InterDiff: Synthesizing Financial Time Series with Inter-Stock Correlations via Classifier-Free Guided Diffusion

Hou-Wan Long[1], Zhoufei Tang[2], Jianhui Zhang[2], Zhuoyang Zhan[2], Tao Lu[3], and Xiaoquan (Michael) Zhang[4] (✉)

[1] Department of Statistics, The Chinese University of Hong Kong, Hong Kong
houwanlong@link.cuhk.edu.hk

[2] Super Quantum Capital Management, China
{tangzhoufei, jianhui, zhanzhuoyang}@superquant.fund

[3] College of Business, Southern University of Science and Technology, China
lut@sustech.edu.cn

[4] Department of Management Science and Engineering, Tsinghua University, China
zhangxiaoquan@sem.tsinghua.edu.cn

Abstract. Stock prediction is hindered by data scarcity, and although existing data augmentation techniques have made significant strides, they often overlook the dynamic inter-stock interactions crucial for robust modeling. To address these challenges, we propose **InterDiff**, a diffusion-based framework that synthesizes realistic financial time series by dynamically modeling both intra- and inter-stock correlations. InterDiff employs hierarchical transformers to learn these correlations, encoding them into a guidance vector that steers a diffusion model via classifier-free guidance. This approach ensures that the synthetic data preserves fidelity while introducing controlled variability. Evaluations on CSI300 and CSI800 show that models trained on InterDiff-augmented data boost the information coefficient by 1.13–4.70% on CSI300 and 40.15–49.60% on CSI800, while delivering cumulative return improvements of 0.57–13.87% on CSI300 and 28.72–51.33% on CSI800 under 0.1% per-trade cost. The framework outperforms alternatives such as DiffsFormer and Quant GAN. Ablation studies reveal a fidelity-diversity tradeoff: while larger guidance strength improves synthetic data fidelity, it does not necessarily enhance prediction performance. Visualizations confirm the preservation of inter-stock correlations and a reduction in overfitting. These results demonstrate InterDiff's ability to enhance robustness and profitability in real-world trading environments and mitigate data scarcity.

Keywords: Data Augmentation · Diffusion Model · Stock Forecasting.

1 Introduction

Stock prediction, which involves forecasting future trends and prices based on historical stock prices and factor time series, is a crucial technique for making profitable investment decisions [1–4]. Numerous machine learning models, such as the State Frequency Memory (SFM) [5], have been proposed for this task. However, the effectiveness of these models heavily depends on the availability of high-quality data. A full year of stock price records typically contains only about 252 daily prices [6]. Furthermore, stock prices exhibit high volatility and are influenced by numerous external factors, leading to a low signal-to-noise ratio (SNR) [7]. These challenges—data scarcity and low SNR—complicate stock prediction, making it difficult to extract meaningful signals and train robust models. As a result, models often suffer from overfitting and poor generalization. A widely adopted approach to mitigate these issues is data augmentation, which generates synthetic sequences to expand the input space while preserving correct labels. This process helps prevent overfitting and improves model generalization [8–10]. Traditional time series augmentation methods rely on transformations in the time, frequency, and time-frequency domains [11–14]. More advanced techniques incorporate decomposition-based methods and statistical generative models [15, 16]. Although these approaches can produce diverse samples, they often struggle to fully capture the complex characteristics of real-world stock data. Recently, deep generative models (DGMs) have gained popularity for time series data augmentation, particularly generative adversarial networks (GANs) and variational autoencoders (VAEs). GAN-based methods, such as TimeGAN [17] and Quant GAN [18], employ an adversarial framework in which a generator and a discriminator compete to produce realistic synthetic data that closely mimics the underlying distribution [19, 20]. Similarly, VAEs, such as TimeVAE [21], use a probabilistic encoder-decoder architecture to learn latent representations, generating synthetic data that maintains temporal dependencies. The most recent advancement in this domain is the application of diffusion models (DMs) for time series augmentation [22]. Methods like Diffsformer [7] leverage a forward and reverse diffusion process to iteratively denoise data, generating high-quality synthetic samples.

A major limitation of existing DGM-based methods is their tendency to treat stocks as independent entities. While these models effectively capture temporal dependencies within individual stocks (intra-stock correlations), they often overlook relationships between different stocks (inter-stock correlations), which contain valuable predictive signals about market behavior [23–28]. For instance, stocks within the same sector or industry frequently exhibit similar long-term trends. Ignoring these relationships can lead to synthetic data that disrupts the statistical dependencies between inter-stock correlations and future returns, ultimately impairing the performance of models trained on such data. A notable exception is DiffsFormer [7], a state-of-the-art model that attempts to incorporate inter-stock correlations by conditioning the data generation process on static industry-sector classifications. This approach assumes that stocks within the same sector behave similarly. However, such a rigid classification oversimplifies real-world market dynamics [29]. In reality, sector-based relationships are fluid and complex—firms within the same industry can respond asymmetrically to market shocks (e.g., competing companies reacting differently to supply chain disruptions), experience intra-sector competition, or even redefine their industry alignment over time (e.g., traditional companies pivoting to AI-driven business models). This raises a critical question: **How can we dynamically capture both intra- and inter-stock correlations to improve financial time series augmentation?**

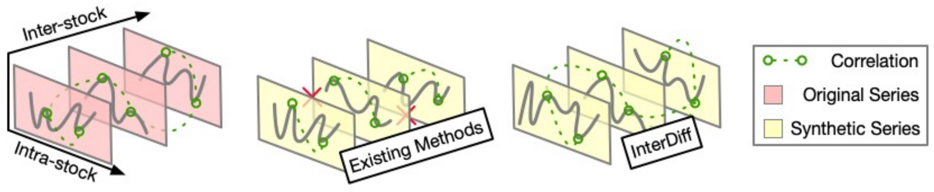


Fig. 1: A comparison of existing methods and InterDiff for financial time series augmentation.

To address this limitation, we present InterDiff, a diffusion-based framework that synthesizes financial time series with dynamic correlations across stocks. InterDiff employs hierarchical transformers to first capture intra-stock correlation and then model inter-stock correlation. These transformers produce a guidance vector that encodes real-world evolving market logic. This vector then steers a diffusion model via classifier-free guidance. During training, InterDiff learns to denoise data by optimizing two objectives: (1) a diffusion loss to match the statistical properties of real data and (2) a supervised loss to align guidance vector with predictive signals for future returns. At inference, InterDiff blends guided predictions (informed by the learned correlations) with unguided predictions, ensuring synthetic data retains realistic inter-stock correlations while introducing controlled variability to avoid overfitting.

This paper makes the following contributions:

(i) **Hierarchical Correlation Learning:** We design a process to dynamically model intra-stock correlations and inter-stock correlations through transformers, encoding these correlations into a guidance vector.
(ii) **InterDiff Framework:** We introduce a diffusion-based method using classifier free guidance, conditioned on learned correlations, to synthesize financial time series that balance fidelity to real data with controlled variability.
(iii) **Empirical Validation:** We demonstrate the framework's effectiveness via comprehensive evaluations on real world datasets (CSI300/CSI800), ablation studies, and visualizations showing preserved market dynamics and improved prediction robustness.

2 Related Work

Deep generative models (DGMs) have demonstrated superior performance over traditional augmentation techniques in capturing the complex characteristics of real data. A pivotal work in this domain is TimeGAN [17], which integrates a generative adversarial network (GAN) with a supervised autoregressive model to preserve temporal dependencies while synthesizing realistic time series data. Another notable model, Quant GAN [18], is specifically designed for financial time series, effectively capturing long-range dependencies such as volatility clustering. Its data-driven approach makes it particularly suited for modeling continuous sequential data with long-term dependencies. A comprehensive review of GAN-based time series synthesis can be found in [30]. Beyond GANs, variational autoencoders (VAEs) offer a probabilistic framework that explicitly models the data distribution, resulting in stable and diverse synthetic samples [34]. TimeVAE [21] enhances this framework by incorporating seasonality and trend modules, improving both model performance and interpretability.

The state-of-the-art model in this field is DiffsFormer [7], which leverages diffusion models (DMs) for stock factor augmentation. Unlike previous methods that generate synthetic data from scratch, DiffsFormer employs transfer learning to refine existing samples. Additionally, it conditions the generation process on static industry classifications, aiming to incorporate inter-stock relationships into the synthetic data. However, we argue that static industry classification is insufficient to capture dynamic inter-stock correlation. Therefore, we propose to condition the generation process on a learned vector that encodes dynamic inter-stock relationships and market trends for realistic synthesis in InterDiff.

3 Background

In this section, we will introduce some definitions in our work and the problem of stock price forecasting.

3.1 Problem Formulation

Stock forecasting aims to predict future normalized returns of stocks based on historical factor data. For each stock $u \in S$, quantifiable factors such as momentum, volatility and liquidity are collected over a historical lookback window of T days, forming a 2-D factor vector $x^u \in \mathbb{R}^{T \times F}$, where F denotes the number of factors. The prediction target is the normalized return ratio r_u, defined as:

$$r_u = \frac{\text{Price}^u_{t+i} - \text{Price}^u_t}{\text{Price}^u_t},$$

where t is the current time and i is the forecast horizon (in days). To mitigate data scarcity, a DGM synthesizes realistic sequences $\hat{x}^u$, augmenting the original dataset with synthetic data that preserves market dynamics. The task requires jointly predicting $\{r_u\}_{u \in S}$ for all stocks using their augmented factors $\{\hat{x}^u\}_{u \in S}$.

3.2 Denoising Diffusion Probabilistic Model

Denoising Diffusion Probabilistic Models (DDPMs) have demonstrated exceptional performance and outperformed Generative Adversarial Networks (GANs) in several areas, particularly in text-to-image generation tasks. The training of a diffusion model involves two main processes: diffusion and denoising.

Diffusion process: Starting with a data point $x_0 \sim q(x_0)$[5], the diffusion process progressively adds noise to create a sequence of step-dependent variables, $\{x_k\}_{k=1}^{K}$. This process can be described as a Markov chain:

$$q(x_{1:K}|x_0) = \prod_{k=1}^{K} q(x_k|x_{k-1}) \tag{1}$$

where $q(x_k|x_{k-1}) = \mathcal{N}(x_k; \sqrt{\alpha_k}x_{k-1}, \beta_k\mathbf{I})$. Here, $\mathcal{N}$ represents a Gaussian distribution, α_k controls the signal retention strength, and β_k governs the scale of the noise added. These scalars, α_k and β_k, are predefined for each step k. A common setting is the variance-preserving process where $\alpha_k = 1 - \beta_k$.

Denoising Process: The goal of the denoising process is to reconstruct the original data by reversing the transformations introduced in the diffusion process. This is accomplished by another Markov chain:

$$p_\theta(x_{0:K}) = p(x_K) \prod_{k=1}^{K} p_\theta(x_{k-1}|x_k) \tag{2}$$

, where $x_K \sim \mathcal{N}(0, \mathbf{I})$. The distribution p_θ is an approximation of the true distribution q. Specifically, $p_\theta(x_{k-1}|x_k) = \mathcal{N}(x_{k-1}; \mu_\theta(x_k, k), \sigma_\theta(x_k, k)\mathbf{I})$, where μ_θ and σ_θ are learned functions of the noisy input x_k and the step k. For each sample in a batch, a time step k is randomly selected from $1, 2, \ldots, K$, and the noise is adjusted accordingly at step k.

Inference Process: Once θ is well-trained, the DM generates samples by initializing $x_K \sim \mathcal{N}(0, \mathbf{I})$ and iteratively denoising through $x_K \to \cdots \to x_k \to x_{k-1} \to \cdots \to x_0$ using $p_\theta(x_{k-1}|x_k)$.

3.3 Classifier and Classifier-Free Guidance

DMs utilize two primary conditioning strategies to incorporate information from a guidance variable c: classifier guidance and classifier-free guidance. Classifier guidance relies on training an auxiliary classifier to estimate $p(c|x_k, k)$. During inference, the gradient $\nabla_{x_k}\log p(c|x_k, k)$ from this classifier is used to steer the synthesis process. Specifically, the predicted noise at each step is adjusted according to

$$\bar{\epsilon} = \epsilon_\theta(x_k, k) - \sqrt{1 - \overline{\alpha}_k}\,\omega\nabla_{x_k}\log p(c|x_k, k) \tag{3}$$

where ω scales the guidance strength.

In contrast, classifier-free guidance eliminates the need for a separate classifier by jointly training a conditional DM and an unconditional DM. These

[5] x_0 denotes the initial (non-noised) step, with the total diffusion steps K set to 500 in our work (or say in the experiment section)

two components are combined during inference using a weighted interpolation of their predicted noise:

$$\hat{\epsilon}_\theta = (1 + \omega_{\text{free}})\epsilon_\theta(x_k, c, k) - \omega_{\text{free}}\epsilon_\theta(x_k, \emptyset, k) \tag{4}$$

where ω_{free} controls the trade-off between conditioning fidelity and sample diversity.

We opt for classifier-free guidance due to a critical limitation inherent to classifier guidance: gradient instability. When classifier gradients are injected into the denoising process, they can behave adversarially. This instability disrupts the generation process, as erratic gradient updates degrade output quality and consistency. Empirical results from [31] confirm that this sensitivity to adversarial-like gradients often renders classifier-guided synthesis inferior to classifier-free approaches.

4 Methodology

The InterDiff framework generates synthetic stock data that balances diversity with correlation-consistency through two key stages. First, for each stock u, a guidance vector e_u is learned to capture its intra-stock correlation and inter-stock correlations through a 3-stage hierarchical transformer (Intra-, Inter-stock and Temporal Aggregation). Next, e_u guides a DM in the denoising and inference process by classifier-free approach.

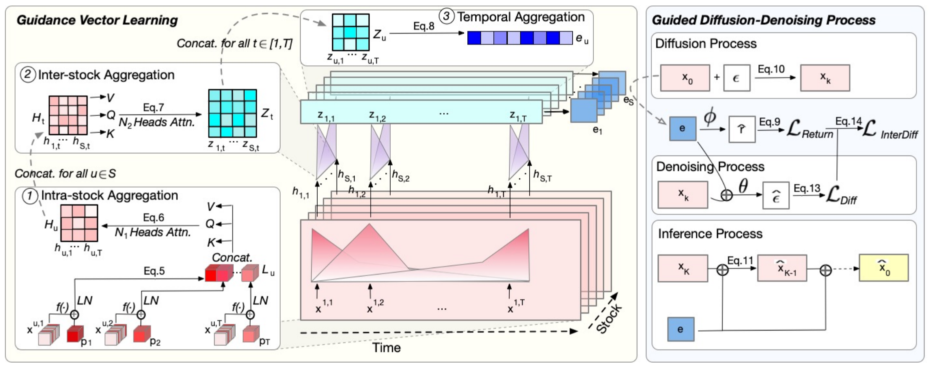


Fig. 2: The Pipeline for InterDiff

4.1 Guidance Vector Learning through Hierarchical Transformer

The guidance vector learning process begins by splitting the set of stock factors $\{x^u\}_{u \in S}$ into $\{x^{u,t}\}_{u \in S, t \in [1,T]}$, where each stock's data is associated with factors at each time step. The learning process proceeds as follows:

Intra-Stock Aggregation: For each stock, the factors at each time step $\{x^{u,t}\}_{u\in S,t\in[1,T]}$ are first encoded into embeddings $l_{u,t} = f(x^{u,t})$ via a linear layer $f(\cdot)$. A transformer encoder then processes these embeddings with sinusoidal positional encodings p_t to preserve chronological order:

$$L_u = ||_{t\in[1,T]} LN(f(x^{u,t}) + p_t) \tag{5}$$

where $||$ denotes concatenation and LN is layer normalization. Multi-head attention (N_1 heads) and feed-forward networks (FFNs) then aggregate intra-stock correlations across time steps. The transformer computes query (Q^1_u), key (K^1_u) and value (V^1_u) matrices from L_u producing local embeddings $h_{u,t}$ enriched with cross-time signals:

$$\begin{aligned} Q^1_u = W^1_Q L_u, \quad K^1_u = W^1_K L_u, \quad V^1_u = W^1_V L_u \\ H^1_u = ||_{t\in[1,T]} h_{u,t} = FFN^1(MHA^1(Q^1_u, K^1_u, V^1_u) + L_u) \end{aligned} \tag{6}$$

These embeddings retain local temporal details while integrating global historical context, ensuring the learned intra-stock correlations reflect both short-term fluctuations and long-term trends.

Inter-Stock Aggregation: At each time step t, local embeddings $\{h_{u,t}\}_{u\in S}$ from all stocks are combined to model inter-stock correlations. Another multi-head attention layer (N_2 heads) computes cross-stock interactions:

$$\begin{aligned} Q^2_t = W^2_Q H_t, \quad K^2_t = W^2_K H_t, \quad V^2_t = W^2_V H_t \\ Z_t = ||_{u\in S} z_{u,t} = FFN^2(MHA^2(Q^2_t, K^2_t, V^2_t) + H^2_t) \end{aligned} \tag{7}$$

where $H^2_t = ||_{u\in S} h_{u,t}$. The temporal embedding $z_{u,t}$ for stock u at time t encodes both its intrinsic features and dependencies on other stocks.

Temporal Aggregation: To summarize the obtained temporal embeddings and obtain a comprehensive stock embedding e_u, we employ a temporal attention layer along the time axis. We use the latest temporal embedding $z_{u,T}$ as the query vector, and compute the attention score $\lambda_{u,t}$ in a hidden space with transformation matrix W_λ,

$$\lambda_{u,t} = \frac{exp(z^T_{u,t} W_\lambda z_{u,T})}{\sum_{i\in[1,T]} exp(z^T_{u,i} W_\lambda z_{u,T})}, \quad e_u = \sum_{t\in[1,T]} \lambda_{u,t} z_{u,t} \tag{8}$$

Return Calibration: For stock forecasting, each input requires a label, but rather than directly generating label as an additional dimension, using ground truth labels to guide data generation and keep the original label is more effective [7]. Therefore, we introduce an auxiliary predictor ϕ that maps the learned guidance vector e_u to the stock return r_u. This ensures e_u captures the dependencies critical for forecasting while allowing us to use the original return as a label through a mean squared error (MSE) loss:

$$\mathcal{L}_{\text{return}} = \mathbb{E}_{u\in S}[||r_u - \phi(e_u)||^2_2] \tag{9}$$

4.2 Classifier-Free Correlation Guided Diffusion-Denoising Process

The InterDiff framework synthesizes stock data with realistic intra- and inter-stock correlations by integrating a DM conditioned on the learned guidance vector e_u. For clarity, we omit the stock-specific superscript u in this section (e.g., $x_0^u \equiv x_0, e_u \equiv e$), as the diffusion and denoising processes operate identically for each stock. The process follows a diffusion-denoising pipeline.

Diffusion Process: The forward process gradually corrupts x_0 over K steps by adding Gaussian noise. At step k, the noised state x_k follows $q(x_k|x_0) = \mathcal{N}(x_k; \sqrt{\overline{\alpha}_k} x_0, (1-\overline{\alpha}_k)\mathbf{I})$, where $\overline{\alpha}_k = \prod_{m=1}^{k} \alpha_m$ and $\alpha_k = 1 - \beta_k$. This allows direct sampling of x_k via reparameterization:

$$x_k = \sqrt{\overline{\alpha}_k} x_0 + \sqrt{1-\overline{\alpha}_k}\epsilon, \quad \text{where } \epsilon \sim \mathcal{N}(0, \mathbf{I}) \tag{10}$$

.

Denoising Process: The denoising process iteratively removes noise from x_k to reconstruct $\hat{x}_0 \sim q(x_0)$. To enable classifier-free guidance, we stochastically mask the guidance vector e with a null token $\emptyset$ with probability p during training, enabling the model to learn both conditional and unconditional denoising. Formally:

$$p_\theta(x_{k-1}|x_k, e) = \mathcal{N}(x_{k-1}; \mu_\theta(x_k, k, e), \Sigma_q(k)\mathbf{I}) \tag{11}$$

with:

$$\mu_\theta(x_k, k, e) = \begin{cases} \frac{1}{\sqrt{\alpha_k}} \left(x_k - \frac{\beta_k}{\sqrt{1-\overline{\alpha}_k}} \epsilon_\theta(x_k, k, e) \right), \text{with probability } (1-p) \\ \frac{1}{\sqrt{\alpha_k}} \left(x_k - \frac{\beta_k}{\sqrt{1-\overline{\alpha}_k}} \epsilon_\theta(x_k, k, \emptyset) \right), \text{with probability } p \end{cases} \tag{12}$$

The mean square error between the true noise ϵ and the predicted noise ϵ_θ is computed across all timesteps and incorporated into the InterDiff loss function to ensure accurate noise estimation and effective denoising.

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{x_0 \sim q(x_0), \epsilon \sim \mathcal{N}(0,\mathbf{I}), k \sim \text{Uniform}(1,K)}[||\epsilon - \epsilon_\theta||_2^2] \tag{13}$$

Loss Function: InterDiff employs an uncertainty-aware loss [32] that dynamically weights the denoising and return calibration tasks to balance the dual objectives of generating relationally consistent stock data and preserving return-predictive signals. The total loss integrates the DM's noise prediction error $\mathcal{L}_{\text{diff}}$ and the return calibration error $\mathcal{L}_{\text{return}}$ through learnable variance parameters σ^2_{diff} and σ^2_{return}, which quantify the intrinsic uncertainty of each task:

$$\mathcal{L}_{\text{InterDiff}} = \frac{1}{2\sigma^2_{\text{diff}}} \mathcal{L}_{\text{diff}} + \frac{1}{2\sigma^2_{\text{return}}} \mathcal{L}_{\text{return}} + \frac{1}{2}(\log \sigma^2_{\text{diff}} + \log \sigma^2_{\text{return}}) \tag{14}$$

The first two terms adaptively scale each loss based on task difficulty, where volatile tasks (higher σ^2) receive lower weights. The third term penalizes excessive uncertainty. This mechanism eliminates manual loss weighting, critical in financial applications where market volatility and return scales vary widely across stocks and time periods.

Inference Process: Synthetic data generation begins with Gaussian noise x_K and a guidance vector e. The model iteratively denoises x_K to $\hat{x}_0$ over K steps. During inference, the predicted noise $\hat{\epsilon}_\theta$ combines conditional and unconditional predictions using a guidance strength ω_{free}:

$$\hat{\epsilon}_\theta = (1 - \omega_{\text{free}}) \cdot \epsilon_\theta(x_k, k, \emptyset) + \omega_{\text{free}} \cdot \epsilon_\theta(x_k, k, e) \tag{15}$$

This blended prediction guides the denoising trajectory, ensuring outputs align with the correlation patterns encoded in e while maintaining statistical diversity. The final synthetic data $\hat{x}_0$ is sampled by progressively refining $\hat{x}_{k-1}$ from $p_\theta(x_{k-1}|x_k, e)$.

5 Experiment

We conduct experiments to address three key questions. **RQ1 (Compatibility)**: Does InterDiff generalize across diverse backbone models while improving their forecasting performance? **RQ2 (Component Efficacy)**: How do key design choices, such as guidance variable and critical parameters like guidance strength, influence the quality of synthetic data and the overall performance of the model? **RQ3 (Correlation Preservation)**: Can InterDiff synthesize data that retains realistic intra- and inter-stock correlations observed in real markets?

5.1 Dataset

We evaluate InterDiff on CSI300 and CSI800—datasets comprising the 300 and 800 largest stocks by market capitalization from the Shanghai and Shenzhen exchanges. Following [33] and [7], we use daily data spanning 2008–2023, partitioned into training (Q1 2008–Q1 2021), validation (Q2 2021), and test (Q3 2021–Q4 2023) sets. Stock features are obtained from the Alpha158 factor suite in Qlib[6], with a lookback window $T = 8$ days and prediction horizon $i = 5$ days, consistent with [29].

5.2 Reproducibility

We implement InterDiff using Python 3.8.5 and PyTorch 1.11.0, running on NVIDIA RTX 3090 GPUs and AMD EPYC 7532 CPUs. To facilitate reproducibility, we outline the key techniques used in our implementation.

First, we use Robust Z-score Normalization, which replaces the mean and standard deviation with the median (MED) and median absolute deviation (MAD). This approach ensures scale-invariant features and reduces sensitivity to outliers:

$$\tilde{x}^u = \frac{|x^u - \text{MED}(X)|}{\text{MAD}(X)}, \tag{16}$$

[6] https://github.com/microsoft/qlib

Second, we apply Extreme Label Filtering, which removes the top and bottom 2.5% of returns. This step mitigates distortions caused by limit-up and limit-down events, preventing extreme values from biasing the model. Third, we employ the "train on synthetic, test on real" (TSTR) strategy to evaluate the effectiveness of the synthetic data generated by InterDiff. This approach involves training the model on the augmented, synthetic dataset and testing it on real-world data, allowing us to assess the generalization of the model when faced with actual market conditions. Lastly, we optimize hyperparameters such as guidance vector size, attention heads and guidance strength through grid search. The best-performing values are highlighted in Table 1.

Parameters	Search Range
layers in DM	{3, **6**}
stop loss thred	{0.6, 0.8, 0.9, 0.95, **0.965**, 1}
Guidance Vector Size	{128, **256**, 512}
Guidance Strength ω_{free}	{1.5, **2**, 3, 4, 5}
Attention Heads N_1	{2, 3, **4**, 5}
Attention Heads N_2	{**2**, 3, 4, 5}

Table 1: Hyper-parameters and the search range, the optimal parameters are indicated in boldface.

5.3 Experimental Setup

To evaluate our framework's performance in stock forecasting, we use five widely adopted models as backbones: Multi-Layer Perceptron (MLP), Long Short-Term Memory (LSTM) [36], Gated Recurrent Unit (GRU) [37], State Frequency Memory (SFM) [5], and a Transformer-based model [35]. These models serve as forecasting backbones, allowing us to compare their effectiveness in predicting stock movements. Each experiment is repeated 10 times, with results averaged for robust model evaluation and statistical reliability. Training follows a daily batch strategy: each batch comprises all stock data for a single day, enabling gradient updates based on return prediction errors per day. The model minimizes the Pearson Loss:

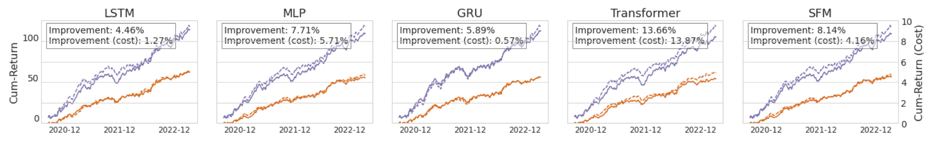
$$\mathcal{L}_{\text{Pearson}} = 1 - \frac{\text{COV}\left(\boldsymbol{r}, \boldsymbol{r}^{\text{pred}}\right)}{\text{SD}(\boldsymbol{r}) \cdot \text{SD}\left(\boldsymbol{r}^{\text{pred}}\right)}, \tag{17}$$

where $\boldsymbol{r}, \boldsymbol{r}^{\text{pred}} \in \mathbb{R}^{|S|}$ denote ground-truth and predicted daily returns for all stocks, respectively. Each r_u represents the normalized daily return of stock u.

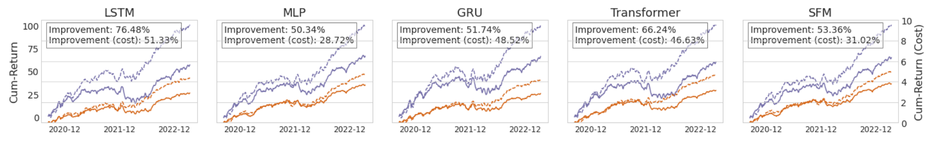
We assess model performance using both ranking metrics and portfolio-based metrics. The ranking metrics include Information Coefficient (IC) [38], which measures the Pearson correlation between predictions and labels, and Rank Information Coefficient (RankIC) [39], which computes the Spearman rank correlation. These metrics provide insights into the model's ability to generate accurate stock rankings. For portfolio-based evaluation, we use cumulative return to measure investment profitability. We simulate stock trading using a 'top30-drop30' strategy under two scenarios: with and without transaction cost. The strategy retains the top 30 stocks with the highest predicted return ratios, while any stock that falls out of the top 30 is dropped, regardless of its previous ranking. For the cost-inclusive scenario, a transaction cost of 0.1% per trade is applied to both entry (buy) and exit (sell) transactions. This dual evaluation helps assess both the theoretical potential and practical viability of forecasting models in real trading environments.

5.4 Performance Comparison (RQ1)

To answer RQ1, we conduct a thorough comparison of key metrics for backbones trained on both original and augmented data. InterDiff consistently enhances cumulative returns across all models and datasets. For CSI300 (Fig. 3a), improvements from synthetic data range from 4.46% to 13.66% without transaction cost. When incorporating a 0.1% per-trade cost, improvements persist but vary by model: the Transformer retains nearly all gains (13.87%), while GRU's improvement drops sharply to 0.57%, reflecting sensitivity to turnover. MLP and SFM exhibit moderate erosion, highlighting model-specific cost tolerance. For CSI800 (Fig. 3b), improvements are far more pronounced, with cost-free gains spanning 50.34% to 76.48%. Transaction costs reduce returns but preserve significant margins. These improvements underscore InterDiff's capability to generate correlation-aware synthetic financial time series that enhance model robustness and profitability across diverse market conditions.



(a) Cumulative Return for CSI300



(b) Cumulative Return for CSI800

Fig. 3: Cumulative returns (in %) for CSI300 and CSI800. Purple lines exclude a 0.1% transaction cost; dashed lines represent models trained on InterDiff-augmented data. Solid lines denote original data.

Table 2a and Table 2b show that ranking metrics such as IC and RankIC also improve universally with the augmented data. The sharper increase in performance for CSI800 (40.15-49.60% for IC and 39.8-68.91% for RankIC) underscores the ability of InterDiff to adapt to markets with more noise and variation. This suggests that InterDiff not only improves predictive accuracy but also enhances the model's ability to handle the complexities of broader, more volatile markets. What sets InterDiff apart from DiffsFormer [7] is its ability to capture dynamic inter-stock correlations, making it highly scalable. By incorporating evolving relationships between stocks, InterDiff amplifies predictive signals in markets with greater volatility and complexity, thereby offering a substantial advantage over traditional methods that rely on static assumptions and data.

Methods	**CSI300**					
	IC			**RankIC**		
	original	augmentation	↑	original	augmentation	↑
MLP	0.0512±0.0005	**0.0535±0.0008**	4.49%	0.0458±0.0011	**0.0464±0.0005**	1.31%
LSTM	0.0496±0.0012	**0.0519±0.0011**	4.64%	0.0365±0.0012	**0.0389±0.0021**	6.58%
GRU	0.0480±0.0003	**0.0502±0.0006**	4.58%	0.0334±0.0001	**0.0351±0.0005**	5.09%
SFM	0.0511±0.0005	**0.0535±0.0008**	4.70%	0.0455±0.0009	**0.0458±0.0011**	0.66%
Transformer	0.0530±0.0008	**0.0536±0.0008**	1.13%	0.0451±0.0010	**0.0454±0.0006**	0.67%

(a) Performance comparison on CSI300. The better results are indicated in boldface.

Methods	**CSI800**					
	IC			**RankIC**		
	original	augmentation	↑	original	augmentation	↑
MLP	0.0269±0.0005	**0.0377±0.0003**	40.15%	0.0334±0.0007	**0.0467±0.0005**	39.80%
LSTM	0.0241±0.0008	**0.0348±0.0008**	44.4%	0.0256±0.0013	**0.0409±0.0012**	59.77%
GRU	0.0231±0.0006	**0.0330±0.0014**	42.86%	0.0231±0.0009	**0.0363±0.0017**	57.14%
SFM	0.0270±0.0008	**0.0379±0.0025**	40.37%	0.0331±0.0011	**0.0471±0.0029**	42.30%
Transformer	0.0250±0.0005	**0.0374±0.0007**	49.60%	0.0267±0.0007	**0.0451±0.0012**	68.91%

(b) Performance comparison on CSI800. The better results are indicated in boldface.

Table 2: Performance Comparison of Different Backbones on CSI300 and CSI800 with Original and Augmented Data

5.5 Ablation Study (RQ2)

To answer RQ2, we focus on the impact of guidance strength and the choice of guidance variable on the fidelity and diversity of synthetic data generated by InterDiff. Fidelity refers to how closely the synthetic data mirrors real data, while diversity refers to the model's ability to generalize by introducing variation in the synthetic data. To evaluate fidelity, we use the Fréchet Inception Distance (FID), a metric commonly used to assess the similarity between distributions of real and generated data. Diversity, on the other hand, is assessed through model performance in terms of Information Coefficient (IC), following [7].

As shown in Fig.4, we observe that as the guidance strength increases, the FID decreases, indicating that the synthetic data becomes more faithful to the original data. Specifically, InterDiff is able to achieve an FID as low as 0.5004, which suggests a high degree of fidelity in the generated data. This finding highlights the effectiveness of the correlation guidance mechanism employed by InterDiff. Furthermore, Fig.4 reveals a fidelity-diversity tradeoff: Beyond a certain point ($\omega_{\text{free}}>2$), the performance starts to degrade. This may occurs due to over-specialization, where the model becomes too focused on specific patterns and fails to capture the broader, more diverse dynamics of the market. This highlights the importance of finding the right balance between fidelity and diversity to ensure that the synthetic data is both accurate and generalizable.

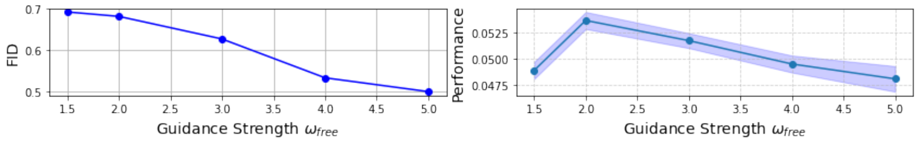


Fig. 4: Synthetic data quality (FID) and model performance (IC).

We compare InterDiff with two variants of DiffsFormer [7] guided by return labels (DiffsFormer-R) and another guided by static industry classifications (DiffsFormer-I), as well as Quant GAN [18]. As shown in Figure 5, InterDiff achieves improvements of 4.30% and 5.42% in IC over DiffsFormer-R and DiffsFormer-I respectively, showing the effectiveness of InterDiff's correlation guidance mechanism for generating synthetic data that enhances predictive performance. Quant GAN degrades IC performance compared to training on original data, reflecting the instability in financial applications. This instability aligns with our rationale for adopting diffusion models in InterDiff.

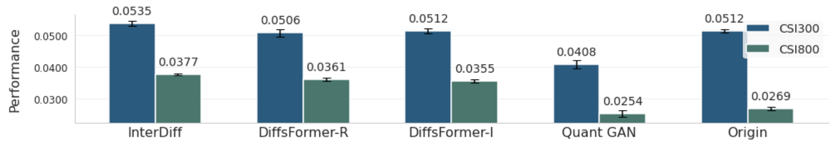


Fig. 5: Comparison of InterDiff with DiffsFormer variants and Quant GAN.

5.6 Visualization (RQ3)

To analyze the learned inter-stock correlations, we visualize attention maps from the inter-stock aggregation transformer in Fig.6. The figure depicts attention scores across three trading dates for two target stocks (SH601857 and SH601398) and 100 randomly sampled stocks. The red box shows regions where attention weights weaken over time, signaling shifting trend patterns, while the green box marks persistent trends with consistently high attention. These observations confirm that the intra-stock aggregation module effectively captures temporal relationships within individual stocks. Notably, in the blue box, SH601857 exhibits sparse correlations, whereas SH601398 shows strong dependencies with multiple stocks, validating Inter-stock aggregation module's ability to dynamically capture asymmetric and evolving market relationships.

To evaluate InterDiff's ability to maintain inter-stock correlations, we compare real and synthetic data from the CSI300 on a randomly selected date (December 30, 2022). Fig.7 (top row) shows pairwise correlation heatmaps, where denoising progressively removes noise while retaining correlation structures. The histogram (bottom-left) confirms synthetic data's correlation distribution increasingly aligns with real data during denoising. The t-SNE plots (bottom-right) visualize overall data distribution alignment by showing how synthetic samples gradually converge to real data manifolds while maintaining controlled variability, demonstrating InterDiff's capacity to generate realistic yet diverse sequences that generalize to out-of-sample scenarios.

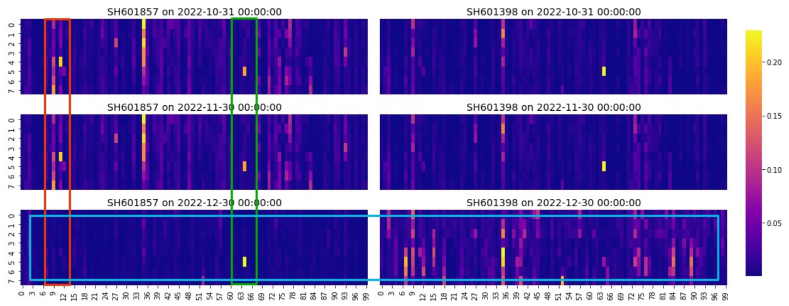


Fig. 6: Attention Map for target stocks SH601857 (left) and SH601398 (right) across three dates, with source stocks (x-axis) and timesteps (y-axis).

5.7 Discussion

To examine how InterDiff helps mitigate overfitting caused by data scarcity, we plot the training loss over time in Fig.8. The three subplots represent training loss curves for models trained on (1) original data, (2) data augmented with random noise, and (3) data augmented by InterDiff. Random noise addition is a simple augmentation method used to improve model robustness.

During the 2020 stock market crash[7] triggered by the COVID-19 pandemic, stock forecasting loss was notably low. We attribute this to the market's temporary simplification, where price movements were dominated by a few strong

[7] https://en.wikipedia.org/wiki/2020_stock_market_crash

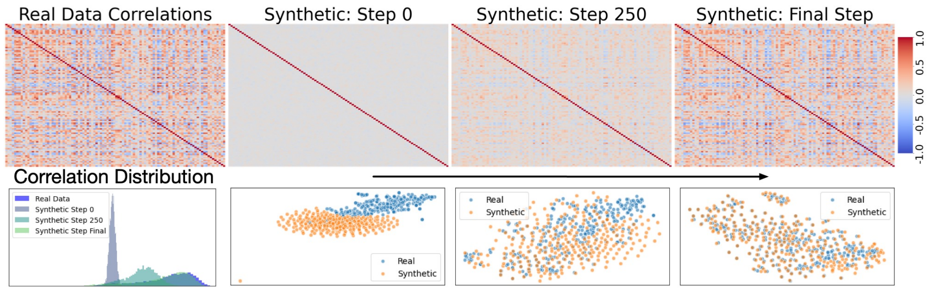


Fig. 7: Synthetic vs. real data distributions visualized via t-SNE, pair-wise correlation heatmaps, and histograms across denoising steps (0–500).

factors, as highlighted by the red dots in Fig.8. However, a model that overfits to this period may struggle to generalize in later years when market patterns become more complex. Simply removing these data points is not an ideal solution, as it would further exacerbate data scarcity. As shown in Fig.8, models trained on InterDiff-augmented data exhibit smoother and more stable loss curves compared to those trained with random noise augmentation. This suggests that InterDiff effectively reduces overfitting while preserving essential market patterns, leading to a more generalizable forecasting model.

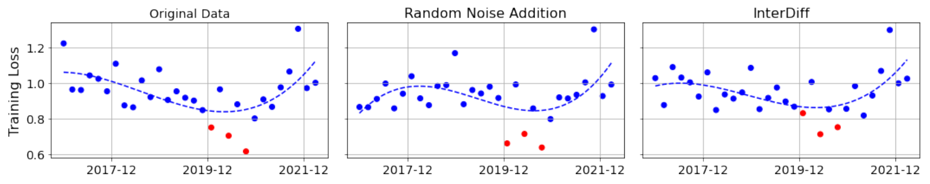


Fig. 8: Training loss for original, noise-augmented, and InterDiff-augmented data.

6 Conclusion

This paper introduces InterDiff, a diffusion-based framework that synthesizes financial time series by dynamically modeling intra- and inter-stock relationships through hierarchical transformers. By encoding these correlations into a guidance vector and leveraging classifier-free guidance during diffusion, InterDiff generates synthetic data that preserves realistic market dynamics while introducing controlled variability. Empirical evaluations on CSI300 and CSI800 datasets demonstrate that InterDiff-augmented data consistently improves forecasting performance across diverse backbone models. Ablation studies confirm the effectiveness of InterDiff's correlation guidance mechanism, demonstrating its ability to balance fidelity and diversity in synthetic financial time series data, while visualizations validate its capacity to capture evolving market correlations. InterDiff mitigates overfitting, as evidenced by smoother training loss curves compared to baseline augmentation methods. These results highlight the importance of dynamic correlation modeling in financial data augmentation, offering a robust solution to enhance stock prediction models in real-world scenarios.

Acknowledgments. This research was supported in part by the Department of Statistics and New Asia College at The Chinese University of Hong Kong, by Super Quantum Capital Management, and by the Young Scientists Fund of the National Natural Science Foundation of China (Grant number 72301125). The research was carried out during Hou-Wan Long's internship at Super Quantum Capital Management.

References

1. Nugroho, FX Satriyo D, Adji, Teguh Bharata, and Fauziati, Silmi. "Decision support system for stock trading using multiple indicators decision tree." In *2014 The 1st International Conference on Information Technology, Computer, and Electrical Engineering*, 2014, pp. 291–296. IEEE.
2. Kamble, Rupesh A. "Short and long term stock trend prediction using decision tree." In *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2017, pp. 1371–1375. IEEE.
3. Xie, Boyi, Passonneau, Rebecca, Wu, Leon, and Creamer, Germán G. "Semantic frames to predict stock price movement." In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 2013, pp. 873–883.
4. Li, Qing, Jiang, LiLing, Li, Ping, and Chen, Hsinchun. "Tensor-based learning for predicting stock movements." In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.
5. Zhang, Liheng, Aggarwal, Charu, and Qi, Guo-Jun. "Stock Price Prediction via Discovering Multi-Frequency Trading Patterns." In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 2141–2149. Association for Computing Machinery.
6. Cetingoz, Adil Rengim, and Lehalle, Charles-Albert. "Synthetic Data for Portfolios: A Throw of the Dice Will Never Abolish Chance." *arXiv preprint*, 2025.
7. Gao, Yuan, Chen, Haokun, Wang, Xiang, Wang, Zhicai, Wang, Xue, Gao, Jinyang, and Ding, Bolin. "Diffsformer: A diffusion transformer on stock factor augmentation." *arXiv preprint*, 2024.
8. Shorten, Connor, and Khoshgoftaar, Taghi M. "A Survey on Image Data Augmentation for Deep Learning." *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019. Springer.
9. Um, Terry T., Pfister, Franz M. J., Pichler, Daniel, Endo, Satoshi, Lang, Muriel, Hirche, Sandra, Fietzek, Urban, and Kulić, Dana. "Data Augmentation of Wearable Sensor Data for Parkinson's Disease Monitoring Using Convolutional Neural Networks." In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, 2017, pp. 216–220.
10. Le Guennec, Arthur, Malinowski, Simon, and Tavenard, Romain. "Data Augmentation for Time Series Classification Using Convolutional Neural Networks." In *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data*, 2016.

11. Yang, Wenbo, Yuan, Jidong, and Wang, Xiaokang. "SFCC: Data Augmentation with Stratified Fourier Coefficients Combination for Time Series Classification." *Neural Processing Letters*, vol. 55, no. 2, 2023, pp. 1833–1846. Springer.
12. Li, Yan, Lu, Xinjiang, Wang, Yaqing, and Dou, Dejing. "Generative Time Series Forecasting with Diffusion, Denoise, and Disentanglement." *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 23009–23022.
13. Gao, Zijun, Liu, Haibao, and Li, Lingbo. "Data Augmentation for Time-Series Classification: An Extensive Empirical Study and Comprehensive Survey." *arXiv preprint*, 2023.
14. Takahashi, Shuntaro, Chen, Yu, and Tanaka-Ishii, Kumiko. "Modeling financial time-series with generative adversarial networks." *Physica A: Statistical Mechanics and its Applications*, vol. 527, 2019, p. 121261. Elsevier.
15. Wen, Qingsong, Sun, Liang, Yang, Fan, Song, Xiaomin, Gao, Jingkun, Wang, Xue, and Xu, Huan. "Time Series Data Augmentation for Deep Learning: A Survey." *arXiv preprint arXiv:2002.12478*, 2020.
16. Ho, Jonathan, Saharia, Chitwan, Chan, William, Fleet, David J, Norouzi, Mohammad, and Salimans, Tim. "Cascaded diffusion models for high fidelity image generation." *Journal of Machine Learning Research*, vol. 23, no. 47, 2022, pp. 1–33.
17. Yoon, Jinsung, Jarrett, Daniel, and Van der Schaar, Mihaela. "Time-series generative adversarial networks." *Advances in Neural Information Processing Systems*, vol. 32, 2019.
18. Wiese, Magnus, Knobloch, Robert, Korn, Ralf, and Kretschmer, Peter. "Quant GANs: deep generation of financial time series." *Quantitative Finance*, vol. 20, no. 9, 2020, pp. 1419–1440. Taylor & Francis.
19. Lee, Tracey Eileen K. M., Kuah, Y. L., Leo, Kee-Hao, Sanei, Saeid, Chew, Effie, and Zhao, Ling. "Surrogate Rehabilitative Time Series Data for Image-Based Deep Learning." In *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5. IEEE.
20. Kollovieh, Marcel, Ansari, Abdul Fatir, Bohlke-Schneider, Michael, Zschiegner, Jasper, Wang, Hao, and Wang, Yuyang Bernie. "Predict, Refine, Synthesize: Self-Guiding Diffusion Models for Probabilistic Time Series Forecasting." *Advances in Neural Information Processing Systems*, vol. 36, 2024.
21. Desai, Abhyuday, Freeman, Cynthia, Wang, Zuhui, and Beaver, Ian. "TimeVAE: A Variational Auto-Encoder for Multivariate Time Series Generation." *arXiv preprint*, 2021.
22. Ho, Jonathan, Jain, Ajay, and Abbeel, Pieter. "Denoising diffusion probabilistic models." *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 6840–6851.
23. Huynh, Thanh Trung, Nguyen, Minh Hieu, Nguyen, Thanh Tam, Nguyen, Phi Le, Weidlich, Matthias, Nguyen, Quoc Viet Hung, and Aberer, Karl. "Efficient integration of multi-order dynamics and internal dynamics in stock movement prediction." In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 2023, pp. 850–858.
24. Wang, Heyuan, Li, Shun, Wang, Tengjiao, and Zheng, Jiayi. "Hierarchical Adaptive Temporal-Relational Modeling for Stock Trend Prediction." In *IJCAI*, 2021, pp. 3691–3698.
25. Liu, Jintao, Lin, Hongfei, Liu, Xikai, Xu, Bo, Ren, Yuqi, Diao, Yufeng, and Yang, Liang. "Transformer-based capsule network for stock movement prediction." In *Proceedings of the First Workshop on Financial Technology and Natural Language Processing*, 2019, pp. 66–73.

26. Ding, Qianggang, Wu, Sifan, Sun, Hao, Guo, Jiadong, and Guo, Jian. "Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction." In *IJCAI*, 2020, pp. 4640–4646.
27. Yoo, Jaemin, Soun, Yejun, Park, Yong-chan, and Kang, U. "Accurate multivariate stock movement prediction via data-axis transformer with multi-level contexts." In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 2037–2045.
28. Feng, Fuli, He, Xiangnan, Wang, Xiang, Luo, Cheng, Liu, Yiqun, and Chua, Tat-Seng. "Temporal relational ranking for stock prediction." *ACM Transactions on Information Systems (TOIS)*, vol. 37, no. 2, 2019, pp. 1–30. ACM New York, NY, USA.
29. Li, Tong, Liu, Zhaoyang, Shen, Yanyan, Wang, Xue, Chen, Haokun, and Huang, Sen. "MASTER: Market-Guided Stock Transformer for Stock Price Forecasting." In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 1, 2024, pp. 162–170.
30. Brophy, Eoin, Wang, Zhengwei, She, Qi, and Ward, Tomás. "Generative Adversarial Networks in Time Series: A Systematic Literature Review." *ACM Computing Surveys*, vol. 55, no. 10, pp. 1–31, 2023. ACM New York, NY.
31. Nichol, Alex, Dhariwal, Prafulla, Ramesh, Aditya, Shyam, Pranav, Mishkin, Pamela, McGrew, Bob, Sutskever, Ilya, and Chen, Mark. "Glide: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models." arXiv preprint, 2021.
32. Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7482–7491).
33. Xu, Wentao, Liu, Weiqing, Wang, Lewen, Xia, Yingce, Bian, Jiang, Yin, Jian, and Liu, Tie-Yan. "Hist: A graph-based framework for stock trend forecasting via mining concept-oriented shared information." *arXiv preprint*, 2021.
34. Long, Qingyue, Wang, Huandong, Li, Tong, Huang, Lisi, Wang, Kun, Wu, Qiong, Li, Guangyu, Liang, Yanping, and Yu, Li. "Practical Synthetic Human Trajectories Generation Based on Variational Point Processes." In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4561–4571, 2023.
35. Vaswani, A. "Attention is all you need." *Advances in Neural Information Processing Systems*, 2017.
36. Hochreiter, S. "Long Short-term Memory." *Neural Computation*, MIT Press, 1997.
37. Chung, Junyoung, Gulcehre, Caglar, Cho, KyungHyun, and Bengio, Yoshua. "Empirical evaluation of gated recurrent neural networks on sequence modeling." *arXiv preprint*, 2014.
38. Lin, Hengxu, Zhou, Dong, Liu, Weiqing, and Bian, Jiang. "Learning Multiple Stock Trading Patterns with Temporal Routing Adaptor and Optimal Transport." In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 1017–1026. Association for Computing Machinery.
39. Li, Zhige, Yang, Derek, Zhao, Li, Bian, Jiang, Qin, Tao, and Liu, Tie-Yan. "Individualized Indicator for All: Stock-wise Technical Indicator Optimization with Stock Embedding." In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 894–902. Association for Computing Machinery.