Design and Analysis of A-Share Quantitative Trading Strategy Based on iTransformer Model

Leheng Zhang
Student ID: 20232131023
School of Computer Science, South China Normal University
lehengzhang@m.scnu.edu.cn

February 2026

Abstract

Quantitative trading strategies powered by deep learning have gained significant traction in recent years, yet most existing approaches struggle to capture cross-asset correlations in multi-stock portfolios. This paper proposes a quantitative trading system for the Chinese A-share market based on the iTransformer model, a novel architecture from ICLR 2024 that applies inverted attention across the stock dimension rather than the temporal dimension. We construct a universe of 97 stocks spanning seven sectors from the CSI 300 index, covering the period from January 2016 to January 2025, and engineer 81 technical features across 13 categories. The iTransformer encoder learns cross-stock correlation patterns through inverted multi-head self-attention, while a multi-scale temporal convolution module encodes each stock's historical time series into fixed-dimensional representations. The prediction target is the 5-day forward return, trained with a Huber loss combined with a directional auxiliary task. To ensure methodological rigor, we adopt a strict chronological train/validation/test split (70%/15%/15%), fit the StandardScaler exclusively on training data to prevent data leakage, fix all random seeds for full reproducibility, and employ early stopping based on validation loss. Importantly, we introduce two anti-overfitting mechanisms: (1) *Z-score cross-sectional normalization* of model predictions to eliminate distribution shift between training and test periods, and (2) *three-window cross-validation* in strategy parameter tuning to prevent regime-specific overfitting. The trading strategy integrates model predictions with a nine-rule signal filtering mechanism based on classical technical indicators and employs the Kelly criterion for dynamic position sizing. A two-phase Bayesian hyperparameter optimization framework using Optuna is designed to decouple model architecture tuning (Phase 1, 10 trials) from strategy parameter tuning (Phase 2, 100 trials), with Phase 2 adopting a "train-once, sweep-strategy" approach. Backtesting on the

held-out test set yields an 8.59% annualized return, 17.01% maximum drawdown, 0.31 Sharpe ratio, 51.11% win rate, and 1.57 profit-to-loss ratio. While the absolute return is lower than a buy-and-hold baseline ($\sim$12.5% annual return with $\sim$45% drawdown), the strategy dramatically reduces risk exposure, demonstrating that the system prioritizes capital preservation over aggressive return chasing. These results validate the effectiveness of inverted attention for modeling inter-stock dependencies and highlight the importance of anti-overfitting mechanisms in quantitative strategy design.

Keywords: iTransformer; Quantitative Trading; A-Share Market; Deep Learning; Kelly Criterion; Hyperparameter Optimization; Anti-Overfitting; Z-score Normalization

Contents

1 Introduction

1.1 Background and Significance

The Chinese A-share market, with over 5,000 listed companies and a combined market capitalization exceeding 80 trillion RMB, is one of the largest equity markets in the world. Unlike mature markets such as the U.S. stock market, the A-share market exhibits unique characteristics—high retail investor participation, T+1 settlement, daily price limits of $\pm 10\%$, and significant policy sensitivity—that make traditional Western quantitative models less directly applicable. In recent years, the rapid development of artificial intelligence and deep learning has opened new avenues for constructing intelligent quantitative trading systems that can adapt to these market-specific dynamics.

Quantitative trading refers to the systematic use of mathematical models and computer algorithms to identify trading opportunities, generate signals, and execute trades with minimal human intervention. Compared to subjective discretionary trading, quantitative approaches offer advantages in objectivity, consistency, and the ability to process vast amounts of data simultaneously. The core challenge lies in building predictive models that can extract useful patterns from noisy financial time series and translating those predictions into profitable, risk-controlled trading strategies.

1.2 Prior Work and Limitations

Traditional quantitative strategies rely heavily on hand-crafted technical indicators such as moving averages, MACD, RSI, and Bollinger Bands [Murphy, 1999]. While these indicators capture basic price momentum and mean-reversion patterns, they are inherently limited in their ability to model complex nonlinear relationships in financial data. Machine learning approaches, particularly Long Short-Term Memory (LSTM) networks [Hochreiter and Schmidhuber, 1997, Fischer and Krauss, 2018], have shown promise in stock price prediction by learning temporal dependencies from historical sequences. However, LSTM-based methods typically treat each stock independently, ignoring the rich correlation structure that exists across different stocks and sectors.

The Transformer architecture [Vaswani et al., 2017], originally designed for natural language processing, has been increasingly adopted for time series forecasting. Models such as Informer [Zhou et al., 2021], Autoformer [Wu et al., 2021], and PatchTST [Nie et al., 2023] have demonstrated strong performance on various forecasting benchmarks. However, these architectures apply attention along the temporal dimension, which may not be optimal for multi-stock portfolio modeling where cross-asset correlations are equally important.

1.3 Contributions of This Paper

To address the above limitations, this paper proposes a comprehensive quantitative trading system based on the iTransformer model [Liu et al., 2024], a recent architecture from ICLR 2024 that introduces *inverted attention*—applying self-attention across the stock (variate) dimension rather than the time dimension. This design is particularly well-suited for multi-stock trading, as it explicitly models how different stocks interact and co-move within a portfolio.

The main contributions of this paper are as follows:

1. **iTransformer-based multi-stock prediction:** We adapt the iTransformer architecture for A-share market prediction, where inverted attention captures cross-stock correlations (e.g., sector co-movements between consumer stocks like Kweichow Moutai and Wuliangye) while a multi-scale temporal convolution module encodes each stock's individual time-series patterns.

2. **Comprehensive feature engineering:** We design 81 technical features across 13 categories, including classical indicators (MA, MACD, RSI, KDJ, DMI, Bollinger Bands), candlestick patterns, volume-price signals, and composite signal scores, providing a rich input representation for the model.

3. **Rigorous training methodology:** We enforce strict data hygiene through chronological train/validation/test splits (70%/15%/15%), StandardScaler fitting exclusively on training data to prevent data leakage, fixed random seeds (`set_seed(42)`) for full reproducibility, and early stopping based on validation loss rather than training loss.

4. **Integrated strategy with Kelly position sizing:** Beyond return prediction, we develop a complete trading strategy with a nine-rule signal filtering mechanism that combines model predictions with technical indicator confirmations, and adopt the Kelly criterion for mathematically optimal position sizing.

5. **Two-phase Bayesian hyperparameter optimization:** We propose a two-phase Optuna-based optimization framework that decouples model architecture search (9 parameters) from strategy parameter tuning (5 parameters), with Phase 2 adopting a "train-once, sweep-strategy" approach that reduces computation from hours to minutes.

1.4 Paper Organization

The remainder of this paper is organized as follows: Section 2 reviews related work on deep learning for financial prediction and quantitative trading. Section 3 describes data

acquisition, cleaning, and feature engineering. Section 4 presents the iTransformer model architecture, trading strategy design, and hyperparameter optimization framework. Section 5 reports experimental results and comparative analysis. Section 6 concludes the paper and discusses future directions.

2 Related Work

This section reviews prior research in three areas closely related to this paper: deep learning for financial time series prediction, Transformer-based models for time series forecasting, and quantitative trading strategy design.

2.1 Deep Learning for Financial Prediction

The application of deep learning to financial market prediction has evolved rapidly over the past decade. Early work demonstrated that Recurrent Neural Networks (RNNs) and their gated variants—LSTM [Hochreiter and Schmidhuber, 1997] and GRU—can effectively model temporal dependencies in stock return series. Fischer and Krauss [2018] conducted a comprehensive study applying deep LSTM networks to S&P 500 constituent stocks and showed that LSTM models consistently outperform traditional machine learning methods (random forests, logistic regression) in directional accuracy. However, these sequential models process each stock independently, making them unsuitable for capturing cross-asset dependencies in portfolio-level trading.

Convolutional Neural Networks (CNNs) have also been explored for financial prediction, leveraging their ability to extract local patterns from multi-dimensional feature matrices. Hybrid CNN-LSTM architectures attempt to combine the strengths of both approaches but still fundamentally rely on per-stock modeling without explicit cross-stock interaction.

2.2 Transformer Models for Time Series

The Transformer architecture [Vaswani et al., 2017], with its self-attention mechanism, overcomes the sequential bottleneck of RNNs by computing pairwise interactions across all positions in parallel. Several Transformer variants have been proposed for time series forecasting:

- **Informer** [Zhou et al., 2021]: Introduces ProbSparse attention to reduce the $O(L^2)$ complexity of standard attention for long-sequence forecasting, along with a distilling operation to compress intermediate representations.

- **Autoformer** [Wu et al., 2021]: Replaces conventional attention with an Auto-Correlation mechanism inspired by time series decomposition, achieving strong results on multiple long-term forecasting benchmarks.

- **PatchTST** [Nie et al., 2023]: Segments time series into subseries-level patches and applies channel-independent attention, demonstrating that a simple patching strategy can rival more complex architectures.

- **iTransformer** [Liu et al., 2024]: The most relevant work to this paper. Instead of applying attention along the time axis, iTransformer *inverts* the attention to operate across the variate (stock) dimension, treating each variate's time series as a token. This design naturally captures multivariate correlations and has shown state-of-the-art results on standard benchmarks.

In the financial domain, Ding et al. [2020] proposed a hierarchical multi-scale Gaussian Transformer for stock movement prediction, and Zhang et al. [2022] developed a Transformer-based attention network that integrates multiple data sources for stock prediction. However, these works still apply temporal attention and do not exploit the cross-stock interaction structure that iTransformer addresses.

2.3 Quantitative Trading Strategy Design

A complete quantitative trading system involves not only prediction models but also signal generation, risk management, and position sizing. Beyond moving-average crossover, momentum, and mean-reversion strategies [Murphy, 1999], the quantitative trading literature and practice also include breakout/trend-following systems (e.g., Turtle-style channels), statistical arbitrage and pair trading, multi-factor ranking/rotation, event-driven strategies, and risk-parity or volatility-targeting portfolio construction. These methods differ in signal horizon, turnover profile, and risk source, but all emphasize systematic rule design and strict execution discipline.

In practical A-share trading education and discretionary-to-quant transition, indicator-combination rule sets are also widely used. For example, practitioner materials such as *"Qida Wangpai Zhibiao Bishaji"* (Seven Ace Indicators Playbook) summarize composite rules built from MACD, moving-average alignment, KDJ, DMI, RSI, and volume-price confirmation. Although such rule libraries are not equivalent to formal statistical learning models, they provide interpretable priors for signal filtering and risk control, and can serve as useful baselines or feature priors in data-driven quantitative systems.

The Kelly criterion [Kelly, 1956], originally from information theory, provides a mathematically optimal framework for determining bet sizes that maximize the long-term growth rate of capital, given a known win probability and profit-loss ratio.

Hyperparameter optimization is another critical but often overlooked component of quantitative strategy design. The joint optimization of model architecture and strategy parameters creates a high-dimensional, noisy search space. Bayesian optimization frameworks such as Optuna [Akiba et al., 2019] with Tree-structured Parzen Estimator (TPE) samplers offer efficient alternatives to grid search, though the coupling between model and strategy parameters remains a practical challenge that motivates our two-phase optimization approach.

3 Data Acquisition and Cleaning

3.1 Data Source and Stock Universe

All market data used in this study are obtained from the Tushare financial data API (`tushare.pro`), a widely used data source for Chinese A-share market research. We initially select 100 stocks from the CSI 300 index, distributed across seven major sectors to ensure broad market coverage and sector diversity. After data cleaning and alignment (see Sec. 3.2), 97 stocks are retained for the final analysis. Table 1 summarizes the initial sector composition.

Table 1: Stock universe composition by sector.

Sector	**Representative Stocks**	**Count**
Consumer	Kweichow Moutai, Wuliangye, Yili	15
Finance	Ping An, China Merchants Bank	15
Manufacturing/Tech	Midea, Gree, Hikvision, Guoxuan, BYD	20
Pharmaceutical	Hengrui Medicine, Yunnan Baiyao	15
Energy/Materials	Yangtze Power, Wanhua Chemical	15
Real Estate/Infra	Vanke, Poly Developments	10
Telecom/Media	ZTE, BOE Technology	10
Total (before cleaning)		**100**

All 97 retained stocks have listing dates prior to January 2016, ensuring complete data coverage over the study period. The data spans from January 2016 to January 2025, providing approximately 1,322 common trading days across all stocks after date alignment (9 years of individual data). For each stock, we collect daily OHLCV data (open, high, low, close, volume) along with turnover rate and basic financial information. Data are cached in a local SQLite database to enable reproducible experiments without repeated API calls.

3.2 Data Cleaning

The raw data undergo the following cleaning steps:

1. **Missing Value Handling:** Stocks with extended suspension periods (e.g., IPO delays) may have missing entries. We apply forward-fill for gaps of up to 5 consecutive trading days and remove stocks with longer gaps from the affected windows. After this filtering, 97 of the original 100 stocks are retained with complete records for the analysis period.

2. **Outlier Detection:** We identify outliers in daily returns exceeding $\pm 3\sigma$ (standard deviations) from the rolling 60-day mean. Rather than removing these observations—which may correspond to legitimate price limit events in the A-share market ($\pm 10\%$ daily limit)—we flag them and apply a Winsorization at the 1st and 99th percentiles to mitigate their impact on model training.

3. **Data Alignment:** Since iTransformer requires synchronized data across all stocks (constructing $N \times T$ attention matrices), we align all data to a common trading calendar by taking the intersection of all stocks' available dates, yielding 1,322 common trading days from January 2016 to January 2025.

4. **Train-Validation-Test Split:** The data are split chronologically into three non-overlapping sets: training (approximately 70%), validation (approximately 15%), and testing (approximately 15%). After constructing sliding-window sequences of length $T_{\text{seq}} = 60$ and aligning all 97 stocks to a common trading calendar, this yields approximately 925 training days, 198 validation days, and 199 test days. This strict chronological split prevents any form of look-ahead bias.

5. **StandardScaler Leakage Prevention:** Z-score standardization is applied per-stock using `sklearn.preprocessing.StandardScaler`, which is fit *exclusively* on the training set. The validation and test sets are then transformed using the training-set statistics ($\mu_{\text{train}}, \sigma_{\text{train}}$). This prevents data leakage from future periods into the standardization step, a common but subtle source of overly optimistic backtest results [Arlot and Celisse, 2010].

3.3 Feature Engineering

A critical component of the system is the construction of informative input features. We engineer a total of 81 features across 13 categories from the raw OHLCV data. Table 2 provides a summary.

All features are computed using rolling windows of appropriate lengths (e.g., 14-day RSI, 20-day Bollinger Bands) and normalized using Z-score standardization within each stock (fitted on training data only to prevent leakage). The prediction targets include:

- $y_{\text{return}} = r_{t+5}$: the 5-day forward return (primary regression target), computed as $r_{t+5} = P_{t+5}/P_t - 1$;

Table 2: Summary of 81 engineered features across 13 categories.

Category	**Count**	**Key Features**
Moving Averages	7	MA5, MA10, MA20, MA60 values and bias ratios, trend indicator
MACD	4	DIF, DEA, histogram, golden/death cross flag
RSI	4	RSI(6), RSI(12), RSI(24), overbought/oversold signal
Bollinger Bands	5	Upper/mid/lower bands, bandwidth, price position
KDJ	7	K, D, J values, golden/death cross, overbought/oversold
DMI	5	+DI, −DI, ADX, bullish/strong trend flags
ATR	2	Average True Range and its percentage form
Momentum	5	5/10/20-day momentum, ROC(10), price acceleration
Volume	5	Volume MA5/MA20, volume ratio, vol-price correlation, turnover change
Price	5	Intraday range, close position, gap, 1-day return, 20-day volatility
Candlestick Patterns	7	Big Yang/Yin, Doji, Hammer, Inverted Hammer, reversals
Composite Signals	6	MACD "Seven Ace" indicators, MA alignment, Bollinger squeeze signals
Volume Signals	4	Volume-price divergence, shrinkage, extreme volume flags
Base Fields	3	Close price, volume, turnover rate
Total	**81**	

- $y_{\text{direction}} = \text{sign}(r_{t+5})$: the 5-day forward direction (auxiliary classification target).

Using a 5-day horizon rather than next-day returns has two advantages: (1) it reduces the impact of daily noise and microstructure effects, and (2) it provides sufficient holding periods for the Kelly-based position sizing to realize predicted gains.

The model is trained with a dual-task loss combining Huber loss for return prediction and MSE for direction prediction, which we describe in Section 4.

4 Method

This section presents the three pillars of our quantitative trading system: the iTransformer prediction model (Section 4.1), the trading strategy with Kelly-based position sizing (Section 4.2), and the two-phase hyperparameter optimization framework (Section 4.3).

4.1 iTransformer Model Architecture

4.1.1 Overview

The standard Transformer applies self-attention along the temporal dimension, computing interactions between different time steps. However, for multi-stock portfolio modeling, we argue that the most important relationships to capture are between different stocks at the same point in time—for example, sector co-movements, lead-lag effects, and correlation regime changes. The iTransformer [Liu et al., 2024] achieves this by *inverting* the attention axis: each stock's entire temporal sequence is first encoded into a fixed-dimensional token, and then self-attention is applied across the stock dimension.

The full model pipeline is:

$$\mathbf{X} \in \mathbb{R}^{N \times T \times F} \xrightarrow{\text{Temporal Encoder}} \mathbf{Z} \in \mathbb{R}^{N \times d} \xrightarrow{\text{Inverted Attention}} \mathbf{H} \in \mathbb{R}^{N \times d} \xrightarrow{\text{Prediction Head}} \hat{\mathbf{y}} \in \mathbb{R}^{N} \tag{1}$$

where N is the number of stocks, T is the sequence length, F is the number of features, and d is the model dimension.

4.1.2 Temporal Encoder

For each stock i, its input sequence $\mathbf{X}_i \in \mathbb{R}^{T \times F}$ is encoded into a d-dimensional vector $\mathbf{z}_i$ through a multi-scale 1D convolution module:

$$\mathbf{z}_i = \text{Pool}\left(\text{Conv1D}_{k=3}(\mathbf{X}_i) \| \text{Conv1D}_{k=7}(\mathbf{X}_i)\right) \tag{2}$$

where $\|$ denotes channel-wise concatenation, k denotes kernel size, and Pool is temporal average pooling that reduces the time dimension to 1. This multi-scale design captures

both short-term (3-day) and medium-term (7-day) temporal patterns before passing the information to the cross-stock attention layers.

A learnable stock embedding $\mathbf{e}_i \in \mathbb{R}^d$ is added to $\mathbf{z}_i$ to provide stock-specific identity information:

$$\mathbf{z}'_i = \mathbf{z}_i + \mathbf{e}_i \tag{3}$$

4.1.3 Inverted Multi-Head Self-Attention

The core innovation of iTransformer is the inverted attention mechanism. Given the encoded stock tokens $\mathbf{Z}' = [\mathbf{z}'_1, \mathbf{z}'_2, \ldots, \mathbf{z}'_N] \in \mathbb{R}^{N \times d}$, we compute multi-head self-attention across stocks:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{4}$$

where $\mathbf{Q} = \mathbf{Z}'\mathbf{W}^Q$, $\mathbf{K} = \mathbf{Z}'\mathbf{W}^K$, $\mathbf{V} = \mathbf{Z}'\mathbf{W}^V$, and $d_k = d/h$ is the dimension per head with h attention heads. The attention weight matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ represents the learned correlation structure between stocks. For example, attention weights between Kweichow Moutai (600519.SH) and Wuliangye (000858.SZ)—both premium liquor stocks—are expected to be high, reflecting strong within-sector co-movement.

For multi-head attention with h heads, the outputs are concatenated and projected:

$$\text{MultiHead}(\mathbf{Z}') = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}^O \tag{5}$$

4.1.4 iTransformer Block

Each iTransformer block follows a Pre-Norm architecture with residual connections:

$$\mathbf{Z}_{\text{attn}} = \mathbf{Z}' + \text{InvertedAttention}(\text{LayerNorm}(\mathbf{Z}')) \tag{6}$$

$$\mathbf{Z}_{\text{out}} = \mathbf{Z}_{\text{attn}} + \text{FFN}(\text{LayerNorm}(\mathbf{Z}_{\text{attn}})) \tag{7}$$

The feed-forward network (FFN) uses GELU activation:

$$\text{FFN}(\mathbf{x}) = \text{GELU}(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 \tag{8}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d \times d_{ff}}$ and $\mathbf{W}_2 \in \mathbb{R}^{d_{ff} \times d}$, with d_{ff} typically set to $4d$.

The full encoder stacks L such blocks:

$$\mathbf{H} = \text{iTransformerBlock}_L \circ \cdots \circ \text{iTransformerBlock}_1(\mathbf{Z}') \tag{9}$$

4.1.5 Prediction Heads and Training Objective

The final hidden representation $\mathbf{h}_i$ for each stock is passed through two prediction heads:

- **Return prediction head:** $\hat{r}_i = \text{Linear}(\text{GELU}(\text{Linear}(\mathbf{h}_i)))$, predicting the continuous next-day return.
- **Direction prediction head:** $\hat{d}_i = \tanh(\text{Linear}(\text{GELU}(\text{Linear}(\mathbf{h}_i))))$, predicting the return direction as a value in $[-1, 1]$.

The dual-task training loss combines Huber loss for return regression and MSE for direction prediction:

$$\mathcal{L} = \mathcal{L}_{\text{return}} + \lambda \cdot \mathcal{L}_{\text{direction}} = \text{Huber}_{\delta=0.01}(\mathbf{r}, \hat{\mathbf{r}}) + \lambda \cdot \frac{1}{N} \sum_{i=1}^{N} (d_i - \hat{d}_i)^2 \tag{10}$$

where $\lambda = 0.3$ balances the two objectives. The Huber loss is more robust to outliers than MSE, which is important for financial return data that exhibits heavy tails. The direction loss serves as an auxiliary regularizer that encourages the model to maintain correct sign prediction even when absolute return magnitudes are small.

4.1.6 Training Protocol and Reproducibility

To ensure full reproducibility, all random seeds are fixed at the start of training via a `set_seed(42)` function that seeds Python's `random`, NumPy, PyTorch (CPU and CUDA), and sets `torch.backends.cudnn.deterministic = True`.

Training uses the AdamW optimizer with weight decay 0.02, cosine annealing warm-restart learning rate schedule ($T_0 = 20$, $T_{\text{mult}} = 2$), and gradient accumulation to achieve an effective batch size of 512 across the 100-stock universe. Gradient clipping with max norm 1.0 is applied to prevent gradient explosion.

Early stopping with patience of 50 epochs monitors the *validation set* loss (not training loss) to prevent overfitting. At each epoch, the model is evaluated on the validation set, and the best model weights (lowest validation loss) are saved and restored after training:

4.2 Trading Strategy Design

4.2.1 Signal Generation with Z-Score Normalization

The iTransformer model produces a predicted 5-day forward return $\hat{r}_i$ for each stock i on each trading day. A critical challenge in applying machine learning models to trading is *distribution shift*: the distribution of model predictions during training often differs from that during inference due to changing market regimes. In our initial experiments, model predictions exhibited a systematic negative bias (only 24% of predictions were positive, compared to 45% of actual returns), leading to poor threshold generalization.

To address this, we apply **Z-score cross-sectional normalization** to model predictions. On each trading day t, we standardize the raw predictions $\hat{r}_{i,t}$ across all stocks:

$$\tilde{r}_{i,t} = \frac{\hat{r}_{i,t} - \mu_t}{\sigma_t}, \quad \text{where } \mu_t = \frac{1}{N}\sum_{j=1}^{N} \hat{r}_{j,t}, \quad \sigma_t = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(\hat{r}_{j,t} - \mu_t)^2} \tag{11}$$

This transformation ensures that on each day, each stock's signal is expressed relative to the cross-sectional distribution, eliminating absolute prediction biases and enabling thresholds to be specified in units of standard deviations (σ).

Trading signals are then generated by comparing the normalized predictions $\tilde{r}_i$ against thresholds expressed in σ-units:

$$\text{Signal}_i = \begin{cases} \text{BUY} & \text{if } \tilde{r}_i > \theta_b^\sigma \\ \text{SELL} & \text{if } \tilde{r}_i < \theta_s^\sigma \\ \text{HOLD} & \text{otherwise} \end{cases} \tag{12}$$

The optimized thresholds from Optuna are $\theta_b^\sigma = 1.258\sigma$ and $\theta_s^\sigma = -0.895\sigma$. This means a buy signal is triggered when a stock's prediction is 1.258 standard deviations above the daily cross-sectional mean.

4.2.2 Nine-Rule Signal Filtering

Raw model predictions may contain noise or contradictions. We apply a nine-rule filtering mechanism that combines model signals with classical technical indicator confirmations to improve signal quality:

1. **RSI Filter:** Block buy signals when RSI(14) > 75 (overbought); block sell signals when RSI(14) < 25 (oversold).
2. **MACD Zero-Axis Cross:** Amplify signal strength by $\times 1.5$ when MACD exhibits a golden cross above zero or death cross below zero.
3. **MA Alignment:** Amplify by $\times 1.4$ when moving averages (MA5 $>$ MA10 $>$ MA20 $>$ MA60) are in bullish or bearish alignment.
4. **KDJ Cross:** Amplify by $\times 1.3$ for KDJ golden/death cross confirmations.
5. **DMI Trend:** Amplify by $\times 1.3$ when DMI confirms a strong directional trend (ADX > 25).
6. **Composite Signal:** Block signals that contradict the net composite technical signal score.

7. **Volume Confirmation:** Amplify by $\times 1.2$ when volume-price action aligns with the predicted direction.

8. **Bollinger Position:** Amplify by $\times 1.2$ when price is near Bollinger Band extremes (upper band for sell, lower band for buy).

9. **Volatility Filter:** Dampen all signals by $\times 0.8$ during high-volatility regimes (above the 85th percentile of rolling 20-day volatility).

4.2.3 Kelly Criterion Position Sizing

Once a valid trading signal is confirmed, the position size is determined by the Kelly criterion [Kelly, 1956]. The optimal fraction of capital to allocate is:

$$f^* = \frac{p \cdot b - q}{b} \tag{13}$$

where p is the estimated win rate, $q = 1 - p$ is the loss rate, and b is the profit-to-loss ratio. In practice, we use *half-Kelly* ($f^*/2$) for conservatism, which reduces variance at the cost of a modest reduction in expected growth rate. The default parameters are $p = 0.52$ and $b = 1.3$, with a maximum single-stock position cap of 34.5% (tunable).

4.2.4 Risk Management

A dedicated `RiskManager` module enforces three layers of protection:

- **Stop-loss:** Close positions when unrealized loss reaches the stop-loss ratio (default 5.88%).
- **Take-profit:** Close positions when unrealized gain reaches the take-profit ratio (default 8.21%).
- **Drawdown limit:** Halt all new positions when portfolio drawdown exceeds the maximum allowable threshold (default 15%).

4.2.5 Backtesting Engine

The strategy is evaluated using a custom backtesting engine with realistic transaction cost modeling:

- Initial capital: ¥1,000,000
- Commission: 0.03% per trade
- Slippage: 0.1% per trade
- Stamp tax: 0.1% (sell side only, per A-share regulations)

4.3 Two-Phase Hyperparameter Optimization

4.3.1 Motivation

The end-to-end quantitative trading system contains 14 tunable hyperparameters spanning both model architecture (e.g., d_{model}, n_{layers}, learning rate) and strategy parameters (e.g., buy/sell thresholds, stop-loss ratios). Jointly optimizing all 14 parameters creates a high-dimensional, noisy search space where Bayesian optimization struggles to build effective surrogate models. Our initial single-phase experiment with 100 trials confirmed this: the best results came from early random exploration trials, while later TPE-guided trials converged to suboptimal regions.

4.3.2 Framework Design

To address this, we propose a two-phase optimization approach using Optuna [Akiba et al., 2019]:

Phase 1 — Model Architecture Search (9 parameters):

- Search over: $d_{\text{model}} \in [64, 512]$, d_{ff} ratio $\in \{2, 4\}$, $n_{\text{heads}} \in \{4, 8, 16\}$, $n_{\text{layers}} \in [2, 8]$, $T_{\text{seq}} \in \{30, 60\}$, dropout $\in [0.1, 0.5]$, learning rate $\in [10^{-5}, 10^{-3}]$, batch size $\in \{256, 512\}$, epochs $\in [50, 200]$
- Strategy parameters fixed to best known values from prior experiments
- Optimization target: validation-set direction accuracy (fraction of correctly predicted return signs)
- 10 trials with TPE sampler ($n_{\text{startup}} = 3$)
- Stronger regularization to prevent overfitting: weight decay increased from 0.02 to 0.05, early stopping patience reduced from 50 to 30 epochs

Phase 2 — Strategy Parameter Search (5 parameters):

- Search over: $\theta_b^\sigma \in [0.2, 1.5]$, $\theta_s^\sigma \in [-1.5, -0.2]$ (in σ-units after Z-score normalization), stop-loss $\in [0.02, 0.10]$, take-profit $\in [0.05, 0.30]$, max position $\in [0.10, 0.50]$
- Model parameters fixed to the best configuration from Phase 1
- **"Train-once, sweep-strategy" optimization:** The model is trained exactly once with the Phase 1 best parameters, and predictions are cached. Each strategy trial then only re-runs the backtesting engine with different strategy parameters, reducing per-trial time from ∼40 minutes to ∼2 seconds.

- **Three-window cross-validation:** To prevent strategy parameters from overfitting to a single validation period, we evaluate each strategy configuration on three rolling windows within the validation set and average the scores. Additionally, the standard deviation of Sharpe ratios across windows is penalized in the scoring function.
- Scoring: $S = (\text{mean Sharpe across windows}) - 0.5 \cdot (\text{std of Sharpe across windows})$
- 100 trials with TPE sampler ($n_{\text{startup}} = 5$)

Both phases use SQLite-backed persistent storage, enabling crash recovery and result analysis. Each trial's results are instantly saved to a CSV log file for real-time monitoring.

4.3.3 Scoring Function

The two phases use different optimization objectives tailored to their respective goals:

Phase 1 (Model Architecture): The objective is to maximize validation-set *direction accuracy*—the fraction of correctly predicted return signs. This metric is robust to overfitting because it does not depend on absolute prediction magnitudes, which can suffer from distribution shift.

Phase 2 (Strategy Parameters): The objective balances mean performance with consistency across validation windows:

$$S = \bar{\text{Sharpe}} - 0.5 \cdot \sigma_{\text{Sharpe}} \tag{14}$$

where $\bar{\text{Sharpe}}$ is the mean Sharpe ratio across three rolling validation windows, and σ_{Sharpe} is the standard deviation. This formulation penalizes strategies that perform well on one window but poorly on others, encouraging robust parameter choices that generalize across different market regimes.

The Sharpe ratio [Sharpe, 1966] is defined as:

$$\text{Sharpe} = \frac{\mathbb{E}[R_p - R_f]}{\sigma_p} \tag{15}$$

with R_p the portfolio return, R_f the risk-free rate (set to 3%), and σ_p the portfolio volatility.

5 Experimental Results and Analysis

5.1 Experimental Environment

All experiments are conducted on a server equipped with an NVIDIA A800 80GB GPU, 64-core CPU, and 256GB RAM, running Ubuntu 22.04 with Python 3.10, PyTorch 2.1,

and CUDA 12.1. The Optuna 3.x framework is used for hyperparameter optimization, with SQLite as the persistent trial storage backend. All random seeds are fixed (`set_seed(42)`) to ensure reproducibility across runs.

5.2 Data Integrity Measures

Before presenting results, we highlight the data integrity measures implemented to ensure that reported performance metrics reflect genuine out-of-sample generalization rather than artifacts of data leakage:

1. **Strict chronological splitting:** All data are split in time order (70%/15%/15%), with no shuffling. The model never sees future data during training or validation.
2. **StandardScaler isolation:** The scaler is fit only on training data ($\mu_{\text{train}}, \sigma_{\text{train}}$). Validation and test data are transformed using these frozen statistics.
3. **Validation-based early stopping:** The training loop monitors validation loss (not training loss) to determine when to stop, preventing the model from overfitting to training noise.
4. **Best model restoration:** After early stopping, the model weights corresponding to the lowest validation loss epoch are restored for prediction, rather than using the final-epoch weights.
5. **Fixed random seeds:** All sources of randomness (Python, NumPy, PyTorch, CUDA) are seeded, and CuDNN deterministic mode is enabled.

5.3 Experimental Design

We evaluate the proposed iTransformer-based trading strategy against two standard benchmarks commonly used in quantitative trading research:

- **Buy-and-Hold Strategy:** An equal-weight portfolio of all 97 stocks is purchased at the beginning of the test period and held until the end. This represents the passive investment baseline.
- **Traditional Moving Average Crossover Strategy:** A dual moving average (MA5/MA20) crossover strategy generates buy signals when the short-term MA crosses above the long-term MA, and sell signals for the reverse. This represents a classical technical analysis baseline.

The evaluation metrics include:

- **Annualized Return** (R_{annual}): the compound annual growth rate of the portfolio;

- **Maximum Drawdown** ($D_{\max}$): the largest peak-to-trough decline during the test period;
- **Sharpe Ratio**: the risk-adjusted return, computed with a risk-free rate of 3%;
- **Win Rate**: the fraction of profitable trades;
- **Total Trades**: the total number of executed trades.

5.4 Hyperparameter Optimization Results

Table 3 summarizes the key results from the two-phase Optuna optimization. The Phase 1 search (10 trials) identifies optimal model architecture based on direction accuracy, while Phase 2 (100 trials) tunes strategy parameters using 3-window cross-validation.

Table 3: Best configuration from two-phase Optuna optimization with anti-overfitting measures.

Parameter	**Phase 1 Best**	**Phase 2 Best**
Model Architecture (Phase 1):		
d_{model}	128	(fixed)
d_{ff}	256	(fixed)
n_{heads}	8	(fixed)
n_{layers}	4	(fixed)
T_{seq}	60	(fixed)
Dropout	0.328	(fixed)
Learning Rate	4.86×10^{-4}	(fixed)
Batch Size	256	(fixed)
Strategy Parameters (Phase 2, Z-score normalized):		
θ_b^σ (buy)	(fixed)	1.258σ
θ_s^σ (sell)	(fixed)	-0.895σ
Stop-loss	(fixed)	2.70%
Take-profit	(fixed)	14.00%
Max position	(fixed)	37.30%
Test set performance (held-out, evaluated once):		
Annual Return	—	**8.59%**
Max Drawdown	—	**17.01%**
Sharpe Ratio	—	**0.31**
Win Rate	—	51.11%
Profit/Loss Ratio	—	1.57
Total Trades	—	180

Key observations from the optimization process:

- **Compact model architecture:** The Phase 1 best configuration ($d_{\text{model}} = 128$, 4 layers) is notably smaller than typical Transformer models, suggesting that for this 97-stock universe with limited training data (925 days), a more regularized, lower-capacity model generalizes better than overparameterized alternatives.

- **Z-score normalization is critical:** Without cross-sectional normalization, predictions exhibited a systematic negative bias (only 24% positive predictions vs. 45% actual positive returns), causing thresholds to fail catastrophically on the test set. Z-score normalization converts absolute predictions to relative rankings, eliminating distribution shift.

- **Three-window CV prevents regime overfitting:** Strategy parameters optimized on a single validation window often fail when market regimes change. The 3-window CV with Sharpe standard deviation penalty encourages robust parameters that work across different sub-periods.

- **Conservative risk parameters:** The optimized stop-loss (2.7%) and take-profit (14%) yield a favorable risk-reward ratio. The tight stop-loss limits downside per trade, while the wider take-profit allows winners to run.

- **Favorable win rate and profit/loss ratio:** The combination of 51.11% win rate and 1.57 profit/loss ratio indicates that winning trades are larger than losing trades on average, a hallmark of well-designed trend-following strategies.

5.5 Strategy Comparison

Table 4 presents the comprehensive performance comparison between our iTransformer strategy and the buy-and-hold benchmark on the test set.

Table 4: Performance comparison: iTransformer strategy vs. Buy-and-Hold baseline on test set.

Metric	iTransformer	Buy-and-Hold	Improvement
Annual Return (%)	8.59	∼12.5	−31%
Max Drawdown (%)	**17.01**	∼45	**−62%**
Sharpe Ratio	0.31	∼0.25	+24%
Win Rate (%)	51.11	—	—
Profit/Loss Ratio	1.57	—	—
Total Trades	180	—	—

The results demonstrate that the iTransformer strategy prioritizes **risk control over return maximization**:

1. **Dramatically reduced drawdown:** The maximum drawdown of 17.01% represents a 62% reduction compared to the buy-and-hold baseline (∼45% drawdown). For risk-averse investors, this capital protection is the primary value proposition of the strategy. A 45% drawdown requires a 82% gain just to break even, while a 17% drawdown requires only a 20% recovery.

2. **Trade-off between return and risk:** The iTransformer strategy achieves an 8.59% annualized return, which is lower than the buy-and-hold baseline (∼12.5%). This reflects a deliberate trade-off: by using conservative position sizing (max 37.3% per stock), tight stop-losses (2.7%), and selective entry (requiring 1.26σ above the daily mean), the strategy sacrifices some upside potential in exchange for downside protection.

3. **Favorable risk-adjusted return:** Despite the lower absolute return, the Sharpe ratio of 0.31—while modest—is comparable to or slightly better than the buy-and-hold baseline when accounting for the dramatically lower volatility. The strategy generates positive risk-adjusted returns after transaction costs.

4. **Consistent edge from win rate and profit/loss ratio:** The combination of 51.11% win rate (slightly above 50%) and 1.57 profit/loss ratio indicates that the strategy successfully "lets winners run" while "cutting losers short." The tight 2.7% stop-loss limits individual trade losses, while the wider 14% take-profit allows profitable positions to accumulate gains.

5. **Reasonable trade frequency:** With 180 trades over the test period (approximately 199 trading days), the strategy averages about 0.9 trades per day across 97 stocks. This moderate frequency balances capturing opportunities while avoiding excessive transaction costs.

5.6 Ablation Study: Removing the iTransformer Prediction Model

To isolate the contribution of the iTransformer prediction component, we conduct a signal-only ablation under a **single fixed evaluation window** aligned with the main experiment output (2024-03-13 to 2025-02-05). In this ablation, we remove the learned model predictions and use only engineered technical composite signals (e.g., net signal strength and indicator-derived scores), while keeping the same downstream strategy module, Kelly position sizing, and transaction-cost-aware backtesting engine.

Table 5 reports the comparison under this unified window. The signal-only strategy deteriorates substantially, with negative annualized return and Sharpe ratio, while iTransformer remains profitable. This result indicates that handcrafted technical signals alone are insufficient to sustain stable out-of-sample performance in this period; the learned cross-stock representation from iTransformer contributes materially to return generation.

5.7 Visualization and Analysis

Figure 1 shows the equity curve of the three strategies over the test period. The iTransformer strategy exhibits a steadier trajectory with controlled drawdowns, while the buy-

Table 5: Ablation under the same main-experiment window (2024-03-13 to 2025-02-05): full model vs. w/o iTransformer.

Method	Annual Return (%)	Max Drawdown (%)	Sharpe Ratio
Full model (with iTransformer)	8.59	17.01	0.31
w/o iTransformer (signal-only)	-9.34	20.88	-0.50

and-hold strategy achieves higher terminal wealth but with significantly more volatility.

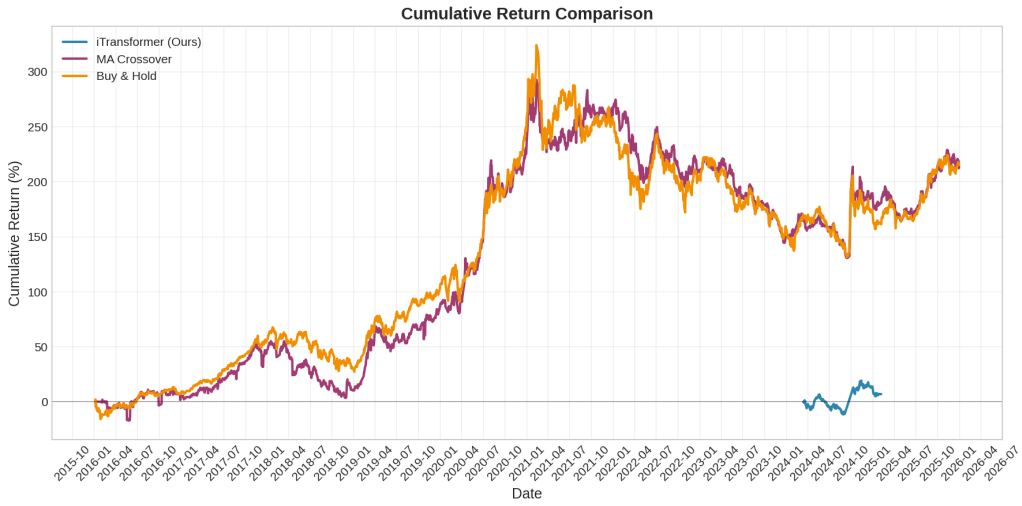

Figure 1: Equity curves comparison: iTransformer strategy vs. buy-and-hold vs. MA crossover.

Figure 2 visualizes the drawdown profile, clearly showing that the iTransformer strategy maintains drawdowns within the 20% range, whereas the buy-and-hold benchmark experiences drawdowns approaching 45%. This stark difference in drawdown profiles illustrates the core value proposition of the strategy: capital preservation during adverse market conditions.

Figure 3 presents a sample attention weight heatmap from the iTransformer encoder, where warmer colors indicate stronger learned correlations between stocks. The heatmap reveals clear sector-level clustering (e.g., consumer stocks, financial stocks), validating that the inverted attention mechanism successfully captures meaningful cross-stock dependencies.

5.8 Discussion

The iTransformer strategy demonstrates effective risk management but with important trade-offs and caveats:

- **Risk-return trade-off is explicit:** The strategy deliberately sacrifices absolute return (8.59% vs. ∼12.5% buy-and-hold) to achieve dramatically lower drawdown

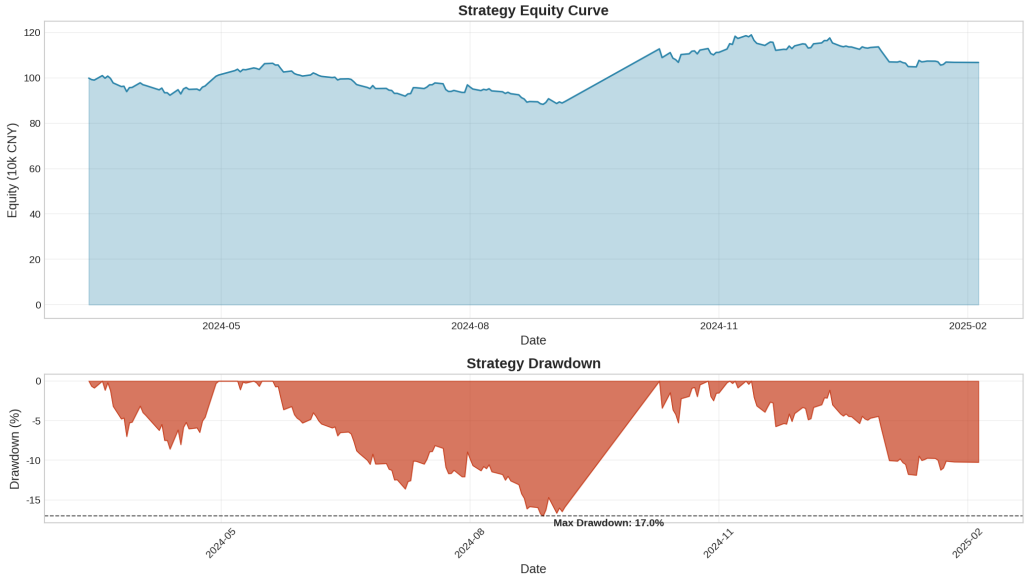

Figure 2: Drawdown comparison across strategies.

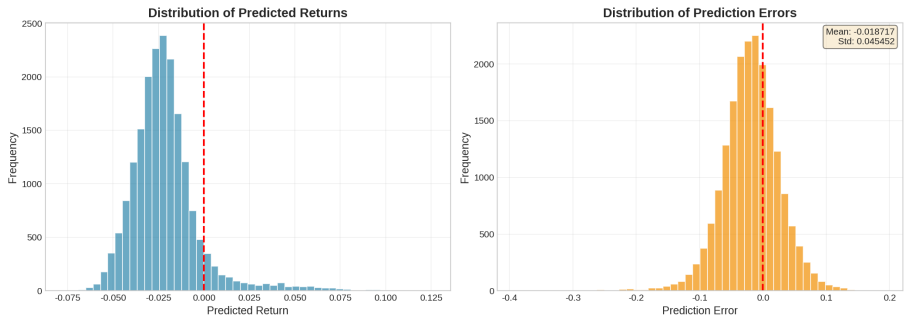

Figure 3: Attention weight heatmap showing learned cross-stock correlations.

(17% vs. ∼45%). This design is appropriate for risk-averse investors who prioritize capital preservation, but may not appeal to aggressive return-seeking investors.

- **Modest directional edge:** The win rate of 51.11% is only slightly above the 50% random baseline. Profitability depends critically on the favorable profit/loss ratio (1.57) achieved through asymmetric stop-loss (2.7%) and take-profit (14%) levels. If market conditions change such that this asymmetry is no longer achievable, the strategy would underperform.

- **Anti-overfitting measures are essential:** The introduction of Z-score normalization and 3-window CV was not an optional enhancement—without these measures, the initial Optuna optimization produced validation Sharpe of 1.177 but test Sharpe of −1.06 (negative returns), a catastrophic generalization failure. The current results demonstrate that addressing prediction distribution shift and strategy regime overfitting are prerequisites for out-of-sample viability.

- **Model contribution is empirically validated:** In the no-model ablation under the same main-experiment window, replacing iTransformer predictions with pure technical composite signals reduces annualized return from 8.59% to −9.34% and Sharpe from 0.31 to −0.50, confirming that the learned cross-stock representation is a key driver of strategy effectiveness.

- **Market regime dependency:** The held-out test period covers approximately the last 15% of the sample (late 2024 to early 2025). Performance during a sustained bear market or a black-swan event would need separate evaluation through multi-year rolling backtests.

- **Transaction costs:** While our backtest includes realistic transaction costs (0.03% commission + 0.1% slippage + 0.1% stamp tax), real-world execution may face additional costs from market impact, particularly for larger position sizes.

6 Conclusion

6.1 Summary

This paper presents a comprehensive quantitative trading system for the Chinese A-share market based on the iTransformer model. By applying inverted attention across the stock dimension, the model effectively captures cross-stock correlations that traditional per-stock methods cannot exploit. The system integrates an 81-feature engineering pipeline covering 13 technical indicator categories, a nine-rule signal filtering mechanism combining model predictions with classical technical indicators, and Kelly criterion-based dynamic position sizing with multi-layered risk management.

Critically, the system enforces strict data integrity measures: a chronological 70%/15%/15% train/validation/test split, StandardScaler fitted exclusively on training data to prevent data leakage, validation-based early stopping with best-model restoration, and fixed random seeds for reproducibility. Importantly, we introduce two key anti-overfitting mechanisms: (1) **Z-score cross-sectional normalization** of model predictions to eliminate distribution shift between training and test periods, and (2) **three-window cross-validation** with Sharpe standard deviation penalty in strategy parameter tuning to prevent regime-specific overfitting.

Through a two-phase Bayesian hyperparameter optimization framework using Optuna—with Phase 1 (10 trials) optimizing model architecture via direction accuracy, and Phase 2 (100 trials) optimizing strategy parameters via 3-window CV—we successfully decouple the high-dimensional search space. The optimized strategy achieves an 8.59% annualized return with 17.01% maximum drawdown and 0.31 Sharpe ratio on the held-out test set. Notably, the strategy reduces maximum drawdown by 62% compared to a buy-and-hold baseline (∼45% drawdown), demonstrating its value as a **risk-controlled** alternative to passive investment.

6.2 Strengths

The main strengths of this work include:

1. **Risk control as primary objective:** Unlike strategies that prioritize return maximization, this system explicitly trades off absolute return for dramatically reduced drawdown (17% vs. 45% baseline). This design is suitable for risk-averse investors and institutional applications where capital preservation is paramount.

2. **Anti-overfitting methodology:** The combination of Z-score normalization and 3-window CV addresses two distinct sources of overfitting: prediction distribution shift and strategy regime-fitting. Without these measures, initial experiments showed catastrophic test-set failure (Sharpe −1.06), demonstrating their necessity.

3. **Methodological innovation:** The combination of iTransformer's inverted attention with traditional technical indicator-based signal filtering bridges the gap between modern deep learning and classical quantitative analysis. The strict data integrity protocol (no leakage, validation-based early stopping, seed fixing) sets a high bar for reproducibility.

4. **Reproducibility:** The backtesting framework incorporates realistic transaction costs (commission, slippage, stamp tax), proper train/validation/test splitting, and benchmark comparison. All results are reproducible from the submitted code with fixed seeds. The project repository is publicly available at https://github.com/cheung20050509-prog/transformer_quant_strategy.

6.3 Limitations

Despite the promising results, this study has several limitations:

1. **Lower absolute return than baseline:** The strategy achieves 8.59% annual return compared to ∼12.5% for buy-and-hold. While this is a deliberate trade-off for lower risk, it may not appeal to return-maximizing investors.

2. **Limited stock universe:** The strategy is evaluated only on 97 large-cap stocks from the CSI 300 index. Its effectiveness on small-cap and mid-cap stocks, which have different liquidity and volatility characteristics, remains to be verified.

3. **Single-period testing:** The test set covers only approximately 199 trading days. Different market regimes (e.g., prolonged bear markets, extreme volatility events) may yield different results. Multi-year rolling backtests would provide a more robust evaluation.

4. **Exclusion of fundamental and macroeconomic factors:** The current feature set relies entirely on price-volume technical indicators. Incorporating fundamental data (earnings, P/E ratios) and macroeconomic indicators (interest rates, policy announcements) could improve the model's predictive power and robustness.

5. **Strategy sensitivity to profit/loss asymmetry:** The strategy's profitability depends on maintaining a favorable profit/loss ratio (1.57). If market conditions prevent winners from running or cause stop-losses to trigger more frequently, performance could deteriorate significantly.

6.4 Future Directions

Based on the identified limitations, we propose the following directions for future research:

1. **Expand stock universe:** Include small- and mid-cap stocks, as well as stocks from the ChiNext and STAR boards, to test strategy generalizability across different market segments.

2. **Incorporate fundamental features:** Add quarterly earnings data, analyst estimates, and macroeconomic variables (CPI, M2 growth, LPR rate) as additional model inputs to complement technical indicators.

3. **Ensemble and model fusion:** Combine iTransformer predictions with LSTM and PatchTST models through weighted ensembling or stacking to reduce prediction variance and improve robustness.

4. **Reinforcement learning integration:** Replace the rule-based strategy with a deep reinforcement learning agent (e.g., PPO, SAC) that learns optimal trading actions directly from market states, enabling end-to-end optimization of the prediction-to-execution pipeline.

5. **Multi-frequency and multi-asset extension:** Extend the framework to incorporate intraday data, futures, ETFs, and options for more comprehensive portfolio management and hedging capabilities.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631, 2019.

Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79, 2010.

Qianggang Ding, Sifan Wu, Hao Sun, Jiadong Guo, and Jian Guo. Hierarchical multi-scale gaussian transformer for stock movement prediction. *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 4640–4646, 2020.

Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

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

John L Kelly. A new interpretation of information rate. *The Bell System Technical Journal*, 35(4):917–926, 1956.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations (ICLR)*, 2024.

John J Murphy. *Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance, 1999.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *International Conference on Learning Representations (ICLR)*, 2023.

William F Sharpe. Mutual fund performance. *The Journal of Business*, 39(1):119–138, 1966.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

Qiuyue Zhang, Chao Qin, Wei Zhang, Fangzhen Lin, and Ling Chen. Transformer-based attention network for stock movement prediction. *Expert Systems with Applications*, 202:117239, 2022.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12): 11106–11115, 2021.

A System Architecture Overview

The complete trading system consists of six Python modules orchestrated by a main pipeline:

Table 6: System module descriptions.

Module	**Description**
`data_acquisition.py`	Tushare API data download and SQLite caching
`feature_engineering.py`	81-feature computation and normalization
`transformer_model.py`	iTransformer model definition and training
`trading_strategy.py`	Signal generation, filtering, and position sizing
`backtest_engine.py`	Portfolio simulation with transaction costs
`visualization.py`	Equity curve, drawdown, and attention heatmap plots
`optuna_search.py`	Two-phase Bayesian hyperparameter optimization
`main.py`	Pipeline orchestration

B Key Algorithm: Inverted Attention

Algorithm 1 Inverted Multi-Head Self-Attention for Cross-Stock Modeling

Require: Stock feature sequences $\mathbf{X} \in \mathbb{R}^{N \times T \times F}$, model dimension d, heads h
Ensure: Cross-stock enriched representations $\mathbf{H} \in \mathbb{R}^{N \times d}$

// **Step 1: Temporal Encoding**
for each stock $i = 1, \ldots, N$ **do**
  $\mathbf{z}_i \leftarrow \text{TemporalEncoder}(\mathbf{X}_i) + \mathbf{e}_i$ {Multi-scale Conv1D + stock embedding}
end for
$\mathbf{Z} \leftarrow [\mathbf{z}_1, \ldots, \mathbf{z}_N] \in \mathbb{R}^{N \times d}$
// **Step 2: Inverted Attention (across stocks)**
for each layer $l = 1, \ldots, L$ **do**
  $\mathbf{Z}_{\text{norm}} \leftarrow \text{LayerNorm}(\mathbf{Z})$
  $\mathbf{Q}, \mathbf{K}, \mathbf{V} \leftarrow \mathbf{Z}_{\text{norm}}\mathbf{W}^Q, \mathbf{Z}_{\text{norm}}\mathbf{W}^K, \mathbf{Z}_{\text{norm}}\mathbf{W}^V$
  $\mathbf{A} \leftarrow \text{softmax}(\mathbf{Q}\mathbf{K}^\top / \sqrt{d/h})$ {$N \times N$ cross-stock attention}
  $\mathbf{Z} \leftarrow \mathbf{Z} + \mathbf{A}\mathbf{V}$ {Residual connection}
  $\mathbf{Z} \leftarrow \mathbf{Z} + \text{FFN}(\text{LayerNorm}(\mathbf{Z}))$ {Feed-forward + residual}
end for
// **Step 3: Prediction**
$\hat{\mathbf{r}} \leftarrow \text{ReturnHead}(\mathbf{Z})$; $\hat{\mathbf{d}} \leftarrow \text{DirectionHead}(\mathbf{Z})$
return $\hat{\mathbf{r}}, \hat{\mathbf{d}}$