_i^{\mathrm{pool}}; \mathbf{r}_i^{\mathrm{pool}}] + \mathbf{b}_{\mathrm{mix}}, \qquad W_{\mathrm{mix}} \in \mathbb{R}^{d\times 2d}. \tag{33}$$

The projection has a learned bias $\mathbf{b}_{\mathrm{mix}} \in \mathbb{R}^d$. The two historical memories are

$$\mathbf{G} = [\mathbf{g}_{q,k}]_{q=1:P,\,k=1:K}, \qquad \mathbf{M} = [\widetilde{\mathbf{m}}_q]_{q=1:P}. \tag{34}$$

The group memory $\mathbf{G} \in \mathbb{R}^{PK\times d}$ retains all historical group tokens, and the global memory $\mathbf{M} \in \mathbb{R}^{P\times d}$ retains market-wide context. Here, factors denote the learned global and group components.

Two parallel attention branches use the same normalized stock query:

$$\mathbf{u}_i^G = \mathrm{MHA}_G(\mathrm{LN}_Q(\mathbf{v}_i), \mathrm{LN}_{Gm}(\mathbf{G}), \mathrm{LN}_{Gm}(\mathbf{G})),$$
$$\mathbf{u}_i^M = \mathrm{MHA}_M(\mathrm{LN}_Q(\mathbf{v}_i), \mathrm{LN}_{Mm}(\mathbf{M}), \mathrm{LN}_{Mm}(\mathbf{M})). \tag{35}$$

The branches have separate attention parameters and memory normalizations. Each adds its retrieved context to the combined stock query:

$$\mathbf{h}_i^G = \mathrm{LN}_G(\mathbf{v}_i + \mathbf{u}_i^G), \qquad \mathbf{h}_i^M = \mathrm{LN}_M(\mathbf{v}_i + \mathbf{u}_i^M). \tag{36}$$

Their outputs are concatenated and scored:

$$s_{t,i} = \mathbf{w}^\top \mathrm{LN}_{\mathrm{out}}([\mathbf{h}_i^G; \mathbf{h}_i^M]) + b, \qquad \mathbf{w} \in \mathbb{R}^{2d}. \tag{37}$$

All layer normalizations in the head have trainable affine parameters. The query projection, attention branches, and scoring layer are newly initialized. The readout uses combined historical stock queries and observed memories to produce ranking scores; it does not transfer either pretraining predictor. Although the memories are shared, attention weights depend on each stock's query.

### A.8 Supervised Fine-Tuning and Inference

The transferred modules and factor-query head are jointly fine-tuned using cross-sectional supervision. For the $n_t$ stocks with valid labels on date $t$, realized returns are transformed into centered rank targets:

$$\widetilde{y}_{t,i} = \frac{2\big(\mathrm{rank}(y_{t,i}^{(H)}) - 1/2\big)}{n_t} - 1. \tag{38}$$

Ranks run from one to $n_t$, using average ranks for ties. The supervised objective combines regression to the rank targets and a pairwise ordering loss:

$$\mathcal{L}_{\mathrm{sel}}(t) = \frac{1}{n_t}\sum_i (s_{t,i} - \widetilde{y}_{t,i})^2 + \frac{\lambda_{\mathrm{rank}}}{n_t^2}\sum_{i,j} \max(0, -(s_{t,i} - s_{t,j})(\widetilde{y}_{t,i} - \widetilde{y}_{t,j}))\,. \tag{39}$$

The sums include only valid labels, and the loss is averaged over training dates. The pairwise term penalizes reversed orderings in proportion to target rank separation. Validation RankIC selects the downstream checkpoint. Portfolio evaluation uses raw realized returns rather than transformed targets.

The downstream model is optimized directly, without EMA updates or pretraining losses. At inference, it reconstructs the historical hierarchy, queries the shared memories, and ranks eligible stocks by $s_{t,i}$. Neither pretraining predictor is required.

# B Additional Experimental Details

This section documents the HiMarket configuration used for the main results.

## B.1 Input Features and Preprocessing

Each stock is represented by 30 daily features. For each of the five OHLCV fields, we include its current value and trailing moving averages over 5, 10, 20, and 30 trading days, yielding 25 level features. Five additional features describe daily changes in the four price fields and trading volume.

Normalization is performed separately for each stock and input view. For each price field, its five level features are divided by the last observed value of that field within the view and shifted by $-1$. The five volume level features are divided by the mean daily volume within the view and shifted by $-1$. The additional price-change features are computed as $p_d/p_{d-1} - 1$ for each price field. The volume-change feature is $(v_d - v_{d-1})/\overline{v}$, where $\overline{v}$ is the mean volume within the view. The change features are set to zero at the first observed day. Volume features are set to zero when the volume normalization denominator is zero. All transformed features are clipped to $[-10, 10]$.

During pretraining, the online view is normalized using only the visible prefix, whereas the teacher view is normalized using the full training window. During downstream training and evaluation, normalization uses the complete observed historical window.

## B.2 Architecture and Pretraining

Table 4 summarizes the shared architecture and optimization settings. The 32-day window contains eight non-overlapping four-day patches. Pretraining exposes the first six patches to the online encoder and predicts targets for the final two patches.

The four market statistics—mean return, cross-sectional return dispersion, signed breadth, and volume change—are multiplied by 100, 100, 1, and 10, respectively, before entering the statistic projection network. These statistics augment the global conditioning token; they are not included in the mean subtracted when constructing stock residuals.

For the relation loss, we use all unordered valid stock pairs when their number does not exceed 4096. Otherwise, we sample 4096 non-self pairs with replacement. The loss is averaged over pairs within each target patch and then over valid patches and batch elements.

We use AdamW for both pretraining and downstream fine-tuning. Each optimization step processes one date-level panel, and training dates are shuffled each epoch. Pretraining validation is performed every five epochs. The checkpoint with the lowest validation pretraining loss supplies the EMA teacher encoder, group aggregator, and statistic projection for downstream initialization.

Table 4: HiMarket settings shared across the three datasets.

| Setting | Value |
|---|---|
| Historical window length | 32 trading days |
| Input feature dimension | 30 |
| Patch length | 4 trading days |
| Visible / target patches | 6 / 2 |
| Encoder dimension / depth | 128 / 2 |
| Encoder attention heads | 4 |
| Predictor dimension / depth | 64 / 2 |
| Number of dynamic groups $K$ | 3 |
| Assignment temperature $\tau$ | 0.2 |
| Assignment balancing iterations | 4 |
| State-loss weight $\lambda_{\mathrm{state}}$ | 0.1 |
| Relation-loss weight $\lambda_{\mathrm{rel}}$ | 0.1 |
| Diversity-loss weight $\lambda_{\mathrm{div}}$ | 0.05 |
| Ranking-loss weight $\lambda_{\mathrm{rank}}$ | 1.0 |
| Pretraining epochs | 100 |
| Pretraining learning rate | $3 \times 10^{-4}$ |
| Weight decay | $10^{-4}$ |
| Gradient clipping norm | 1.0 |
| Initial / final EMA momentum | 0.996 / 1.0 |
| Factor-query attention heads | 4 |
| Factor-query attention dropout | 0 |
| Batch size | One date-level stock panel |

### B.3 Downstream Training and Evaluation

All main-result runs use the Z+R factor-query readout. For stock $i$, the time-averaged encoded representation and hierarchical residual are concatenated and projected to form the query:

$$\mathbf{v}_i = W_{\mathrm{query}} \left[ \frac{1}{P} \sum_{q=1}^{P} \mathbf{z}_{q,i} \,;\, \frac{1}{P} \sum_{q=1}^{P} \mathbf{r}_{q,i} \right] + \mathbf{b}_{\mathrm{query}}. \tag{40}$$

The query projection maps $2d$ dimensions to $d$ dimensions. The group memory contains all $PK = 24$ historical group tokens, and the global memory contains all $P = 8$ historical global-state tokens. The transferred modules and newly initialized readout are jointly fine-tuned.

The downstream learning rate is $3\times10^{-4}$ for NASDAQ and CSMD300, and $10^{-4}$ for NYSE. We use a validation-driven learning-rate scheduler with a reduction factor of $0.5$, patience of eight epochs, and minimum learning rate of $10^{-6}$. Early stopping is enabled after 40 epochs, with patience of 25 epochs and a maximum budget of 400 epochs. An improvement in validation RankIC must exceed $10^{-5}$ to update the saved checkpoint. The downstream objective combines regression to centered return ranks with the pairwise ranking loss, using $\lambda_{\mathrm{rank}} = 1$.

The individual HiMarket runs reported in the main results use random seeds 3333, 1111, and 2222 for NASDAQ, NYSE, and CSMD300, respectively. These entries report individual-run results rather than averages across seeds. Within each run, downstream checkpoint selection uses validation RankIC. Test evaluation uses the daily Top-5 portfolio and reports IRR and SR computed from raw realized returns.