Physica A 664 (2025) 130442

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Enhancing stock ranking forecasting by modeling returns with heteroscedastic Gaussian Distribution

Jiahao Yang [a], Ran Fang [a], Ming Zhang [b], Wenkai Zhang [c], Jun Zhou [a,*]

[a] The Institute of Acoustics of the Chinese Academy of Sciences (IACAS), Haidian District, Beijing, China
[b] School of Microelectronics, Hubei University, Wuhan, China
[c] Center for Pervasive Communications and Computing (CPCC), University of California, Irvine, USA

ARTICLE INFO

Keywords:
Stock ranking forecasting
Deep learning
Maximum likelihood principle
Heteroscedastic
Gaussian distribution

ABSTRACT

Accurately selecting stocks with the highest returns is crucial for profitable investing. However, predicting stock price movements is challenging due to the high degree of randomness caused by factors such as market opacity, unexpected events, erratic trades, etc. Previous research has primarily focused on extracting more information from inputs to map to the observed returns, such as modeling the complex relations of different stocks. However, they overlooked the uncertainty of returns caused by the randomness market. To mitigate it, we propose a novel analytical framework. The starting point is that the stock returns follow some distributions, so the observed returns are samples from them, and the variances are the source of randomness. After analysis, past studies were equivalent regarding the returns of different stocks at each time following homoscedastic Gaussian distributions, aiming to predict the mean of these distributions. We find that the hypothesis to be unreasonable and extend these distributions to the heteroscedastic case, presenting a revised model structure and learning objectives. The proposed method aims to simultaneously predict the mean and the standard deviation of distributions from inputs, and the model is trained based on the maximum likelihood principle. Experiment results on the stock members of the CSI 100, 300, and 500 Chinese market indexes show significant improvements compared with the previous methods. The annualized return of the Top 20 stock portfolios improved absolutely 2%, 20%, and 50%, proving the effectiveness of our framework. We discuss the roles of the obtained mean and standard deviation in pursuing more profits, and we extend our theory to a more general form through mathematical derivation.

1. Introduction

Constructing portfolios with the highest returns from candidate assets is the key to profiting from the stock market, which necessitates the ability to forecast the return rankings of stocks accurately. Although investors have discovered the patterns of stock price movement and profit from them, such as the mean-reversion of prices [1], their methods are too simple to capture the complex and time-varying market conditions and are ineffective over extended periods [2]. Consequently, the stock market's randomness, driven by factors such as market opacity, unexpected events, and erratic trades, makes it challenging for investors to make accurate forecasts

* Corresponding author.
E-mail addresses: yangjiahaostu@gmail.com (J. Yang), fangran@hccl.ioa.ac.cn (R. Fang), zhangming.peter@gmail.com (M. Zhang), wenkaiz1@uci.edu (W. Zhang), zhoujun@hccl.ioa.ac.cn (J. Zhou).

https://doi.org/10.1016/j.physa.2025.130442
Received 10 September 2024; Received in revised form 15 November 2024
Available online 21 February 2025

An intuitive approach is to assume a correlation between previous and future prices and model it using signal processing methods, like the Auto-Regressive Integrated Moving Average (ARIMA) model [3], Kalman filter [4], and the Prophet proposed by Facebook [5]. These methods tend to have good explainability due to the assumption of price changes, but they are too simple to model the complicated nonlinear connections between the inputs and outputs. Then, some studies use machine learning methods to instruct nonlinearity mapping and capture more complex stock price change patterns, like Hidden Markov Model (HMM) [6], Support Vector Machine (SVM) [7,8], Extreme Gradient Boosting (XGBoost) [9], and Random Forest (RF) [10]. Moreover, some unsupervised methods, like Fourier Transform (FT), Principle Component Analysis (PCA), and wavelet analysis, are combined with machine learning methods to mitigate the impact of noise and encode more representative features [11,12]. These methods further relax assumptions about the data generation process, but they are too simple to model complicated price movement patterns and always require manually designed features.

Owing to their powerful modeling capabilities and strong generalization [13], deep learning-based methods have gradually become a focal point for researchers and have demonstrated impressive performance. In recent years, various network structures have been utilized for forecasting, including the basic Multi-layer Perceptron (MLP) [14,15], Recurrent Neural Network (RNN) [16] and its variant Long-Short Term Memory (LSTM) and Gate Recurrent Unit (GRU) [17,18], Convolutional Neural Network (CNN) [19,20], Generative Adversarial Network (GAN) [21], Transformer [22–24], and their combination [25–27], and some structures have been proved effective to capture the stock price patterns from historical inputs. Moreover, some researchers have considered combining some heterogeneous data to mine features that affect stock price changes, such as analyzing investor sentiment from text [28–30] or constructing an inter-stock relation graph to model the comovements of stocks [31–33] and treat the heterophilic graph [34]. Recent work on causal spatiotemporal series modeling has also brought new insights [35,36]. To better simulate the time-varying market, the Routing Adaptor mechanism [37] and Mixture of Experts (MOE) [38] are regarded as conditional modeling methods to adapt to the stock forecasting task on various market conditions. In summary, relying on the powerful representation capabilities of deep learning, these methods might better extract relevant features from the input and model the mapping between them and the output.

However, these methods share a common flaw. Due to market noise, like the opaque market, unexpected events, reckless trades, etc., stock price movement is uncertain, so there are indecipherable parts in future returns. In the past, the stock return ranking methods aimed to model the factors affecting stock price as much as possible, such as inter-stock relations [33,39], stock sectors [32,40], and market conditions [41]. However, they did not address how to model the inherent randomness in the process. Even with identical inputs, future returns may remain constant, suggesting that a direct mapping between input and output should not always exist. As a result, commonly used forecast models and learning objectives are less suitable for stock prediction, limiting performance in the noisy market

To solve the problem above, we propose a novel framework to analyze the stock return. Inspired by previous studies [41], which suggest that modeling the uncertainty of stock returns can aid in prediction, we assume that returns for different stocks at each time follow specific distributions with unknown parameters, with the future returns being samples drawn from them. The variances are the source of randomness, and the means represent the profitability, both vary with time. On this basis, we analyze the forecast model and optimization objectives commonly used in the past. From the perspective of maximum likelihood probability, we deduce that they are equal to introducing the assumption that all stock returns at each time obey a prior Gaussian distribution with a fixed standard deviation, and the forecast result is the mean of the return distribution. The market is volatile, making a fixed standard deviation unsuitable for accurately reflecting real market conditions [42]. In the paper, we relax this assumption by adding a forecast of the standard deviation of future returns from the input. We have enhanced the original model structures and proposed a new optimization objective. Additionally, we conducted experiments on the CSI 100, 300, and 500 indexes to verify the effectiveness of our approach compared to previous methods. These experiments also allowed us to expand our method on more general assumptions.

The main contributions of our work are summarized as follows.

- We propose a novel method based on the maximum likelihood principle to analyze the stock returns.
- We analyze previous methods and improve the model structure and learning objectives based on our method. Experiments on three indexes in the A-share market prove the profitability improvement of our method.
- We present an extension of our method and completed the pivotal analysis and mathematical derivation.

2. Related work

Past studies primarily focused on statistics and machine learning methods, which relied on assumptions about the data generation and required manufactured features. In recent years, with the development of deep learning in various fields, it has become popular in stock prediction tasks, and its representation and generalization capabilities have brought great benefits.

2.1. Statistical and machine learning method for stock forecasting

Statistical methods often rely on assumptions about the data generation process to construct corresponding models. ARIMA, a standard model for handling non-stationary time series, incorporates autoregression, differencing, and moving average operations. This model has been applied to predict stock movements in the NYSE and NSE stock markets and has shown competitive performance, particularly in short-term forecasting [3]. On the other hand, the Kalman filter, which tracks time series signals through

state equations, proves feasible for tracking future price changes using well-designed stock price-based state-space models [4]. The results demonstrate that the Kalman filter, while simple, is an effective method. Prophet is a comprehensive algorithm that decomposes time series data into trend, seasonal, holiday, and residual components. This versatility has made it effective in stock forecasting tasks [5].

Machine learning methods, in contrast to some traditional statistical approaches, do not require assumptions about the data generation process. Instead, they focus on capturing the mapping rules between inputs and outputs. SVM, a classic discriminant method, has been utilized to predict the direction of weekly NIKKEI 225 index movements. It has demonstrated superior performance compared to other statistical methods [8]. As the most commonly used boosting method, XGBoost is a strong contender for the Kaggle competition championship and outperformed many methods on stock price prediction tasks [9]. As a prominent example of generative methods, high-order Hidden Markov Models (HMM) have been employed to model the Markov properties of financial time series, achieving satisfactory results [6]. Additionally, unsupervised methods have been utilized to preprocess the time series data, aiming to reduce the impact of noise in the inputs [12].

2.2. Deep learning for stock forecasting

Deep learning methods have recently gained significant attention in stock prediction. RNNs and their variants, known for their memory mechanism, are particularly noted for modeling sequences with Markovian properties. To address the limitations of RNNs in modeling long-term dependencies, Qin et al. introduced the DA-RNN method, which combines an attention mechanism with RNNs for enhanced multi-step forecasting [16]. In a novel approach, the State Frequency Memory (SFM) LSTM prototype was proposed for modeling stock movements at multiple frequency granularities, demonstrating superior performance over traditional LSTM [43].

Unlike RNNs, CNNs exhibit robustness to local noise and excel in learning local features. This makes them well-suited for data from diverse sources and effective in various market conditions [16]. With the rise of the transformer structure in multiple fields, the HMG-TF model — a transformer-based approach — was developed to test its efficacy in stock forecasting. Applied to the Nasdaq and CSI 500 indexes, it showed comparative effectiveness [22]. Additionally, some studies have explored the combination of ARIMA, CNN, and LSTM to predict both linear and nonlinear aspects of price movements. These studies confirm that the CNN and RNN combination can effectively model short-term and long-term patterns for nonlinear series forecasting [27].

Many factors influencing stock prices extend beyond historical prices alone. To reduce forecasting uncertainty, some studies have incorporated prior knowledge of the stock market. Feng et al. for instance, utilized first- and second-order relations from Wikipedia to construct graphs, applying LSTM and Temporal Graph Convolutional Network (GCN) to model price-based and relation-based features. They validated the effectiveness of this approach on the NASDAQ and NYSE indexes [31].

Yoo et al. explored the influence of market movements on individual stocks by introducing index trading movements into transformer structures, enabling the dynamic construction of asymmetric inter-stock relations for forecasting purposes [26]. Additionally, the use of the Dynamic Time Warping (DTW) algorithm to calculate correlations between historical prices of different stocks as relation weights has proven to be a feasible strategy for stock prediction [24].

Liu et al. took a different approach by employing the Non-negative Matrix Factorization (NMF) algorithm to calculate sector-based embeddings of stocks using sector information and historical data. They introduced the Channel-Temporal Dual Attention Module (CTAM) for return forecasting [32]. Similarly, recognizing the interconnectedness of stocks in different industries, some studies have found value in constructing both intra- and inter-sector relation graphs [33].

Yang et al. proposed a method to avoid the need to gather and update additional information. They decomposed relations into short- and long-term components, encouraging the model to self-learn these aspects from the input price series and the total training set [43]. To address the limitation of static prior knowledge in representing the dynamic nature of inter-stock relations, Lei et al. introduced the Dynamic Routing Graph Attention Network (DR-GAT). This model integrates prior knowledge with price similarity to construct relation graphs for stock forecasting dynamically. Their experiments demonstrated the method's superiority on the NASDAQ and NYSE [44]. To eliminate the distraction effect caused by dissimilar neighbors in the graph, a causal graph attention network is utilized in the heterophilic graphs to process different kinds of neighbor nodes [34].

In summary, these methods aim to model the input–output mapping for prediction purposes. They approach this by making assumptions about the data generation process, incorporating prior knowledge to reduce uncertainty, and employing suitable network structures to navigate the noisy stock market environment better. Our approach, however, diverges from previous work by focusing on the uncertainty of the returns, viewed as the output. We seek to address the errors arising from the random aspects of labels, offering a distinct perspective and a valuable addition to existing methodologies.

3. Methods

3.1. Problem formulation

To mitigate the interday inventory risk, we plan to trade intraday, buy some assets at the opening time, and sell them when closing. So our goal is to forecast the intraday return ranking of all candidate stocks and build portfolios with the highest returns to trade.

For stock i at trading time t, the input series is the N historical trading prices $X_i^t = [p_i^{t-N}, \dots, p_i^{t-1}]$. $p_i^t = [p_{i,o}^t, p_{i,c}^t, p_{i,h}^t, p_{i,l}]^\top \in \mathbf{R}^4$ represents the corresponding opening, closing, high, and low prices. The forecasting output and real returns represent $\hat{r}_i^t$ and r_i^t, and

form as follows.

$$\begin{aligned} \hat{r}_i^t &= \mathcal{F}(p_i^{t-N}, \dots, p_i^{t-1}|\theta) \\ r_i^t &= \frac{p_{i,c}^t}{p_{i,o}^t} - 1.0 \end{aligned} \tag{1}$$

where $\mathcal{F}$ represents the prediction model, and θ is the trainable parameters.

3.2. Return and objective modeling

Many studies consider predicting the order of stock returns to profit, and the most commonly used objective function is proposed in paper [31] called **PR-Loss** which consists of the pointwise regression and pairwise ranking-sensitive items. We regard there are M stocks as candidates to trade. For simplicity, we ignore the superscript t. The specifics are as follows.

$$\mathcal{L}_{\mathcal{PR}}(r, \hat{r}) = \underbrace{\sum_{i=1}^{M}(r_i - \hat{r}_i)^2}_{Pointwise} + \alpha \underbrace{\sum_{i=1}^{M}\sum_{j=1}^{M} max(0, -(\hat{r}_i - \hat{r}_j)(r_i - r_j))}_{Pairwise} \tag{2}$$

where the first item is the regression tasks for forecasting to avoid some trivial solution and the second term is for return ranking, which is the focus of trading [39]. We believe that the PR-Loss objective contains some implicit modeling for returns, so we try to analyze it.

To model the randomness of stock price movements caused by market noise, we assume all stocks' returns on every trading day follow an unknown Gaussian distribution. The means and standard deviations of different stocks at various times are uncertain, but they are relevant to historical information, and we may capture them from the input. When we obtain them, the returns are determined, and we can trade relying on them. For stock i at time t, the return distribution is represented as $\hat{r}_i^t \sim \mathcal{N}(\mu_i^t, \sigma_i^t)$, and μ_i^t and σ_i^t is the mean and standard deviation, which is calculated by models from historical trading prices as input. In the following, we omit the superscript t for simplicity.

According to the Maximum Likelihood Estimation (MLE) method, we need to adjust the distribution parameters to make the observed data as likely as possible. For returns, the observed data are $(X_1, r_1), \dots, (X_M, r_M)$, and the objectives are expressed as follows:

$$\begin{aligned} &\max_{\theta} \prod_{i=1}^{M} \frac{1}{\sqrt{2\pi}\sigma_i} \exp(-\frac{(r_i - \mu_i)^2}{2\sigma_i^2}) \\ &s.t. \quad \forall i \in \{1, \dots, M\}, \quad \mu_i, \sigma_i = \mathcal{F}(X_i, \theta) \end{aligned} \tag{3}$$

where $\mathcal{F}$ is the model to model return distributions, and θ is trainable parameters. We update θ to obtain the maximum observation probability, and the objective is equivalent to the following forms:

$$\begin{aligned} &\max_{\theta} \sum_{i=1}^{M} -(\frac{\log(2\pi)}{2} + \log(\sigma_i) + \frac{(r_i - \mu_i)^2}{2\sigma_i^2}) \\ &\Rightarrow \min_{\theta} \sum_{i=1}^{M} 2\log(\sigma_i) + \frac{(r_i - \mu_i)^2}{\sigma_i^2} \end{aligned} \tag{4}$$

Moreover, we consider the relative differences in returns among different stocks. We assume the returns of different stocks are independent, so the return difference $\hat{r}_i - \hat{r}_j$ also satisfies the Gaussian distribution $\mathcal{N}(\mu_i - \mu_j, \sqrt{\sigma_i^2 + \sigma_j^2})$. We utilize $\boldsymbol{\Phi} = \{(1,2), \dots, (M-1, M)\}$ to represent all existing stock pairs, and MLE for return differences as follows:

$$\begin{aligned} &\max_{\theta} \prod_{(i,j)\in\boldsymbol{\Phi}} \frac{1}{\sqrt{2\pi(\sigma_i^2 + \sigma_j^2)}} \exp(-\frac{((r_i - r_j) - (\mu_i - \mu_j))^2}{2(\sigma_i^2 + \sigma_j^2)}) \\ &s.t. \quad \forall i \in \{1, \dots, M\}, \quad \mu_i, \sigma_i = \mathcal{F}(X_i, \theta) \end{aligned} \tag{5}$$

The objective is equivalent to the following forms:

$$\begin{aligned} &\max_{\theta} \sum_{(i,j)\in\boldsymbol{\Phi}} -(\log(2\pi) + \log(\sigma_i^2 + \sigma_j^2) + \frac{((r_i - r_j) - (\mu_i - \mu_j))^2}{\sigma_i^2 + \sigma_j^2}) \\ &\Rightarrow \min_{\theta} \sum_{(i,j)\in\boldsymbol{\Phi}} \log(\sigma_i^2 + \sigma_j^2) + \frac{((r_i - r_j) - (\mu_i - \mu_j))^2}{\sigma_i^2 + \sigma_j^2} \\ &\Rightarrow \min_{\theta} \sum_{(i,j)\in\boldsymbol{\Phi}} \log(\sigma_i^2 + \sigma_j^2) + \frac{(\mu_i - \mu_j)^2 - 2(r_i - r_j)(\mu_i - \mu_j) + (r_i - r_j)^2}{\sigma_i^2 + \sigma_j^2} \end{aligned} \tag{6}$$

If we assume that the distribution of the different returns has the same and fixed standard deviation, i.e., $\sigma_1 = \sigma_2 = \cdots = \sigma_M = \sigma$. The formulas (4) and (6) are equivalent to the transformed forms.

$$\min_{\theta} \sum_{i=1}^{M} (r_i - \mu_i)^2 \tag{7}$$

$$\min_{\theta} \sum_{(i,j)\in\Phi} (\mu_i - \mu_j)^2 - \sum_{(i,j)\in\Phi} 2(r_i - r_j)(\mu_i - \mu_j) \tag{8}$$

Next, modify form (8) and combine it with (7), and we can get the objective (2), where μ_i plays the role of $\hat{r_i}$. To obtain more accurate results, we believe that the returns satisfy **Gaussian distribution** and both the μ and σ can be estimated, and propose a new objective GPR-Loss, as follows:

$$L_{GPR}(r,\mu,\sigma) = \sum_{i=1}^{M} (2\log(\sigma_i) + \frac{(r_i - \mu_i)^2}{\sigma_i^2} + \alpha \sum_{j=1}^{M} \log(\sigma_i^2 + \sigma_j^2) + \frac{(r_i - r_j - \mu_i + \mu_j)^2}{\sigma_i^2 + \sigma_j^2}) \tag{9}$$

Considering that we do not want the forecasting return to be too uncertain, we **Constrain** the obtained standard deviation and propose a CGPR-Loss.

$$L_{CGPR}(r,\mu,\sigma) = L_{GPR}(r,\mu,\sigma) + \sum_{i=1}^{M} \sigma_i \tag{10}$$

3.3. Model structure

Most deep learning models can be divided into two components: the encoder part is for feature extraction from inputs, and the decoder part is for forecast returns. Past work has mainly focused on designing and combining different model structures for better features, and the decoder part often only requires a simple MLP for prediction. We found the CNN and RNN structures are complimentary to process time series, so the combination is better [20,25,27,45]. The transformer-based structure can model the long-distance correlations of inputs well, which is adequate for better forecast performance [22–24]. Therefore, we adopted a structure combining them to verify the framework proposed in this paper. The model consists of three parts: the *Inception-GRU Encoder* Module, *Transformer-Based Encoder* Module, and *Decoder* Module, and the specific overall structure is as Fig. 1 shows.

3.3.1. Inception-GRU encoder module

We can view the convolution operations as filtering in signal processing to smooth the input. The Inception structure utilized in this paper consists of the 1×, 3×, and 5× parts to extract different time-span moving average features. Then, these features are concatenated and input chronologically to a three-layer GRU structure to obtain the timing-embedding f_t corresponding to the last input position.

GRU is a variant of LSTM with two gates: update gate z_t and reset gate r_t. The process is as follows.

$$\begin{aligned} z_t &= \delta(W_z \cdot [h_{t-1}, x_t]) \\ r_t &= \delta(W_r \cdot [h_{t-1}, x_t]) \\ \hat{h}_t &= \tanh(W \cdot [r_i \odot h_{i-1}, x_t]) \\ h_t &= ((1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t) \end{aligned} \tag{11}$$

where W_z, W_z and W are trainable parameters. x_t and h_t are the input and output features of time t. $\odot$ is the Hadamard produce.

3.3.2. Transformer-based encoder module

The Transformer model is first proposed for translation tasks to model contextual connections and gradually utilized in other tasks [46]. First, we add the input and position embeddings. Then, we pass them through Multi-Head Attention (MHA), Normalization, linear layer, and Normalization mechanism one by one several times to obtain the corresponding features. Finally, we get the attention-embedding f_a corresponding to the last input position. The MHA mechanism is the core of the Transformer model, which is also the key to its ability to model long-distance inputs and play a role in stock prediction tasks. The specific process is as follows.

$$\begin{aligned} MH - Attn(Q,K,V) &= W^o(Attn(Q_1,K_1,V_1) \oplus \cdots \oplus Attn(Q_H,K_H,V_H)) \\ Attn(Q_h,K_h,V_h) &= softmax(\frac{W^q Q_h \cdot (W^k K_h)^T}{\sqrt{d}}) \cdot W^v V_h \end{aligned} \tag{12}$$

where W^q, W^k, W^v and $W^o \in \mathbf{R}^{d\times d}$ are trainable parameters. d is the input dimension. H is the head number, and it is satisfied that it is divisible by the dimension. *softmax* is an activation function for normalization, as follows.

$$softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)} \tag{13}$$

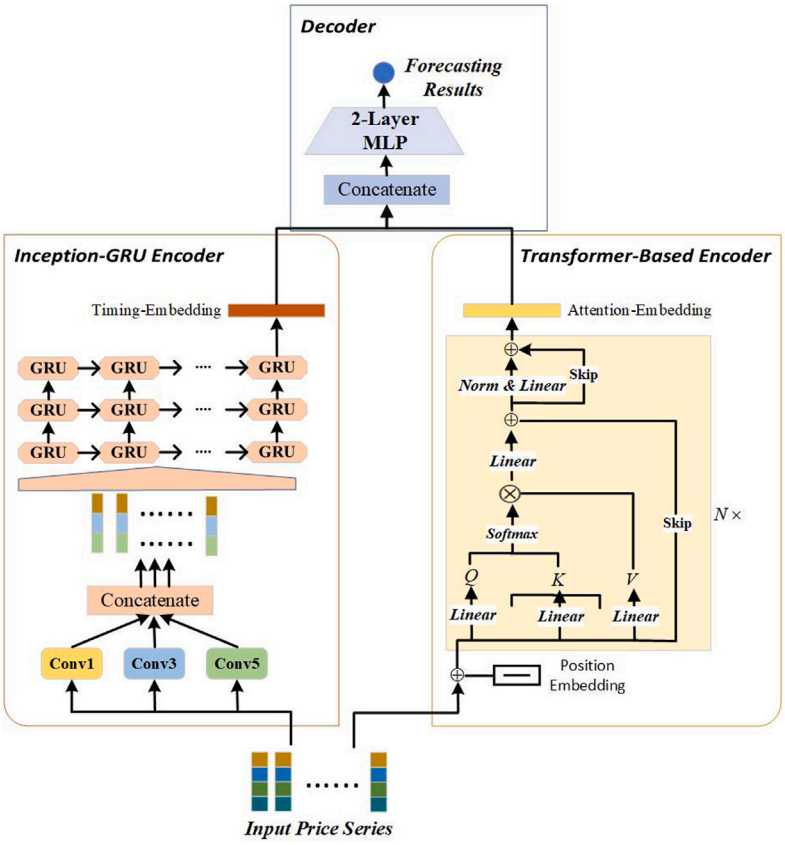


Fig. 1. The overall model structures.

3.3.3. Decoder module

The A-share market limits the range of stock returns, so we preprocess the returns to limit them to between −1 and 1. According to our viewpoint above, past methods often only needed to use an MLP as the Decoder module to predict the mean of the return distribution. Regarding the outputs of the Inception-GRU encoder and the Transformer-based encoder as input, the process is as follows:

$$o = 2.0 \times \delta(MLP(f_t, f_a)) - 1.0 \tag{14}$$

where formula (14) represents the past process, o is the prediction results, *MLP* is the trainable Decoder module. δ represents the Sigmoid function that transforms the input from 0 to 1. The method proposed in this paper requires forecasting the mean and standard deviation, so we use two MLPs to calculate them separately. The specific processes are as follows:

$$\begin{aligned} o_m &= \delta(MLP1(f_t, f_a)) - 0.5 \\ o_{sd} &= 0.4 \times \delta(MLP2(f_t, f_a)) \end{aligned} \tag{15}$$

Considering the range of returns, we set some restriction conditions to make the training process easier, so the mean results range from −0.5 to 0.5, and the standard deviation results range from 0 to 0.4.

4. Experiments

4.1. Data preparation

We take the members of the CSI100, CSI300, and CSI500 Chinese stock indexes from Jun 12, 2010, to Jun 12, 2020, as the research object [39], which represent the large-cap, blue-chip, small- and mid-cap stocks and include different bull and bear market status. We eliminated the stocks under five years old for the entire period. To verify if the method is robust for the volatile and random stock environment, we utilize the rolling-window method for backtesting [39,47]. To simulate the actual trading conditions, we set the model to be updated every three months [39] and determine the length of training data by several experiments. For a period, the data of the first 65 trading days (nearly three months) are validation data, the subsequent 504 trading days (roughly

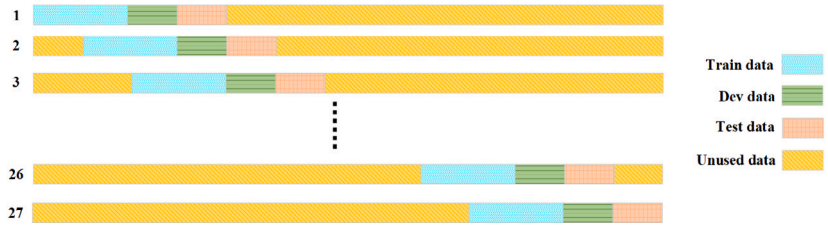


Fig. 2. The rolling-window method for splitting training, validation, and test data.

Table 1
The data set overall statistics.

Dataset	Samples × Days	Stocks Per-day	Intraday return range (%)	Average intraday return (‰)
CSI 100	174 141	65~77	−7.33~8.31	1.305
CSI 300	508 030	193~222	−7.56~8.18	1.314
CSI 500	762 275	297~331	−9.12~7.61	1.483

two years) are for training, and we test our model on the next 65 trading days. In this paper, the whole period consists of 27 phases for experiments, and Fig. 2 visually illustrates the splitting process.

After several experiments, we chose the input historical price series length to be $N = 15$, because longer inputs cannot improve performance but bring more overhead. Since we cannot prove that a stock's absolute value does not affect the stock price movement, and deep learning can extract features from raw data, we do not normalize the data and use raw prices directly as input. The statistical information corresponding to the three indexes is shown in Table 1, including the number of stocks and Samples×Days of indexes, and the range of intraday returns, and the average intraday return of the individual stocks of the index. We can observe the return distributions of the three kinds of stocks.

4.2. Experimental details

In this subsection, we depict the method of participating in comparison, the construction of evaluation metrics, the selection of some experimental hyperparameters, and the experimental environment.

4.2.1. Comparison methods

Regarding the novel analysis framework proposed in this paper, we redesign the decoder module (Section 3.3.3) and present two relative optimization objectives, called GPR-Loss and CGPR-Loss (Section 3.2). To verify the effectiveness of our methods, we compare the performance separately relative to the original model with classic *MAE*, *MSE*, and *PR-Loss* as loss functions.

The model structure we used is in Section 3.3, and the baseline and the modified model have the same encoder structures. It combines the Inception-GRU and Transformer structures as the encoder module to extract relevant features and utilizes the MLP as the decoder module.

As mentioned above, the baseline and modified models share the same encoder structure, which comprises an Inception-GRU encoder and a transformer-based encoder module. The Inception-GRU encoder includes both an Inception component and a GRU component, with an input dimension of 4. The Inception component features three 15-channel convolutional kernels with sizes of 1, 3, and 5, while the GRU component is a three-layer GRU model with input, hidden, and output sizes all set to 36. The Transformer module consists of 4 layers, each with a hidden size of 36 and 4 attention heads.

4.2.2. Evaluation metrics

Our purpose is to forecast the returned ranking of candidate stocks and select the assets with the highest return to build a portfolio to maximize profits. Therefore, we can regard the problem as a ranking task, and we consider analyzing the results from two aspects: ranking and profitability performance. Suppose we choose the stocks with Top-K forecasting ranking results, we can calculate the corresponding metrics.

Regarding ranking performance, we use Mean Reciprocal Rank (MRR) and average Investment Return Ratio (IRR) as metrics measurement, which is already used in this task and larger values are better [31]. The former is widely used for ranking performance comparison, and the latter represents the significance of returns for top-ranked stocks. Assume that the actual rankings of the predicted top-k stocks are $o_1, \ldots, o_K$, and the corresponding real returns are $r_1, \ldots, r_K$, and the average return of all candidate stocks is r_{avg}. The results of the two indicators are expressed as follows.

$$MRR_K = \frac{1}{K}\sum_{i=1}^{K}\frac{1}{o_i} \tag{16}$$

$$IRR_K = \frac{1}{K}\sum_{i=1}^{K} r_i - r_{avg} \tag{17}$$

Table 2
The return ranking performances of all methods.

K	Methods	CSI 100		CSI 300		CSI 500	
		MRR	IRR (%)	MRR	IRR (%)	MRR	IRR (%)
1	MAE	0.067	0.081	0.046***	0.048	0.034***	0.108*
	MSE	0.064	0.048	0.041***	0.069	0.039***	0.137*
	PR	0.071	0.159**	0.036**	0.146**	0.035***	0.082
	GPR	**0.118*****	**0.216*****	0.069***	0.294***	0.063***	**0.558*****
	CGPR	0.111***	0.195***	**0.074*****	**0.45*****	**0.068*****	0.461***
5	MAE	0.064	0.041	0.035***	0.107***	0.026***	0.116***
	MSE	0.061	0.037	0.032***	0.094***	0.03***	0.082**
	PR	0.068*	0.092***	0.032***	0.087**	0.026***	0.06*
	GPR	0.103***	0.095**	0.057***	0.205***	0.053***	**0.383*****
	CGPR	**0.103*****	**0.113*****	**0.066*****	**0.228*****	**0.056*****	0.358***
20	MAE	0.066**	0.021	0.029***	0.058***	0.021*	0.069***
	MSE	0.061**	0.02**	0.029***	0.028***	0.023***	0.065***
	PR	0.065	0.036**	0.03***	0.036*	0.022***	0.069***
	GPR	**0.084*****	**0.05*****	0.047***	**0.129*****	0.042***	**0.214*****
	CGPR	0.083***	0.045***	**0.051*****	0.114***	**0.043*****	0.206***

* Depict the p-value of significant test is less than 0.05.
** Depict the p-value of significant test is less than 0.02.
*** Depict the p-value of significant test is less than 0.001.

As for the profitability measurement, we utilize some widely used indicators including excess Annualized Return (AR), Sharpe Ratio (SR), and Maximum Drawdown (MD), which reflect the average returns, risk compensation, and risk exposure measurements respectively. Among them, *AR* is the one we are most concerned about because it directly reflects profitability. The specific calculation is as follows.

$$AR_K = \left(\frac{P_K - P_0}{P_0}\right)^{ret} - \left(\frac{P_{avg} - P_0}{P_0}\right)^{ret} \tag{18}$$

$$SR_K = \frac{\mathbb{E}(R_K) - R_f}{\sigma_K} \tag{19}$$

$$MD_K = 1 - \min_i \frac{P_K^i}{\max_j P_K^j} \quad \forall j < i, i \in [1, \ldots, T] \tag{20}$$

where P_0 and P_K represent the portfolio values at the start and the end of the trading process. P_{avg} is the value of a baseline portfolio at the end for comparison, and the special portfolio represents trading all candidate stocks with the same value. *ret* represents the reciprocal of the trading process length in years. $\mathbb{E}(R_K)$ and σ_K denote the expectation and standard deviation of excess returns in the trading process, and R_f is the risk-free rate and we set 0 in this paper. The whole trading time is set from 1 to T, and P_K^i represents the values at time i, so *MD* depicts the largest loss in the whole trading process.

4.2.3. Settings

In this paper, we utilize the classic **Adam** optimizer to train models. The initial learning rate is selected from the set $\{0.001, 0.0001\}$, with the weight decay is 0.001. The hyperparameter α in the *PR-Loss*, *GPR-Loss* and *CGPR-Loss* are searched from the set $\{0.1, 1.0, 10.0\}$. We use a grid search method to determine the parameter values for each method and show the best results.

The hardware environment consists of 2 T P100 GPUs with 12 GB RAM and an Intel Xeon CPU Processor with 128.0 GB RAM, and the software environment is Pytorch.

4.3. Main results

To demonstrate the effectiveness of our proposed objectives and the corresponding framework, we used different objectives to train models on three indexes and took Top-1, Top-5, and Top-20 forecasting stocks to build a portfolio. Table 2 shows the mean of *MRR* and *IRR* of the obtained results of different methods and its corresponding significance test results to illustrate the ranking performance of different methods. Table 3 shows the corresponding *AR*, *SR*, and *MD* indicators to compare their profitability.

First of all, compared with other classic objectives, the MRR and IRR performances corresponding to our proposed objectives *GPR* and *CGPR* are obviously improved for different K values on the three indexes and always rank first and second. Also, the statistical significance of our methods is higher. These results indicate that our objectives perform significantly better than classic methods and are more stable. In addition, as the K value increases, both MRR and IRR become smaller, especially for our methods, which indicates that the stocks with top forecasting returns have better real returns, in line with our expectations.

Then, the AR indicators of our proposed method with different K values on the three indexes are significantly higher than other methods, and the corresponding SR indicators are also the best in most cases, especially on the CSI500, which demonstrates that

Table 3
The trading performance of all methods on the test phases.

K	Methods	CSI 100			CSI 300			CSI 500		
		AR (%)	SR	MD	AR (%)	SR	MD	AR (%)	SR	MD
1	MAE	17.57	0.682	0.448	30.42	0.861	0.596	19.86	0.633	0.574
	MSE	8.48	0.433	0.501	11.44	0.475	0.58	28.32	0.779	0.583
	PR	40.73	1.18	**0.303**	35.04	0.983	0.415	12.18	0.478	0.526
	GPR	**57.83**	**1.31**	0.56	86.16	1.497	**0.314**	**239.32**	**2.329**	**0.511**
	CGPR	51.1	1.249	0.353	**171.01**	**2.128**	0.323	164.0	1.874	0.521
5	MAE	9.69	0.724	0.233	28.39	1.354	0.192	31.16	1.456	0.27
	MSE	8.71	0.71	0.501	24.44	1.276	0.26	20.19	1.005	0.299
	PR	24.26	1.437	**0.167**	22.5	1.189	0.238	14.28	0.803	0.351
	GPR	24.56	1.21	0.299	61.28	1.889	0.201	**150.45**	**3.229**	0.266
	CGPR	**30.35**	**1.445**	0.215	**71.06**	**2.101**	**0.191**	135.2	2.939	**0.218**
20	MAE	5.02	0.652	0.162	14.93	1.265	**0.106**	17.98	1.572	0.146
	MSE	5.04	0.694	**0.084**	15.03	1.335	**0.106**	16.77	1.397	0.171
	P	9.18	1.161	0.109	8.94	0.826	0.182	18.21	1.591	0.199
	GPR	**11.55**	**1.164**	0.124	**36.52**	**1.924**	0.115	**69.0**	**3.228**	**0.112**
	CGPR	11.549	1.164	0.124	31.44	1.749	0.159	65.43	3.023	0.128

our methods have the best profitability and risk-compensation ability. Besides, compared with other methods, the MD indicators of our methods can also achieve good results, especially performing best on the CSI500 index. This indicates that our method has good risk resistance and does not bring about an increase in potential losses while obtaining significantly better returns. It is worth noting that as K increases, the values of AR tend to increase and MD decrease, which reflects a common sense in economics that it is often necessary to sacrifice the stability of strategy profits and take greater risks to pursue higher profits. Therefore, for people with different risk preferences, different trading strategies need to be considered under the same forecasting results.

Fig. 3 shows the excess profits and MRR indicators during the test period. We can clearly see that our method can accumulate more profits and obtain better ranking result distribution compared to the classic *PR-Loss* as an objective on the condition of different K values. A larger K value brings smaller return fluctuations and a more concentrated MRR distribution. Also, the return of trading on the CSI500 is higher than that of the CSI100 in the results, which may be due to the difference in the size effect [48,49].

4.4. The influence of K value on profits

In the above, we conclude that the K value would affect the profitability of the strategy. In this section, we discuss the specific influence of the K value on profits. We set the range of K from 1 to 20 on the three indexes, and compare the results of utilizing the classic *PR-Loss* and our proposed *GPR-Loss* and *CGPR-Loss* as the model training objectives, respectively. Fig. 4 shows the specific AR performance comparison.

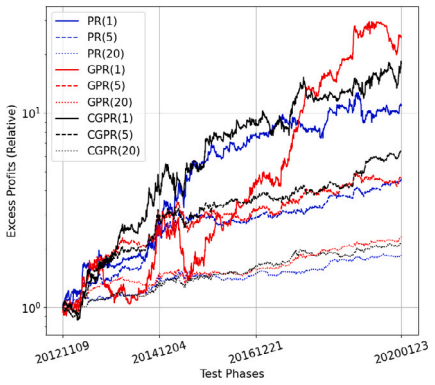
We can find two obvious conclusions: the profitability of the two objectives proposed is significantly better than the classic *PR-Loss* on different K conditions, and the effects are more obvious when the K value is smaller. As the K value increases, the profitability shows a gradually decreasing trend, but the excess returns still significantly exceed 0, which means that the portfolio can obtain returns that exceed the market.

These conclusions bring us some investment inspiration: first of all, our objectives can directly replace the *PR-Loss* to enhance profitability; then, we should choose stocks with a higher forecasting ranking mean when building an investment portfolio. When trading on the market, we can obtain more profits under different levels of risk by changing the value of K. Risk and return are a pair of indicators with negative correlation, we need to carefully choose the value of K to ensure the optimal result and exceed the market portfolios.

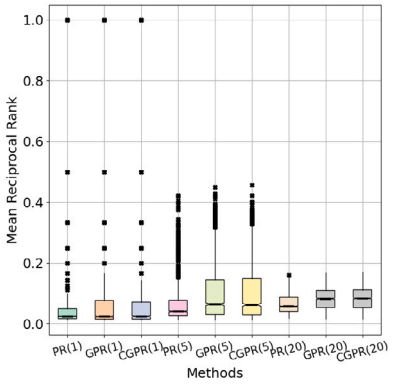
4.5. The role of standard-deviation for trading

The mean part in the forecasts is helpful for trading. In this section, we explore the role of the standard deviation part in the forecasts. Intuitively, the standard deviation represents the degree of uncertainty in the return distribution, so the larger the error between the forecasting mean and the observed real return, the larger the standard deviation should be. To verify it, we calculate the Pearson Correlation Coefficient (PCC) between the absolute value of the error and the standard deviation of the forecast results for each trading day. The PCC depicts the strength of the correlations of two series $X = [X_1, \dots, X_t]$ and $Y = [Y_1, \dots, Y_t]$ and is calculated as follows, and relevant statistical results on three indexes are shown in Fig. 5.

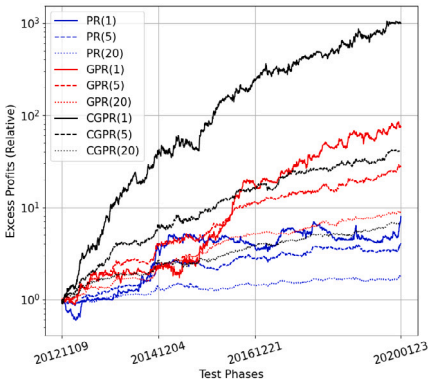
$$\mathbf{PCC}(X,Y) = \frac{\sum_{i=1}^{t}(\mathbf{X}_i - \bar{X})(\mathbf{Y}_i - \bar{Y})}{\sqrt{\sum_{i=1}^{t}(\mathbf{X}_i - \bar{X})^2}\sqrt{\sum_{i=1}^{t}(\mathbf{Y}_i - \bar{Y})^2}}$$
$$where \quad \bar{X} = \frac{1}{t}\sum_{i=1}^{t} X_i, \quad \bar{Y} = \frac{1}{t}\sum_{i=1}^{t} Y_i, \tag{21}$$



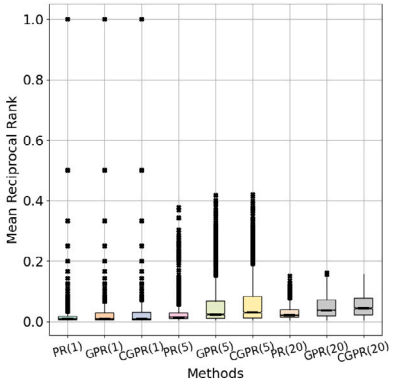
(a) Excess profits on CSI100.



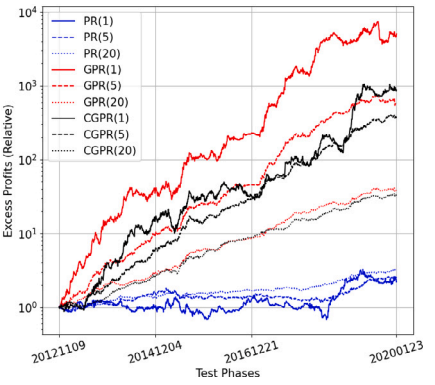
(b) MRR on CSI100.



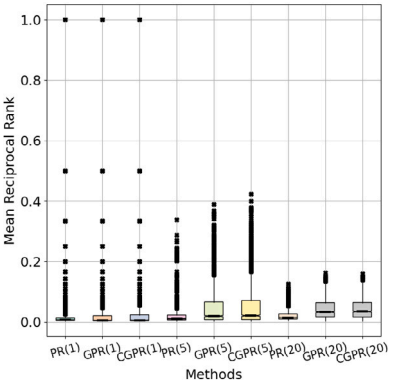
(c) Excess profits on CSI300.



(d) MRR on CSI300.



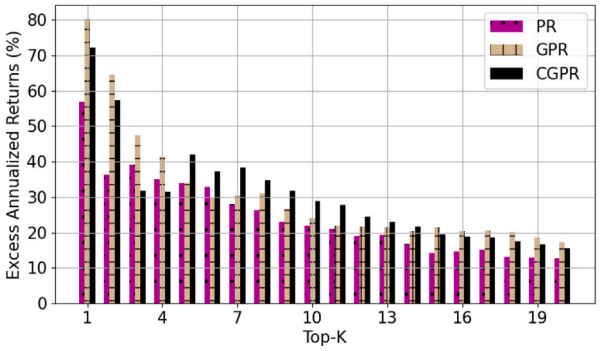
(e) Excess profits on CSI500.



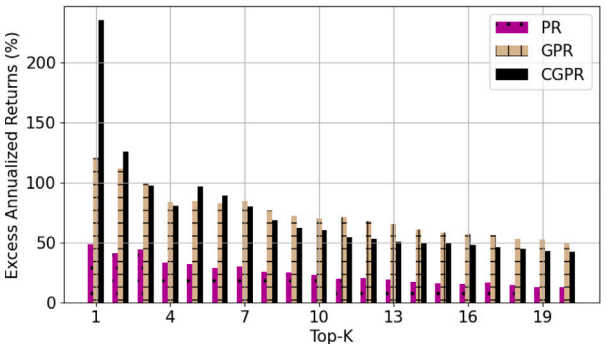
(f) MRR on CSI500.

Fig. 3. The excess profits and MRR comparison over the overall test period.

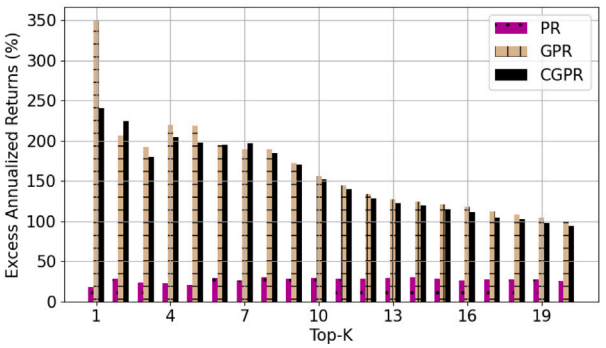
We can observe that most of the PCC between standard deviation and error are positive, indicating that there exist positive correlations between them. However, the mean of the PCC is around 0.2, depicting that their correlation is weak, possibly due to the difficulty in forecasting returns based on historical price series. To further utilize the standard deviation to enhance profitability, we divide the selected Top-K stocks into two parts with larger and smaller standard deviations, denoted as **-HS** and **-LS** respectively. In the case of different K values, the corresponding returns of the two parts are shown in Fig. 6.



(a) AR on CSI100.

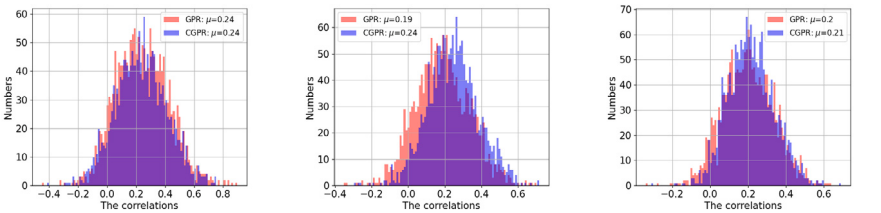


(b) AR on CSI300.



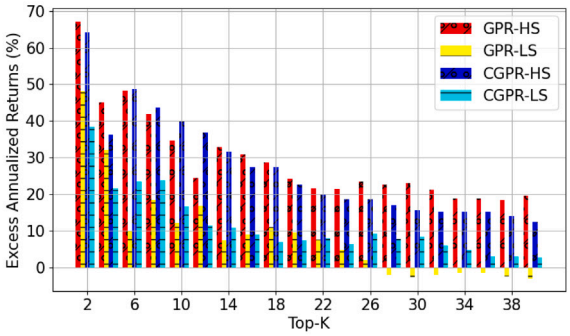
(c) AR on CSI500.

Fig. 4. The AR performances with different K over the overall test period.

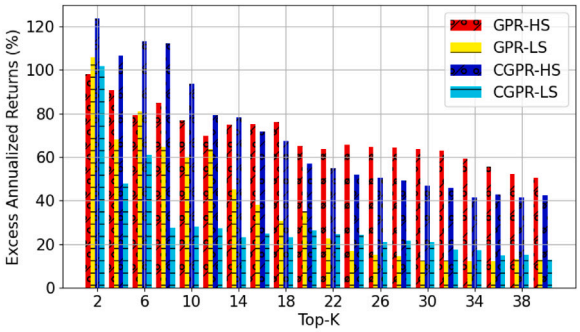


(a) correlations on CSI100. (b) correlations on CSI300. (c) correlations on CSI500.

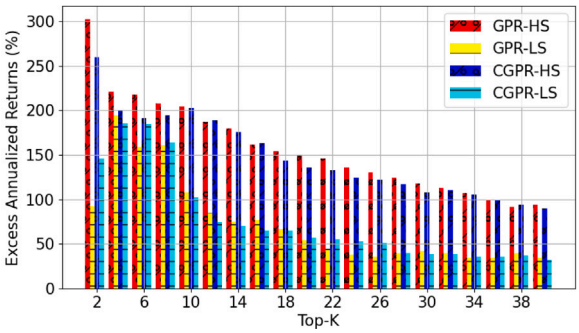
Fig. 5. The Pearson correlation coefficients between σ and forecasting error.



(a) AR on CSI100.



(b) AR on CSI300.



(c) AR on CSI500.

Fig. 6. The AR performance comparison with low and high standard-deviation assets.

It is obvious that with *GPR-Loss* and *CGPR-Loss* as the objectives, the returns of stocks with higher standard deviations in the Top-K portfolio are significantly higher than those with lower standard deviations. To some extent, we can assume that stocks with higher returns have larger forecast mean and standard deviation forecasting results, and the two promote each other. Compared with *PR-Loss*, the proposed method is equivalent to dynamically weighting the training data. Since the unpredictable part of the return accounts for a large proportion, the higher the true return, the larger the prediction error, and will play a more important role during the training process. Our methods reduce the weight of training data with higher returns, prevent samples from overfitting on data with higher returns, and therefore achieve better performance. In the future, we can improve by reducing the impact of noise, and modeling the relationship between sample uncertainty and the training weights is possible.

Our work is inspired by previous Ref. [39], which essentially weights training samples during the training process by curriculum learning. Similar but more in-depth, we consider more analysis based on the objectives. In this paper, it is assumed that returns follow a Gaussian distribution, which is equivalent to modeling the unexplainable return parts caused by environmental noise as a Gaussian distribution. In the next section, we release this assumption and explore a possible direction for future improvements.

4.6. Extension analysis

The real distribution of stock returns may not obey Gaussian distribution, so we can consider using a Gaussian Mixture Model (GMM) to model its real distribution due to its ability to model any form of distribution [50]. For variable x consists of P components, it is represented as follows.

$$\begin{aligned} p(x) &= \sum_{p=1}^{P} \omega_p \cdot \frac{1}{\sqrt{2\pi\sigma_p^2}} e^{\frac{-(x-\mu_p)^2}{2\sigma_p^2}} \\ s.t. \quad & \sum_{p=1}^{P} \omega_p = 1 \end{aligned} \tag{22}$$

where w_i, μ_i, and σ_i separately represent the weight, mean and standard-deviation of the ith components, and the distribution includes P components.

From a more general perspective, the output of our Decoder module is expanded from 2 to $3K$, representing P groups of weights, means and standard deviations, and all the outputs satisfy the relevant constraints. We first maximize the probability of observed returns using the maximum likelihood criterion, expressed as follows.

$$\begin{aligned} &\max_{\theta} \quad \prod_{i=1}^{M} \sum_{m=1}^{P} \omega_{i,m} \frac{1}{\sqrt{2\pi}\sigma_{i,m}} \dot{\exp}(-\frac{(r_i-\mu_{i,m})^2}{2\sigma_{i,m}^2}) \\ &\Rightarrow \quad \max_{\theta} \sum_{i=1}^{M} \log(\sum_{m=1}^{P} \frac{\omega_{i,m}}{\sigma_{i,m}} \exp(-\frac{(r_i-\mu_{i,m})^2}{2\sigma_{i,m}^2})) \\ &\Rightarrow \quad \min_{\theta} \sum_{i=1}^{M} (\log\sigma_{i,t} + \frac{(r_i-\mu_{i,t})^2}{2\sigma_{i,t}^2} - \log\omega_{i,t} - \log\sum_{m=1}^{P}\beta_m) \\ &where \quad t = \mathbf{argmax}_{m}(\frac{\omega_{i,m}}{\sigma_{i,m}} \exp(-\frac{(r_i-\mu_{i,m})^2}{2\sigma_{i,m}^2})) \\ &\beta_P = \frac{\omega_{i,P}\sigma_{i,t}}{\omega_{i,t}\sigma_{i,P}} \dot{\exp}(\frac{(r_i-\mu_{i,t})^2}{2\sigma_{i,t}^2} - \frac{(r_i-\mu_{i,P})^2}{2\sigma_{i,P}^2}) \end{aligned} \tag{23}$$

where P represents the number of components of the return distribution, and the optimal value requires multiple experiments to determine. In addition, we should limit the range of means of different components to avoid overlapping the distributions of different components.

Then, we continue to maximize the distribution of return differences among individual stocks as mentioned above, and the specific process can be expressed as follows:

$$\begin{aligned} &\max_{\theta} \prod_{(i,j)\in\Phi} \sum_{m=1}^{P} \sum_{n=1}^{P} \frac{\omega_{i,m}\dot{\omega}_{j,n}}{\sqrt{2\pi(\sigma_{i,m}^2+\sigma_{j,n}^2)}} \dot{\exp}(-\frac{((r_i-r_j)-(\mu_{i,m}-\mu_{j,n}))^2}{2(\sigma_{i,m}^2+\sigma_{j,n}^2)}) \\ &\Rightarrow \max_{\theta} \sum_{(i,j)\in\Phi} \log(\sum_{m=1}^{P}\sum_{n=1}^{P} \frac{\omega_{i,m}\dot{\omega}_{j,n}}{\sqrt{\sigma_{i,m}^2+\sigma_{j,n}^2}} \exp\frac{-(r_i-r_j-\mu_{i,m}+\mu_{j,n})^2}{2(\sigma_{i,m}^2+\sigma_{j,n}^2)}) \\ &\Rightarrow \min_{\theta} \sum_{(i,j)\in\Phi} \frac{\log(\sigma_{i,t1}^2+\sigma_{j,t2}^2)}{2} + \frac{(r_i-r_j-\mu_{i,t1}+\mu_{j,t2})^2}{2(\sigma_{i,t1}^2+\sigma_{j,t2}^2)} - \log\omega_{i,t1} - \log\omega_{j,t2} - \log\sum_{m=1}^{P}\sum_{n=1}^{P}\beta_{m,n} \\ &where \quad t1,t2 = \mathbf{argmax}_{m,n}(\frac{\omega_{j,n}\omega_{i,m}}{\sqrt{\sigma_{i,m}^2+\sigma_{j,n}^2}} \exp\frac{-(r_i-r_j-\mu_{i,m}+\mu_{j,n})^2}{2(\sigma_{i,m}^2+\sigma_{j,n}^2)}) \\ &\beta_{m,n} = \frac{\omega_{i,m}\omega_{j,n}\sqrt{\sigma_{i,t1}^2+\sigma_{j,t2}^2}}{\omega_{i,t1}\omega_{j,t2}\sqrt{\sigma_{i,m}^2+\sigma_{j,n}^2}} \dot{\exp}(\frac{(r_i-r_j-\mu_{i,t1}-\mu_{i,t2})^2}{2(\sigma_{i,t1}^2+\sigma_{i,t2}^2)} - \frac{(r_i-r_j-\mu_{i,m}-\mu_{i,n})^2}{2(\sigma_{i,m}^2+\sigma_{i,n}^2)}) \end{aligned} \tag{24}$$

5. Conclusion

Considering the impact of market noise on stock returns, we present a novel analysis framework. It assumes the returns of different stocks at each time following Gaussian distribution to model the randomness of the market. The mean and standard deviation are unknown and we can obtain them from the historical input. In the framework, past studies are equivalent in that all returns follow a homoscedastic Gaussian distribution. In this paper, we extend it to the more general heteroscedastic situation and propose a revised model structure and corresponding objectives to forecast the mean and standard deviation of the return distribution based on the maximum likelihood principle. Experiment results on the CSI100, CSI300, and CSI500 indexes depict how

our methods improve forecasting performance on various metrics, proving the effectiveness of our framework. We demonstrate that the obtained mean and standard deviation are beneficial for making more profits. Finally, we extend the return distribution to a more generalized Gaussian Mixture Model, which can fit any distribution, and we deduce the corresponding learning objects.

Due to the limitations of computational sources, the more general GMM conditions are not tested. Next, we will consider relevant experiments. At the same time, introducing more multi-source data related to stock prices and studying their impact on the return distribution is also a possible research direction for us in the future.

CRediT authorship contribution statement

Jiahao Yang: Writing – original draft, Software, Methodology, Formal analysis, Conceptualization. **Ran Fang:** Validation, Methodology, Investigation. **Ming Zhang:** Visualization, Validation, Investigation. **Wenkai Zhang:** Writing – review & editing, Methodology, Formal analysis. **Jun Zhou:** Supervision, Resources, Project administration, Funding acquisition.

Declaration of competing interest

All authors disclosed no relevant relationships.

Acknowledgments

This work has been supported by The Youth Innovation Promotion Association of the Chinese Academy of Sciences (E1291902), Jun Zhou (2021025).

Data availability

Data will be made available on request.

References

[1] J.M. Poterba, L.H. Summers, Mean reversion in stock prices: Evidence and implications, NBER Work. Pap. Ser. (1987).
[2] A. Timmermann, C.W. Granger, Efficient market hypothesis and forecasting, Int. J. Forecast. 20 (1) (2004) 15–27.
[3] A.A. Ariyo, A.O. Adewumi, C.K. Ayo, Stock price prediction using the ARIMA model, in: 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, IEEE, 2014, pp. 106–112.
[4] H. Haleh, B.A. Moghaddam, S. Ebrahimijam, A new approach to forecasting stock price with EKF data fusion, Int. J. Trade Econ. Financ. 2 (2) (2011) 109.
[5] A. Garlapati, D.R. Krishna, K. Garlapati, U. Rahul, G. Narayanan, et al., Stock price prediction using Facebook Prophet and Arima models, in: 2021 6th International Conference for Convergence in Technology, I2CT, IEEE, 2021, pp. 1–7.
[6] M. Zhang, X. Jiang, Z. Fang, Y. Zeng, K. Xu, High-order Hidden Markov Model for trend prediction in financial time series, Phys. A 517 (2019) 1–12.
[7] Y. Kara, M.A. Boyacioglu, Ö.K. Baykan, Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange, Expert Syst. Appl. 38 (5) (2011) 5311–5319.
[8] W. Huang, Y. Nakamori, S.-Y. Wang, Forecasting stock market movement direction with support vector machine, Comput. Oper. Res. 32 (10) (2005) 2513–2522.
[9] K.K. Yun, S.W. Yoon, D. Won, Prediction of stock price direction using a hybrid GA-XGBoost algorithm with a three-stage feature engineering process, Expert Syst. Appl. 186 (2021) 115716.
[10] R.A. Kamble, Short and long term stock trend prediction using decision tree, in: 2017 International Conference on Intelligent Computing and Control Systems, ICICCS, IEEE, 2017, pp. 1371–1375.
[11] H. Huang, W. Zhang, G. Deng, J. Chen, Predicting stock trend using fourier transform and support vector regression, in: 2014 IEEE 17th International Conference on Computational Science and Engineering, IEEE, 2014, pp. 213–216.
[12] J. Nobre, R.F. Neves, Combining principal component analysis, discrete wavelet transform and XGBoost to trade in the financial markets, Expert Syst. Appl. 125 (2019) 181–194.
[13] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.
[14] E. Chong, C. Han, F.C. Park, Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies, Expert Syst. Appl. 83 (2017) 187–205.
[15] H. Sun, W. Rong, J. Zhang, Q. Liang, Z. Xiong, Stacked denoising autoencoder based stock market trend prediction via k-nearest neighbour data selection, in: Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24, Springer, 2017, pp. 882–892.
[16] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, G.W. Cottrell, A dual-stage attention-based recurrent neural network for time series prediction, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI, 2017, pp. 2627–2633.
[17] F. Feng, H. Chen, X. He, J. Ding, M. Sun, T.-S. Chua, Enhancing stock movement prediction with adversarial training, IJCAI (2019).
[18] Y. Liu, Z. Wang, B. Zheng, Application of regularized GRU-LSTM model in stock price prediction, in: 2019 IEEE 5th International Conference on Computer and Communications, ICCC, IEEE, 2019, pp. 1886–1890.
[19] S. Chen, H. He, Stock prediction using convolutional neural network, in: IOP Conference Series: Materials Science and Engineering, vol. 435, IOP Publishing, 2018, 012026.
[20] E. Hoseinzade, S. Haratizadeh, CNNpred: CNN-based stock market prediction using a diverse set of variables, Expert Syst. Appl. 129 (2019) 273–285.
[21] X. Zhou, Z. Pan, G. Hu, S. Tang, C. Zhao, Stock market prediction on high-frequency data using generative adversarial nets, Math. Probl. Eng. 2018 (2018) 1–11.
[22] Q. Ding, S. Wu, H. Sun, J. Guo, J. Guo, Hierarchical multi-scale Gaussian transformer for stock movement prediction, in: Proceedings of the 29th International Joint Conference on Artificial Intelligence, IJCAI, 2021, pp. 4640–4646.

[23] Q. Zhang, C. Qin, Y. Zhang, F. Bao, C. Zhang, P. Liu, Transformer-based attention network for stock movement prediction, Expert Syst. Appl. 202 (2022) 117239.
[24] H. Wang, T. Wang, S. Li, J. Zheng, S. Guan, W. Chen, Adaptive long-short pattern transformer for stock investment selection, in: Proceedings of the 31th International Joint Conference on Artificial Intelligence, IJCAI, 2022, pp. 3970–3977.
[25] W. Lu, J. Li, Y. Li, A. Sun, J. Wang, A CNN-LSTM-based model to forecast stock prices, Complexity 2020 (2020) 1–10.
[26] J. Yoo, Y. Soun, Y.-c. Park, U. Kang, Accurate multivariate stock movement prediction via data-axis transformer with multi-level contexts, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2037–2045.
[27] Y. Zhao, Z. Chen, Forecasting stock price movement: New evidence from a novel hybrid deep learning model, J. Asian Bus. Econ. Stud. 29 (2) (2022) 91–104.
[28] Y. Xu, S.B. Cohen, Stock movement prediction from tweets and historical prices, in: ACL, 2018.
[29] H. Maqsood, I. Mehmood, M. Maqsood, M. Yasir, S. Afzal, F. Aadil, M.M. Selim, K. Muhammad, A local and global event sentiment based efficient stock exchange forecasting using deep learning, Int. J. Inf. Manage. 50 (2020) 432–451.
[30] S. Li, Z. Tian, Y. Li, Residual long short-term memory network with multi-source and multi-frequency information fusion: An application to China's stock market, Inform. Sci. 622 (2023) 133–147.
[31] F. Feng, X. He, X. Wang, C. Luo, Y. Liu, T.-S. Chua, Temporal relational ranking for stock prediction, ACM Trans. Inf. Syst. 37 (2) (2019) http://dx.doi.org/10.1145/3309547.
[32] G. Song, T. Zhao, S. Wang, H. Wang, X. Li, Stock ranking prediction using a graph aggregation network based on stock price and stock relationship information, Inform. Sci. 643 (2023) 119236, http://dx.doi.org/10.1016/j.ins.2023.119236.
[33] Y.-L. Hsu, Y.-C. Tsai, C.-T. Li, FinGAT: Financial graph attention networks for recommending top-KK profitable stocks, IEEE Trans. Knowl. Data Eng. 35 (1) (2023) 469–481, http://dx.doi.org/10.1109/TKDE.2021.3079496.
[34] S. He, Q. Luo, X. Fu, L. Zhao, R. Du, H. Li, CAT: A causal graph attention network for trimming heterophilic graphs, Inform. Sci. (2024) 120916.
[35] K. Wang, H. Wu, Y. Duan, G. Zhang, K. Wang, X. Peng, Y. Zheng, Y. Liang, Y. Wang, NuwaDynamics: Discovering and updating in causal spatio-temporal modeling, in: The Twelfth International Conference on Learning Representations, 2024.
[36] Q. Luo, S. He, X. Han, Y. Wang, H. Li, LSTTN: A Long-Short Term Transformer-based spatiotemporal neural network for traffic flow forecasting, Knowl.-Based Syst. 293 (2024) 111637.
[37] H. Lin, D. Zhou, W. Liu, J. Bian, 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.
[38] S. Sun, X. Wang, W. Xue, X. Lou, B. An, Mastering stock markets with efficient mixture of diversified trading experts, 2023, pp. 2109—-2119.
[39] J. Yang, S. Feng, W. Zhang, M. Zhang, J. Zhou, P. Zhang, An efficient loss function and deep learning approach for ranking stock returns in the absence of prior knowledge, Inf. Process. Manage. 61 (1) (2024) 103579.
[40] J. Yang, M. Zhang, S. Feng, X. Zhang, X. Bai, A hierarchical deep model integrating economic facts for stock movement prediction, Eng. Appl. Artif. Intell. 133 (2024) 108320.
[41] J. Yang, W. Zhang, X. Zhang, J. Zhou, P. Zhang, Enhancing stock movement prediction with market index and curriculum learning, Expert Syst. Appl. 213 (2023) 118800.
[42] Y. Song, X. Tang, H. Wang, Z. Ma, Volatility forecasting for stock market incorporating macroeconomic variables based on GARCH-MIDAS and deep learning models, J. Forecast. 42 (1) (2023) 51–59.
[43] L. Zhang, C. Aggarwal, G.-J. Qi, Stock price prediction via discovering multi-frequency trading patterns, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 2141–2149.
[44] Z. Lei, C. Zhang, Y. Xu, X. Li, DR-GAT: Dynamic routing graph attention network for stock recommendation, Inform. Sci. 654 (2024) 119833, http://dx.doi.org/10.1016/j.ins.2023.119833.
[45] H. Liu, T. Zhao, S. Wang, X. Li, A stock rank prediction method combining industry attributes and price data of stocks, Inf. Process. Manage. 60 (4) (2023) 103358.
[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[47] X. Zhou, Z. Pan, G. Hu, S. Tang, C. Zhao, Stock market prediction on high-frequency data using generative adversarial nets, Math. Probl. Eng. 2018 (2018) 1–11.
[48] E.F. Fama, K.R. French, Common risk factors in the returns on stocks and bonds, J. Financ. Econ. 33 (1993) 3–56.
[49] J. Liu, R.F. Stambaugh, Y. Yuan, Size and value in China, J. Financ. Econ. 134 (1) (2019) 48–69.
[50] R. Herzallah, D. Lowe, A mixture density network approach to modeling and exploiting uncertainty in nonlinear control problems, Eng. Appl. Artif. Intell. 17 (2004) 145–158.