Enhancing Portfolio Optimization with Weighted Scoring for Return Prediction Through Machine Learning and Neural Networks

Ruili Sun, Qiongchao Xia, Shiguo Huang
Zhengzhou University of Light Industry
School of Mathematics and Information Science, Zhengzhou, China

***Abstract*—Accurately predicting stock return can enhance the effectiveness of portfolio optimization models. Many previous studies typically divide machine learning algorithms and portfolio optimization into two separate stages: the first step leverages the powerful modeling capabilities of machine learning algorithms to select stocks, and the second step optimizes weights using traditional portfolio models. This separation means that the modeling strengths of machine learning are only utilized in the stock selection phase and not fully exploited during weight optimization. Therefore, this study proposes a portfolio construction method based on Return Prediction Weighted Scoring (RPWS). RPWS generates a stock ranking by assigning weighted scores to each stock, cleverly maps this ranking to weight biases, and then optimizes actual weights using a traditional covariance matrix. This process successfully integrates the modeling capabilities of machine learning into the weight optimization phase, ensuring its full utilization throughout the portfolio construction process. Backtesting experiments are conducted using the U.S. stock market, A-share market, and major cryptocurrencies as datasets, with Support Vector Regression (SVR), Transformer, and other machine learning algorithms as prediction models. Empirical results from these three markets show that the SVR-RPWS and Transformer-RPWS models significantly outperform mainstream funds and traditional portfolio models in terms of annualized returns, sharpe ratio, and drawdown control.**

***Keywords*—Machine learning; stock return prediction; portfolio optimization; support vector regression; transformer; NASDAQ stock market; a-share stock market; cryptocurrency market**

I. Introduction

The optimization of a portfolio selection entails the strategic distribution of assets to maximize returns and minimize risk, two pivotal elements in this process. The endeavor to enhance profits while mitigating risk is a primary goal for investors. The genesis of portfolio selection as a domain of study is attributed to the introduction of the Mean-Variance model (Markowitz [1]), which evaluates return in terms of mean and risk in terms of variance. Subsequently, variance emerges as a commonly used risk measure, and studies related to models such as the Mean-Variance model gain popularity within the academic community, drawing significant attention from scholars such as Björk et al. [2], Liagkouras and Metaxiotis [3], Rosadi et al. [4] and Katsikis et al. [5].

Mean-Variance model relies on the anticipated return and risk associated with different assets to generate optimal portfolios corresponding to various levels of expected return and risk (Beheshti [6]). Consequently, through the careful selection of assets for inclusion in the optimization process, the Mean-Variance model has the potential to enhance overall portfolio performance (Thakur et al. [7]). In recent years, numerous prediction-based portfolio selection models have been introduced in the realm of portfolio management. When confronted with equivalent expected returns, a portfolio selection model boasting a more favorable efficient frontier can effectively reduce risk exposure. Therefore, enhancing the efficacy of prediction-based portfolio selection models remains a crucial endeavor in maximizing portfolio performance.

In financial market, individual investors are generally interested in understanding the fluctuations in the returns of their investment assets today, the potential trends in returns for tomorrow, and the strategies that should be implemented to optimize their portfolio composition (Zhang et al. [8]). Consequently, integrating forecasting theory into portfolio construction holds significant promise for financial investment (Kolm et al. [9]). Forecasting financial time series is consistently seen as one of the most formidable tasks due to the dynamic, nonlinear, unstable, and complex characteristics, coupled with the long-term fluctuations of the financial market (Chen and Hao [10]; Paiva [11]). Nevertheless, reliable investment decisions should be based on long-term observations and behavioral patterns of asset data rather than short-term data (Chong et al. [12]). Accordingly, it is imperative to examine the changes and volatility of financial data over an extended historical period to adequately prepare for future trend forecasts and investment decisions. Numerous broadly accepted empirical studies suggest that financial time series exhibit memory of past periods, indicating that financial markets are, to some extent, predictable. The long-term behavior of an asset significantly influences the risks and returns of a portfolio, thereby further impacting investment decisions (Liu and Loewenstein [13]).

To date, the predominant focus of prior research has been on the utilization of statistical methodologies and machine learning techniques in the predictive analysis domain. Statistical methodologies endeavor to anticipate forthcoming trends by scrutinizing historical pricing attributes such as autoregressive integrated moving average (ARIMA) (Anderson et al. [14]), auto-regressive conditional heteroscedasticity (ARCH) (Engle [15]; Shephard [16]), and generalized auto-regressive conditional heteroscedasticity (GARCH) (Bollerslev [17]; Garcia et al. [18]), whereas prevalent machine learning methodologies encompass support vector machines (Villegas et al. [19]), random forests (Kavzoglu and Teke [20]), and neural networks (Ma [21]). Notwithstanding the capacity of statistical methods

to contribute to stock price forecasting, empirical investigations have revealed that machine learning exhibits superior efficacy in addressing challenges associated with non-stationarity and non-linearity when compared to traditional statistical models (Boulesteix and Schmid [22]; Zhang et al. [23]). Recently, several studies (Freitas et al. [24], Jiang et al. [25], Kaczmarek [26], Padhi et al. [27]) have incorporated the pre-selection of assets into portfolio selection models utilizing machine learning techniques. For example, Deng and Min [28] implemented a linear regression model with ten factors to select stocks from both U.S. and international markets, subsequently constructing a mean-variance-based portfolio that accounted for practical risk tolerance, error monitoring, and turnover constraints. Their results demonstrated that the risk-adjusted return of the global equities model surpassed that of the domestic equity universe, with portfolio returns increasing in conjunction with systemic monitoring errors and risk tolerance. Similarly, Ma [21] employed two machine learning models—random forest (RF) and support vector regression—alongside three deep learning models: convolutional neural network (CNN), long short-term memory (LSTM) neural network, and deep multilayer perceptron (DMLP), to forecast returns for portfolio construction. They subsequently utilized these forecasts to enhance omega portfolio optimization and Mean-Variance model. Their findings revealed that the return predictions for Mean-Variance model and omega models using the RF algorithm outperformed those produced by the other models.

The utilization of machine learning and neural networks in the realm of stock forecasting has garnered considerable success, as evidenced by various studies (Ballings et al. [29]; Sezer and Ozbayoglu [31]; Tsai and Hsiao [32]). These findings advocate for an integration of machine learning methodologies with traditional models of portfolio optimization for the creation of investment portfolios. Numerous academics have been delving into this field of study, primarily branching out into three distinct avenues. Specifically, the first approach involves certain researchers employing Machine Learning models solely during the initial stock selection phase, subsequently utilizing the chosen stocks to construct portfolios through traditional portfolio optimization techniques (Paiva [11]; Vo et al. [33]; Wang et al. [34]). Evidence from these studies suggests that the utilization of stocks selected via machine learning algorithm can enhance the efficacy of conventional portfolio optimization frameworks. In the second vein, several scholars adopt machine learning algorithm for forecasting future stock returns, thereafter incorporating these forecasts to formulate novel objective functions. These newly developed functions aim to refine and augment the objective functions inherent in classical portfolio optimization models (Ma et al. [35]; Ustun and Kasimbeyli [36]; Yu et al. [37]; Yu et al.). Findings from these inquiries indicate that objective functions, enriched by predictive insights from machine learning algorithm, can significantly improve upon traditional portfolio optimization methods. Lastly, a third group of researchers leverages machine learning algorithm to forecast stock returns, opting to utilize the forecast errors in lieu of historical returns when constructing traditional portfolio optimization models (i.e., prediction-based portfolio optimization models). This preference stems from the observation that the normality of forecast errors surpasses that of historical returns (Freitas et al. [39]; Freitas et al. [24]; Hao et al. [40]; Ma et al. [41]). Insights from these studies reveal that forecast errors derived from machine learning algorithm prove more adept than historical returns for the purpose of enhancing portfolio optimization models.

Many previous studies primarily focus on using machine learning algorithms and neural network models to predict stock returns for stock selection, followed by optimizing weights through traditional portfolio models (for example, Minimum Variance Model, Mean VaR Model, Mean Absolute Deviation Model, and others) to enhance investment decisions. This study introduces a novel approach to portfolio construction that combines return predictions from machine learning and neural network models with historical covariance matrix, employing a weighted scoring method to enhance portfolio optimization. This method consists of two main steps, aimed at optimizing portfolio construction and risk management. Specifically, the first step involves using a trained machine learning model or neural network to predict the expected future returns of each asset in the backtest dataset. Once the expected returns are obtained, the assets are ranked according to these return values, and different weights are assigned to each asset through squared weighting. A higher weight corresponds to a higher expected return. The squared weighting method helps to increase the investment proportion in assets with higher expected returns, while relatively reducing the weight of assets with lower expected returns, thereby enhancing the overall potential return of the portfolio. The second step is to calculate the historical covariance matrix. The covariance matrix of the portfolio reflects the correlation and volatility of returns between different stocks. Then the covariance matrix is used to constrain the optimal portfolio weights. This method uses the covariance matrix optimization mechanism to improve expected returns while reasonably controlling portfolio risk, thereby achieving a more stable and optimized asset allocation in a complex market environment. This proposed approach will undergo backtesting in the constituents of the CSI 300 Index, the NASDAQ Index, and major cryptocurrencies, with comprehensive comparisons to traditional portfolio models based on the backtest results.

The rest of this study is organized as follows: Section II provides an overview of the machine learning algorithms, neural network models, and traditional portfolio models used in this study. Section III introduces the novel portfolio construction method proposed in this study, as well as the data selection and experimental design process. Section IV and Section V evaluate the performance of the predictive models and analyze the empirical results. Finally, Section VI concludes the work.

II. Methodology

This section will introduce the methods used in this study, including prediction models and traditional portfolio models.

A. Machine Learning Methods

XGBoost, which stands for "eXtreme Gradient Boosting" was introduced by Chen and Guestrin [42]. This algorithm is characterized by its low computational complexity, rapid execution speed, and high accuracy. The objective function of XGBoost integrates the standard penalty term with the loss function to derive the optimal solution. The inclusion of the regularization term serves to minimize the model's variance, thereby mitigating the risk of overfitting.

The k-nearest neighbors (KNN) algorithm, also known as k-NN, is a non-parametric, supervised learning method applied to both classification and regression tasks. In the context of KNN regression, the algorithm functions by identifying a subset of points in the dataset that are nearest to the "query" point, as determined by a specified distance metric, such as Euclidean, Minkowski, or Manhattan distance [43]. KNN also employs a hierarchical tree-based data structure to manage the dataset efficiently [44].

Support Vector Regression (SVR) represents a supervised learning algorithm specifically designed for regression tasks, as initially proposed by Cortes and Vapnik in 1995 [45]. The underlying principle of SVR is Vapnik's Structural Risk Minimization (SRM), which is adept at tackling various regression problems. The incorporation of kernel functions facilitates the handling of diverse types of input data, thereby enhancing the model's flexibility and applicability. Given its robustness and accuracy, SVR has emerged as a quintessential machine learning technique, extensively utilized in the field of stock market prediction (Emir [46]; Matías and Reboredo [47]; Rasel et al. [48]).

B. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM), a variant of Recurrent Neural Network (RNN), was first proposed by Hochreiter and Schmidhuber [49]. The LSTM model, distinct from traditional RNNs, possesses a memory function, enabling it to retain data over an extended period [50]. This unique feature is facilitated by gate structures, namely an input gate, a forget gate, and an output gate, which filter incoming information to enhance and sustain memory cells. LSTMs are particularly favored in financial time-series prediction due to their ability to effectively manage redundancy in historical data [51].

The operational formulas of LSTM are as follows:

Forget gate:

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f)$$

Input gate:

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o)$$

$$\tilde{c}_t = \tanh(w_c[h_{t-1}, x_t] + b_c)$$

Output gate:

$$c_t = f_t c_{t-1} + i_t \tilde{c}_t$$

$$h_t = o_t \tanh(c_t)$$

In these formulas, f_t, i_t, and o_t signify the forget gate, input gate, and output gate, respectively.

The weight of the matrix is represented by w, while b_t, b_i, and b_o denote the bias of the forget gate, input gate, and output gate, respectively.

The sigmoid function is denoted by σ, and x_t and h_t indicate the input and the current output at time t, respectively. The value from the input gate at time t is represented by c_t, and the hyperbolic function is represented by $\tanh$.

C. Transformer

The Transformer model, introduced by Vaswani et al. [52], represents a significant advancement in natural language processing and sequence modeling. Unlike recurrent architectures such as LSTMs, the Transformer relies entirely on self-attention mechanisms to process input sequences, enabling parallelization and enhancing efficiency. This self-attention mechanism allows the model to assess the importance of different words in a sequence when encoding information, effectively capturing long-range dependencies without the constraints of sequential processing.

The Transformer architecture comprises an encoder and a decoder, each consisting of multiple layers. The encoder processes an input sequence to generate a continuous representation, while the decoder utilizes this representation to produce an output sequence. The core operations within the encoder can be described by the following formulas:

First, input embeddings are augmented with positional encodings to preserve the order of the sequence:

$$z_i = x_i + PE(i)$$

where, z_i denotes the input embedding at position i, x_i is the original embedding, and $PE(i)$ represents the positional encoding for position i.

The self-attention mechanism computes attention scores for each position in the input sequence:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where, Q, K, and V are the query, key, and value matrices derived from the input embeddings, and d_k is the dimensionality of the keys, used to scale the dot-product attention scores.

Subsequently, the output from the attention mechanism is processed through a feed-forward neural network:

$$\text{FFN}(z) = \text{ReLU}(W_1 z + b_1)W_2 + b_2$$

where, W_1 and W_2 are weight matrices, and b_1 and b_2 are bias terms.

Each layer of the encoder includes residual connections and layer normalization to stabilize training:

$$\text{Output} = \text{LayerNorm}(z + \text{FFN}(\text{Attention}(Q, K, V)))$$

The decoder follows a similar structure, incorporating masked self-attention to prevent attention to future tokens, ensuring autoregressive output generation.

The Transformer has demonstrated exceptional performance across various tasks, including machine translation and text generation, owing to its efficiency in capturing complex dependencies in data. Its flexibility and scalability have established it as a cornerstone of modern deep learning architectures, significantly influencing subsequent models in the field [53].

D. Informer

Informer is a novel model designed for long sequence time-series forecasting, as proposed by Zhou et al. [54]. Unlike traditional time-series models, Informer employs a self-attention mechanism specifically optimized for efficient handling of long sequences. Its key innovation is an adaptive attention mechanism that focuses on relevant input features while discarding less relevant information, thus alleviating the computational burden associated with standard attention mechanisms.

The Informer architecture consists of an encoder and a decoder, similar to the Transformer model, but with enhancements for long-sequence performance. The core components of Informer are as follows:

The adaptive attention mechanism generates a sparse attention matrix using a probability distribution D:

$$D_{ij} = \frac{\exp(\text{score}(Q_i, K_j))}{\sum_{k \in \mathcal{N}} \exp(\text{score}(Q_i, K_k))}$$

where, $\text{score}(Q_i, K_j)$ represents the score function computed between query Q_i and key K_j, and $\mathcal{N}$ denotes the neighborhood of keys attended to by the query. This ensures that each query focuses on a subset of keys, significantly reducing computational complexity.

The encoder processes the input sequence to generate context-aware representations through multiple layers of self-attention and feed-forward networks:

$$Z^{(l)} = \text{LayerNorm}\left(Z^{(l-1)} + \text{Attention}(Z^{(l-1)}, Z^{(l-1)}, Z^{(l-1)})\right)$$
$$H^{(l)} = \text{LayerNorm}\left(Z^{(l)} + \text{FFN}(Z^{(l)})\right)$$

where, $Z^{(l)}$ is the output of layer l, and $H^{(l)}$ represents the result after applying the feed-forward network FFN.

In the decoder, the predicted output is generated based on the encoder's output and previous outputs. The decoder employs masked self-attention to ensure autoregressive prediction:

$$\hat{Y}_t = \text{LayerNorm}\left(Y_{t-1} + \text{MaskedAttention}(Y_{t-1}, Z)\right)$$

where, $\hat{Y}_t$ is the predicted output at time t, and Y_{t-1} denotes the previous outputs.

Informer performs well in forecasting tasks by effectively managing long-term dependencies and reducing computational overhead. Compared to previous methods, it shows significant improvements in accuracy and efficiency, Lu et al. [55].

E. Mean-Variance Model

The selection of investment strategies has always been a vital area in financial research. Amongst a wide array of investment strategies, the Mean-Variance Model, put forth by Harry Markowitz in 1952, has gained substantial recognition [1].

The model's theoretical rigor and practicality have led to its broad utilization in the practical realm of investment decisions. Its main idea is to find the investment proportion that minimizes the total risk of the portfolio at a given expected return level; or at a given risk level, find the investment proportion that maximizes the total return of the portfolio. Let $R = (R_1, R_2, \ldots, R_n)^T$, where $R_i = E(r_i)$ represents the expected return of the i-th asset; $X = (x_1, x_2, \ldots, x_n)^T$ is the weight vector of the portfolio; $\Sigma = (\sigma_{ij})_{n \times n}$ is the covariance matrix between n assets; $E(r_p)$ and σ_p^2 are the expected return and variance of the portfolio, respectively.

The mathematical formula for the Mean-Variance Model is as follows:

$$\begin{aligned} \min \quad & \sigma_p^2 = X^T \Sigma X \\ \max \quad & E(r_p) = X^T R \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1 \end{aligned}$$

Minimizing the variance (risk) of the portfolio and maximizing the expected return of the portfolio are two objectives that usually cannot be achieved simultaneously, so a trade-off between risk and return is needed. To compare with the portfolio methods proposed in this study, we will solve two special solutions of the mean-variance model through numerical methods.

The first extended model is the **Minimum Variance Portfolio Model (MVM)**, which aims to find the portfolio weight that minimizes the risk of the portfolio among all possible portfolio weight. The formula for this model is as follows:

$$\begin{aligned} \min \quad & \sigma_p^2 = X^T \Sigma X \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1 \end{aligned}$$

The second extended model is the **Maximum Sharpe Ratio Portfolio Model (MSRM)**, which aims to find the weight distribution that maximizes the Sharpe ratio (the ratio of expected return to risk) of the portfolio among all possible weight distributions. The formula for this model is as follows:

$$\max \quad \frac{X^T R - r_f}{\sqrt{X^T \Sigma X}}$$
$$\text{s.t.} \quad \sum_{i=1}^{n} x_i = 1$$

where, r_f stands for the risk-free rate, which is set to 0 in this study.

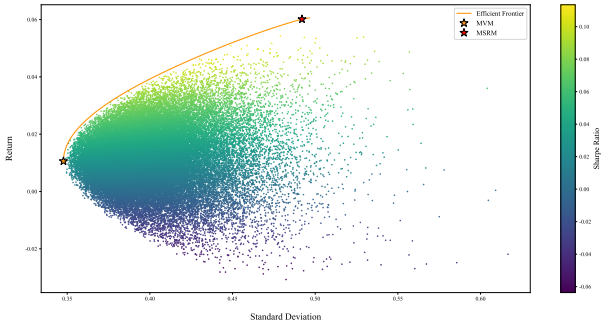

Fig. 1. Efficient frontier with random portfolios.

In Fig. 1, we can see a scatter plot composed of many random investment portfolios. The picture shows a curve called the efficient frontier. Efficient frontier points represent portfolios offering the optimal risk-return trade-off, either maximizing return for a given risk or minimizing risk for a given return. Among these points, two points are specially marked, one is the point with the minimum variance, and the other is the point with the maximum Sharpe ratio. These two points correspond to the solutions of the two models we mentioned above.

F. Risk Parity Model (RPM)

The risk parity strategy, proposed by Dr. Edward Qian, the Chief Investment Officer of Pan Agora, is renowned for its innovative approach to investment. This model was later applied in practice by Bridgewater Associates and achieved significant success for a time. From the perspective of risk budgeting, risk parity involves evenly distributing the overall risk of the portfolio among different assets, meaning that each asset contributes equally to the overall risk of the portfolio. In other words, in the risk parity model, assets with higher volatility are allocated lower weights, while those with lower volatility receive higher weights.

The marginal risk contribution (MRC) of an asset to a portfolio is defined as the partial derivative of the portfolio volatility with respect to the weight of the asset:

$$MRC_i = \frac{\partial \sigma_p}{\partial x_i} = \frac{\partial \sqrt{X^T \Sigma X}}{\partial x_i}$$

The contribution of the asset to the total risk (RC) is the product of the asset's weight and its marginal risk contribution:

$$RC_i = x_i \cdot MRC_i$$

For a risk parity portfolio, the contributions to total risk from each asset are equal, i.e.:

$$RC_i = RC_j$$

Calculating the weights for a risk parity portfolio essentially involves solving a quadratic optimization problem, which can be mathematically described as:

$$\min \quad \sum_{i=1}^{n} \sum_{j=1}^{n} (RC_i - RC_j)^2$$
$$\text{s.t.} \quad \sum_{i=1}^{n} x_i = 1$$
$$0 \leq x_i \leq 1$$

III. Newly Proposed Strategy for Portfolio Selection

A. Proposed Method: Return Prediction Weighted Scoring Strategy (RPWS)

In the field of portfolio optimization, Markowitz's (1952) mean-variance model stands as the most classical theoretical framework, constructing optimal portfolios by balancing expected returns and risk (volatility) [1]. However, the model's reliance on historical data assumptions may fail in rapidly changing markets. With advancements in machine learning and neural network technologies, numerous studies have validated the effectiveness of these techniques in stock prediction. For instance, Chaweewanchon et al. (2022) proposed a portfolio strategy combining the CNN-BiLSTM prediction model with the Markowitz Mean-Variance method. Backtesting experiments show that it outperforms other benchmark models in terms of Sharpe ratio, average return, and risk control [30].

Behera et al. (2023) proposed a portfolio construction method based on machine learning algorithms and the Mean-Value at Risk (Mean-VaR) model. First, stock returns are predicted using algorithms including XGBoost and AdaBoost to select stocks with higher expected returns; then, the Mean-VaR model is used for portfolio optimization. Based on stock data from the Bombay Stock Exchange in India, the Tokyo Stock Exchange in Japan, and the Shanghai Stock Exchange in China, experimental results show that the Mean-VaR model combined with AdaBoost prediction outperforms other combinations in performance [38].

Previous studies typically divide machine learning algorithms and portfolio optimization into two separate stages: the first step leverages the powerful modeling capabilities of machine learning algorithms to select stocks, and the second step optimizes weights using traditional portfolio models. This separation means that the modeling strengths of machine learning are only utilized in the stock selection phase and not fully exploited during weight optimization. Therefore, this study proposes a portfolio construction method based on Return Prediction Weighted Scoring (RPWS). RPWS generates a stock ranking by assigning weighted scores to each stock, maps this ranking to weight biases, and then optimizes actual

weights using a traditional covariance matrix. This process successfully integrates the modeling capabilities of machine learning into the weight optimization phase, ensuring its full utilization throughout the portfolio construction process.

The RPWS algorithm's mathematical model achieves dynamic weight optimization through two core components: 1) constructing a reference weight vector based on return prediction and rank weighting, and 2) covariance-constrained weight allocation optimization. The specific steps are as follows: Let $P = (p_1, p_2, p_3, \ldots, p_n)^T$, where p_i represents the ranking of the i-th stock among all stocks based on the expected return predicted by machine learning and neural network models. Let $\Sigma = (\sigma_{ij})_{n\times n}$ be the covariance matrix between n assets. The reference weight vector X_0 is constructed as:

$$X_0 = \frac{p_j^{-2}}{\sum_{j=1}^{n} p_j^{-2}}$$

The objective of this reference weight vector is to allocate higher weights to higher-ranked stocks. The optimal weights are then solved through convex optimization with risk budget constraints:

$$\begin{aligned} \min \quad & X^T \Sigma X + \lambda \|X - X_0\|_2 \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1 \end{aligned}$$

where,

- λ: Regularization parameter controlling the emphasis on the reference matrix (larger λ forces weights closer to X_0, degenerating to minimum variance model when $\lambda = 0$)
- $\|X - X_0\|_2$: L2 norm measuring the deviation between optimal weights and reference weights.

This formulation integrates machine learning predictions through X_0 while maintaining risk constraints through covariance matrix optimization. Considering that this is a convex optimization problem, in order to simplify the hyperparameter tuning, this study sets $\lambda = 1$ to solve the problem in the experiment.

B. Experimental Process

In financial markets, investors consistently aim to optimize their portfolios using scientific methods and strategies to achieve the optimal balance between risk and return.

However, the uncertainty and complexity of the market make constructing an efficient portfolio a challenging task. This study focuses on prediction as its core research direction, leveraging machine learning and neural network techniques to model and assign higher weights to stocks with higher potential returns and manageable risks, thereby constructing investment portfolios. By adopting this method, not only can the overall return expectations of the portfolio be improved, but the reliability and guidance of the decision-making process for investors can also be significantly enhanced. This methodology involves two main phases:

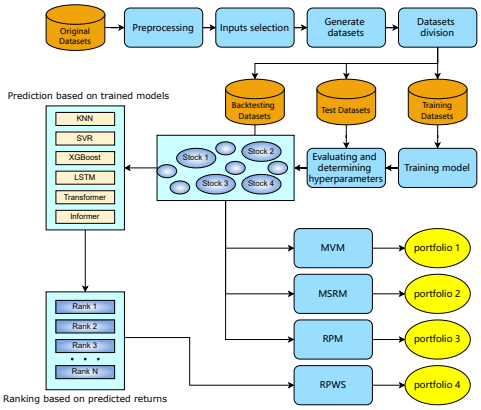

Fig. 2. Experimental process of the study.

1) Model training and performance evaluation: In this experimental approach, various models, including XGBoost, KNN, SVR, LSTM, Transformer, and Informer, are employed for stock return prediction. To evaluate the accuracy and performance of these models, three metrics are used: Mean Absolute Percentage Error (MAPE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). The training in this stage is conducted on a dataset separate from the backtesting dataset. The primary goal at this stage is to determine appropriate model parameters and assess the models' performance. The hyperparameters for each model are detailed in Table I.

2) Portfolio backtesting: In this phase, portfolio backtesting is conducted to compare the performance of different portfolio models. The models include the Minimum Variance Portfolio Model (MVP), Maximum Sharpe Ratio Portfolio Model (MSR), Risk Parity Model (RP), and the Return Prediction Weighted Scoring Model (RPWS). For the A-share and U.S. stock markets, portfolio allocations are adjusted every 20 trading days. In contrast, for the cryptocurrency market—which operates year-round without interruptions—allocations are adjusted every 7 days. When using the Return Prediction Weighted Scoring Model, a rolling training approach is applied, utilizing the 500 most recent historical samples prior to each current date. This ensures the model adapts to the trends and dynamics of changing market conditions.

The overall experimental workflow is illustrated in Fig. 2.

IV. Data and Performance Evaluation

A. Datasets and Pre-processing

The selection of the dataset is crucial for model evaluation, and this experiment will use index component stocks as the data source. Indices are reliable financial indicators of high-quality stocks and reflect mainstream investment trends in the market. Index component stocks are of high quality, with minimal occurrences of suspension or delisting, ensuring the

TABLE I. PARAMETERS FOR PREDICTION MODELS

Model	KNN	XGBoost	SVR	LSTM	Transformer	Informer
Parameter	n_neighbors	n_estimators	C	epochs	epochs	epochs
(Value)	(3)	(150)	(10)	(50)	(50)	(50)
		max_depth	gamma	batch_size	batch_size	batch_size
		(10)	(0.1)	(64)	(64)	(64)
		learning_rate		optimizer	optimizer	optimizer
		(0.01)		(adam)	(adam)	(adam)
		gamma		dropout_rate	dropout_rate	dropout_rate
		(2)		(0.1)	(0.1)	(0.1)
				units	embed_dim	embed_dim
				(64)	(8)	(8)
					num_heads	num_heads
					(2)	(2)
					ff_dim	ff_dim
					(64)	(64)

TABLE II. RANDOMLY SELECTED TICKERS AND MAJOR CRYPTOCURRENCIES

U.S. Stock Market		A-share Market		Crypto Market
AAPL	MCHP	600028	601398	ADA
ADBE	MDLZ	600031	601601	AVAX
ADP	MNST	600036	601628	BNB
ALGN	MSFT	600050	601633	BTC
AMD	NFLX	600089	601668	DOGE
AMZN	NTES	600111	601669	ETH
CHTR	NXPI	600150	601857	LINK
CMCSA	PAYX	600276	601899	SHIB
CSCO	PEP	600309	601919	SOL
CSX	QCOM	600406	601988	XRP
CTSH	ROST	600436		
DLTR	SBUX	600438		
EA	SWKS	600519		
EBAY	TTWO	600809		
GOOGL	TXN	600893		
IDXX	VRSK	601088		
INTU	WDAY	601166		
ISRG	WDC	601288		
LULU	XEL	601318		
MAR	ZBRA	601390		

continuity and completeness of the experimental data samples. This quality is advantageous for model construction, as portfolios based on index component stocks tend to be more robust and offer better potential returns. Due to the immaturity and high volatility of the cryptocurrency market, only a few selected major currencies are included in the dataset.

To validate the robustness of the model, this study will conduct backtesting across multiple markets, including the A-share market, the U.S. stock market, and the cryptocurrency market. The updated dataset includes the following components:

- CSI 300 Index Component Stocks: Daily data from January 2017 to January 2024, with a random selection of 30 stocks.
- NASDAQ Index Component Stocks: Daily data from January 2014 to January 2024, with a random selection of 40 stocks.
- Cryptocurrency Data: Daily data from January 2023 to October 2024, covering 10 major cryptocurrencies.

The dataset includes various factors such as opening price, closing price, highest price, lowest price, trading volume, and three technical indicators: the Relative Strength Index (RSI) with a 12-period, the Moving Average Convergence Divergence (MACD) with a 12-period, and the On-Balance Volume (OBV). This comprehensive datasets aims to enhance the reliability and stability of the experimental results, providing a solid foundation for validating the model's performance across different financial markets. The specific stocks selected are detailed in Table II.

Data normalization involves transforming data with different scales and units into a uniform format, allowing for comparability across different dimensions and facilitating feature extraction during the model training process. In this experiment, we use the Max-Min normalization method to process each feature factor in a dimensionless manner. The specific formula for normalization is shown below:

$$y_i = \frac{x_i - \min\limits_{1 \le j \le n} \{x_j\}}{\max\limits_{1 \le j \le n} \{x_j\} - \min\limits_{1 \le j \le n} \{x_j\}}$$

where, x_i represents the original data, y_i denotes the normalized dimensionless data, and j is the j-th element in the sequence.

B. Performance Evaluation

This study employs six metrics, namely MSE, RMSE and MAE, to provide a comprehensive analysis of various models' performance in predicting stock returns. These metrics are

TABLE III. PERFORMANCE OF DIFFERENT MODELS ACROSS THREE MARKETS

		U.S. Stock Market			A-share Market			Crypto Market		
Model	Metric	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE
KNN	Mean	0.0338	0.1725	0.1421	0.0360	0.1646	0.1377	0.0916	0.2769	0.2461
	Std	0.0302	0.0651	0.0595	0.0416	0.0962	0.0881	0.0714	0.1291	0.1309
	Var	0.0009	0.0042	0.0035	0.0017	0.0093	0.0078	0.0051	0.0167	0.0171
SVR	Mean	0.0702	0.2355	0.2014	0.0613	0.2201	0.1927	0.1041	0.2705	0.2478
	Std	0.0687	0.1239	0.1268	0.0572	0.1158	0.1063	0.1202	0.1855	0.1900
	Var	0.0047	0.0154	0.0161	0.0033	0.0134	0.0113	0.0144	0.0344	0.0361
XGBoost	Mean	0.0377	0.1767	0.1481	0.0416	0.1712	0.1461	0.0989	0.2707	0.2443
	Std	0.0374	0.0818	0.0788	0.0530	0.1131	0.1052	0.1058	0.1687	0.1738
	Var	0.0014	0.0067	0.0062	0.0028	0.0128	0.0111	0.0112	0.0285	0.0302
LSTM	Mean	0.0112	0.1004	0.0806	0.0167	0.1131	0.0907	0.0273	0.1485	0.1210
	Std	0.0086	0.0337	0.0292	0.0195	0.0641	0.0545	0.0243	0.0767	0.0672
	Var	0.0001	0.0011	0.0009	0.0004	0.0041	0.0030	0.0006	0.0059	0.0045
Transformer	Mean	0.0412	0.1879	0.1588	0.0864	0.2402	0.2126	0.1339	0.3295	0.3105
	Std	0.0383	0.0782	0.0699	0.1096	0.1730	0.1656	0.1037	0.1677	0.1686
	Var	0.0015	0.0061	0.0049	0.0120	0.0299	0.0274	0.0107	0.0281	0.0284
Informer	Mean	0.0389	0.1818	0.1529	0.1509	0.3003	0.2715	0.0951	0.2786	0.2434
	Std	0.0359	0.0779	0.0713	0.2395	0.2515	0.2524	0.0773	0.1396	0.1344
	Var	0.0013	0.0061	0.0051	0.0573	0.0633	0.0637	0.0060	0.0195	0.0181

commonly utilized as evaluative indicators due to their comprehensive portrayal of predictive capabilities (Freitas [24]; Gandhmal and Kumar [56]; Ma et al. [41]).

1) Mean Squared Error (MSE): MSE calculates the average squared differences between predicted and actual values, emphasizing larger errors due to the squaring operation. The formula for MSE is:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

2) Root Mean Squared Error (RMSE): RMSE represents the square root of the average squared differences between predicted and actual values, providing a measure of the standard deviation of residuals. The formula for RMSE is:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

3) Mean Absolute Error (MAE): MAE measures the average magnitude of errors between predicted and actual values, providing a straightforward assessment of model performance. The formula for MAE is:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

In this study, we evaluate the performance of six models—KNN, SVR, XGBoost, LSTM, Transformer, and Informer—in predicting stock returns. Table III presents the results for each model based on the performance metrics MSE, RMSE, and MAE.

In the U.S. stock market, the LSTM model clearly outperforms the others, exhibiting the lowest error and highest stability. XGBoost and KNN follow closely behind, while Transformer and Informer show similar performances, though they lag behind XGBoost and KNN. The SVR model performs the worst overall. In the Chinese A-share market, the performance patterns of the models are similar to those observed in the U.S. market. The LSTM model again stands out, with the smallest MSE, RMSE, and MAE, and good stability. SVR exhibits relatively large errors and instability, while Transformer and Informer perform the worst, with significantly higher errors. In the cryptocurrency market, the errors of all models increase due to the immature nature and high volatility of the market, making prediction a particularly challenging task. However, the overall performance trends of the models remain similar to those in the U.S. and A-share markets, with LSTM still showing the best performance and SVR and Transformer showing the worst.

It is important to note that the performance metrics presented here are based solely on the results from the test dataset. The data used for the rolling backtest, which is entirely separate from this test dataset, may not reflect the actual predictive performance of each model throughout the backtesting process.

V. EMPIRICAL RESULTS

After comparing the performance of different models in stock return prediction, a backtesting simulation is conducted to study the investment performance of various portfolios, with an initial capital of 100,000. In the A-share and U.S. stock markets, portfolio weights are adjusted every 20 days, while in the cryptocurrency market, portfolio weights are adjusted every 7 days. For simplicity, dividends and transaction fees are disregarded, and leverage and short-selling are not considered during the portfolio rebalancing process. In this

TABLE IV. DIFFERENT PORTFOLIO OPTIMIZATION MODELS FOR COMPARISON

Abbreviation	Description
QQQ (ETF)	Invesco QQQ Trust (Nasdaq: QQQ).
CSI 300 (ETF)	CSI 300 Index Securities Investment Fund.
BTC	Bitcoin: A mainstream cryptocurrency.
1/N	Equal Weight Strategy.
RPM	Risk Parity Model.
MVM	Minimum Variance Portfolio Model.
MSRM	Maximum Sharpe Ratio Portfolio Model.
SVR-RPWS	SVR for prediction and ranking, RPWS for portfolio construction.
KNN-RPWS	KNN for prediction and ranking, RPWS for portfolio construction.
XGBoost-RPWS	XGBoost for prediction and ranking, RPWS for portfolio construction.
LSTM-RPWS	LSTM for prediction and ranking, RPWS for portfolio construction.
Informer-RPWS	Informer for prediction and ranking, RPWS for portfolio construction.
Transformer-RPWS	Transformer for prediction and ranking, RPWS for portfolio construction.
SVR-MVM(N=10)	SVR for prediction and screening, MVM for portfolio construction.

Notes:
1. To more clearly highlight the performance of RPWS, the SVR-MVM model is introduced as a comparative experiment. This model uses the best-performing SVR algorithm to select the top 10 stocks based on predicted returns. We select 10 stocks for the following reasons: Evans and Archer (1968) have demonstrated that the traditional rule of thumb suggesting 8 to 10 stocks, established by pioneering studies, is indeed sufficient to achieve optimal diversification effects [58]. Alexeev and Dungey (2015) assert that an equally weighted portfolio of 10 stocks would suffice for an average investor aiming to diversify away 90% of the risk [59]. Recognizing that individual investors may struggle to track too many assets, we have decided that the number of stocks we select will be 10. Optimizes portfolio weights using the Minimum Variance Model (MVM).

2. Cryptocurrencies market, due to limited samples, are no longer included in the comparison.

TABLE V. METRICS FOR MEASURING PORTFOLIO PERFORMANCE

Metrics	Formula	
Annualized Return	$R = \left(\prod_{i=1}^{N}(1+r_i)\right)^{\frac{252}{N}} - 1$	
Annualized Volatility	$V = \sqrt{\frac{252}{N}\sum_{i=1}^{N}(r_i-\mu)^2}$	$\mu = \frac{1}{N}\sum_{i=1}^{N} r_i$
Sharpe Ratio	Sharpe Ratio $= \frac{R - r_f}{V}$	
Sortino Ratio	Sortino Ratio $= \frac{R - r_f}{D}$	$D = \sqrt{\frac{252}{N}\sum_{i=1}^{N}\min(r_i - r_f, 0)^2}$
Maximum Drawdown	$\text{MDD} = \max_{1 \le i \le N}\left(\frac{P_{\text{peak},i} - P_i}{P_{\text{peak},i}}\right)$	
VaR(95%)	VaR(95%) $= \mu - 1.645 \times \sigma$	$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(r_i-\mu)^2}$

Notes:

1. r_i denotes the return in the i-th day, N is the total number of days, and 252 is the number of trading days in a year.

2. r_f is the risk-free rate, which is set to 0 in this study.

3. P_i represents the return on the i-th day, while $P_{\text{peak},i}$ denotes the historical maximum return up to the i-th day.

4. 1.645 is the quantile of the standard normal distribution at the 95% confidence level.

study, six key metrics are used to evaluate the performance of different portfolios: annualized return, Sharpe ratio, maximum drawdown, Sortino ratio, annualized volatility, and Value at Risk (VaR) at the 95% confidence level. Table IV shows different portfolio optimization models for comparison. The details of each indicator can be found in Table V.

From 2018 to 2024, the U.S. stock market experienced dramatic fluctuations, including a bull market recovery, a bear market adjustment, and dramatic fluctuations caused by the epidemic. The market was affected by multiple factors such as interest rate hikes, inflation, and stimulus policies, and generally showed a fluctuating upward trend. In the backtest, the SVR-RPWS model performs best overall, followed by the Transformer-RPWS model. The SVR-RPWS model achieves the highest annualized return (28.11%), the highest Sharpe ratio (1.1436) and the Sortino ratio (1.6704), and is also excellent in drawdown control, with the maximum drawdown (-22.74%) second only to the LSTM-RPWS model (-21.86%). The SVR-RPWS and Transformer-RPWS models outperform QQQ (QQQ is an exchange-traded fund that tracks the Nasdaq 100 Index, which includes the 100 largest non-financial companies listed on the Nasdaq stock market) in all indicators and perform significantly better than other models overall. In addition, all models using the RPWS algorithm also outperform QQQ overall, especially in terms of seeking returns. The traditional portfolio model performs well in risk control, with annualized volatility and value at risk (VaR) slightly better than the model using the RPWS algorithm at the 95% confidence level, but performs poorly in terms of annualized returns and Sharpe ratio, and failed to surpass QQQ. The SVR-MVM model performs quite well overall, with a Sharpe ratio second only to SVR-RPWS and Transformer-RPWS, and it also demonstrates good risk control. Compared to the MVM model without any stock screening, it shows significant improvement, reflecting the effectiveness of stock screening. However, in terms of pursuing returns, it still lags considerably behind models using the RPWS algorithm. The detailed backtest performance and cumulative returns are available in Table VI and Fig. 3.

From 2021 to 2024, affected by factors such as the slowdown in economic growth, the Sino-US trade war and the external environment, the A-share market was dominated by a bear market, continued to adjust, and continued to be sluggish. Although there was a short-term rebound, the overall trend was weak. The three-year annualized return of the CSI 300 ETF (the CSI 300 ETF tracks the CSI 300 Index, which represents the 300 largest and most liquid stocks listed on the Shanghai and Shenzhen Stock Exchanges) was -13.25%. In the backtest, the SVR-RPWS and Transformer-RPWS models perform similarly to the US stock market, both perform well and significantly outperform all other models. Among them, the SVR-RPWS model has the highest annualized return (25.30%), the highest Sharpe ratio (1.2401), and the highest Sortino ratio (2.0532). At the same time, while maintaining high returns, the Transformer-RPWS model performs best in risk control, with a maximum drawdown of only -12.58%, an annualized volatility of 15.29% and a VaR (95%) of -1.29%. It is worth noting that although the traditional models perform well in terms of risk control, the overall returns are disappointing. The MVM model is the only traditional model that achieves positive returns in the bear market, but the return is only 1.73%. In contrast, all models using the RPWS algorithm achieve positive returns and are better than the MVM model. The SVR-MVM model performs similarly in the A-share market as in the U.S. stock market. While maintaining good risk control, the annualized return of the MVM model, after SVR stock selection, increases from 1.73% to 3.75%, representing a 105.84% improvement in the original annualized return metric. Although this is already a significant improvement, it still significantly underperforms portfolios using the RPWS model in terms of annualized return, Sharpe ratio, and Sortino ratio. The detailed backtest performance and cumulative returns are available in Table VII and Fig. 4.

From January 2023 to October 2024, the cryptocurrency market experienced significant growth, with major cryptocurrencies such as Bitcoin and Ethereum seeing sharp price increases, with Bitcoin breaking through its all-time high. The gradual acceptance of cryptocurrencies by institutional investors and the traditional financial industry has played a key role in driving mainstream market applications. Investor sentiment remains optimistic, and a large amount of capital has flowed into the market, jointly driving a strong bull market trend. Bitcoin dominates with a market share of 67.3%, and in this bull market, it has achieved an annualized return of 42.87% in just over a year. However, compared with traditional stock markets, the cryptocurrency market is relatively immature, extremely volatile, and greatly affected by market sentiment, making modeling and prediction particularly difficult. As a result, most models based on the RPWS algorithm perform poorly. Surprisingly, the Transformer-RPWS model, which performs well in both the US and A-share markets, performs poorly in the cryptocurrency market, with an annualized return of only 28.85%. In contrast, the SVR-RPWS model continues to show excellent performance, achieving an ultra-high annualized return of 75.01%, and its Sharpe ratio (1.2522) and Sortino ratio (2.1192) are the highest among all models. Whether in terms of returns or risk control, the performance of traditional models is only above average at best, and fails to outperform Bitcoin. The detailed backtest performance and cumulative returns are available in Table VIII and Fig. 5.

While the LSTM model demonstrates excellent predictive performance on the test set, its performance in backtesting is relatively mediocre. In contrast, models such as SVR and Transformer, which perform poorly on the test set, show exceptional results when combined with the RPWS algorithm in the backtest. I hypothesize two potential reasons for this discrepancy. First, the time-series nature of financial market data makes overfitting particularly pronounced. Models may learn incidental patterns or short-term fluctuations in historical data that do not persist in the future, thereby limiting their generalization ability. As a result, models that perform well on the test set may fail to produce satisfactory results in subsequent backtests. Second, models that underperformed on the test set may exhibit stronger robustness—such as the SVR model, which demonstrated good stability across multiple markets. These models may be better equipped to adapt to varying market conditions or data volatility, leading to superior performance in the backtest phase.

Overall, in mature markets such as the U.S. and A-share stock markets, models incorporating the RPWS algorithm have demonstrated strong investment performance, often outper-

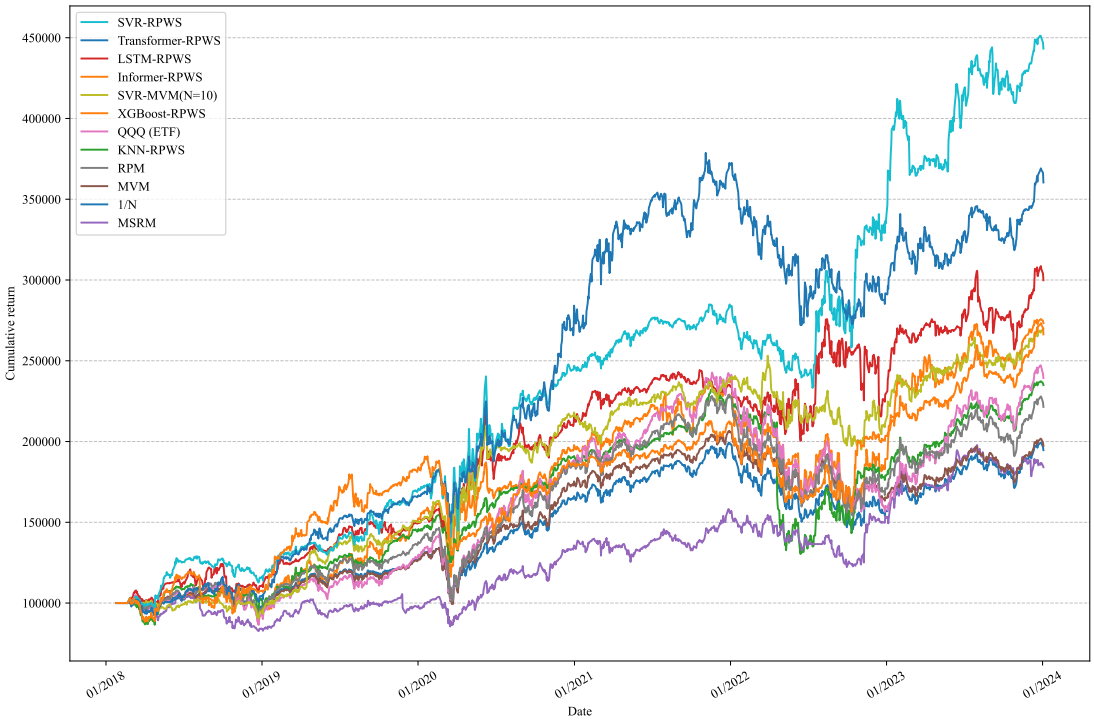

Fig. 3. Backtest cumulative return for the U.S. stock market.

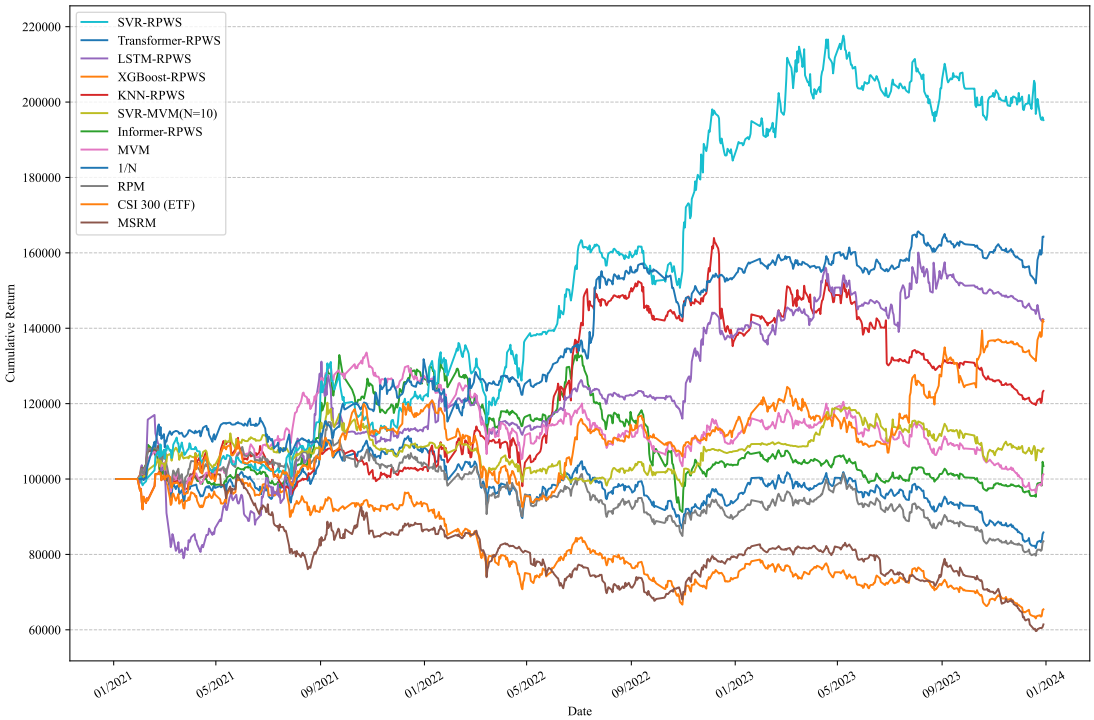

Fig. 4. Backtest cumulative return for A-share market.

TABLE VI. BACKTEST PERFORMANCE OF THE U.S. STOCK MARKET

U.S. Stock Market	Annualized Return	Sharpe Ratio	Maximum Drawdown	Sortino Ratio	Annualized Volatility	VaR (95%)
SVR-RPWS	**28.11%**	**1.1436**	-22.74%	**1.6704**	24.58%	-1.89%
Transformer-RPWS	23.97%	1.1069	-28.18%	1.5858	21.66%	-2.00%
SVR-MVM(N=10)	18.76%	0.9319	-22.95%	1.3659	20.13%	-1.72%
LSTM-RPWS	21.60%	0.8684	**-21.86%**	1.1458	24.87%	-2.13%
Informer-RPWS	20.95%	0.8629	-33.60%	1.1562	24.28%	-2.26%
XGBoost-RPWS	18.99%	0.8572	-31.99%	1.0688	22.16%	-1.97%
KNN-RPWS	16.88%	0.7691	-43.88%	0.8381	21.95%	**-1.62%**
QQQ (ETF)	17.82%	0.7176	-35.57%	0.9339	24.83%	-2.56%
1/N	12.86%	0.7143	-26.94%	0.9386	**18.01%**	-1.67%
RPM	16.04%	0.7013	-32.69%	0.9049	22.87%	-2.19%
MVM	13.33%	0.6916	-25.95%	0.9071	19.27%	-1.75%
MSRM	5.71%	0.3059	-27.09%	0.3795	18.65%	-1.89%

TABLE VII. BACKTEST PERFORMANCE OF THE A-SHARE MARKET

A-share Market	Annualized Return	Sharpe Ratio	Maximum Drawdown	Sortino Ratio	Annualized Volatility	VaR (95%)
SVR-RPWS	**25.30%**	**1.2401**	-15.66%	**2.0532**	20.40%	-1.87%
Transformer-RPWS	18.40%	1.2038	**-12.58%**	1.8676	15.29%	-1.29%
XGBoost-RPWS	14.05%	0.7150	-23.44%	1.1837	19.65%	-1.73%
LSTM-RPWS	15.19%	0.6591	-31.86%	0.9548	22.59%	-1.91%
KNN-RPWS	9.45%	0.4556	-27.06%	0.5923	20.74%	-1.87%
SVR-MVM(N=10)	3.56%	0.2727	-18.49%	0.3941	**13.05%**	**-1.26%**
Informer-RPWS	2.76%	0.1551	-31.82%	0.2244	17.78%	-1.57%
MVM	1.73%	0.1076	-27.92%	0.1625	16.09%	-1.63%
1/N	-4.01%	-0.2500	-26.70%	-0.3778	16.05%	-1.65%
RPM	-4.96%	-0.3077	-30.80%	-0.4523	16.13%	-1.65%
CSI 300 ETF	-13.25%	-0.7743	-41.97%	-1.1289	17.11%	-1.70%
MSRM	-15.26%	-0.8414	-45.16%	-1.1381	18.13%	-1.96%

TABLE VIII. BACKTEST PERFORMANCE OF THE CRYPTOCURRENCY MARKET

Cryptocurrency Market	Annualized Return	Sharpe Ratio	Maximum Drawdown	Sortino Ratio	Annualized Volatility	VaR (95%)
SVR-RPWS	**75.01%**	**1.2522**	-39.61%	**2.1192**	59.90%	-4.41%
BTC	42.87%	1.2131	-26.24%	1.7970	35.34%	-3.34%
MSRM	50.13%	1.1605	-45.00%	1.4531	43.20%	-4.06%
MVM	44.98%	1.1389	-40.72%	1.6393	39.49%	-3.54%
1/N	52.46%	1.1196	-45.36%	1.5988	46.85%	-4.39%
KNN-RPWS	53.59%	1.1018	-45.33%	1.6127	48.64%	-4.37%
Informer-RPWS	34.04%	1.0674	**-24.01%**	1.6081	**31.89%**	**-2.72%**
RPM	41.62%	0.9026	-43.60%	1.2611	46.11%	-4.32%
Transformer-RPWS	28.85%	0.7420	-43.38%	0.9349	38.88%	-3.50%
LSTM-RPWS	33.59%	0.6434	-34.55%	1.0374	52.21%	-4.54%
XGBoost-RPWS	21.48%	0.4881	-47.38%	0.6637	44.00%	-4.02%

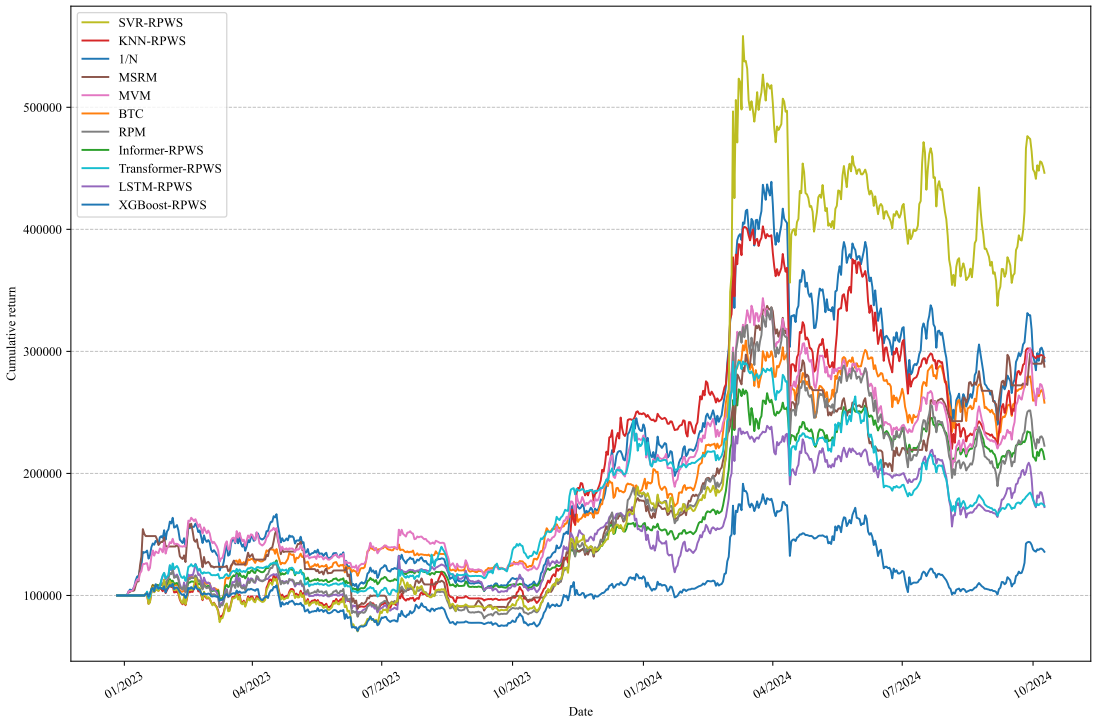

Fig. 5. Backtest cumulative return for cryptocurrency market.

forming traditional portfolio models and major ETFs. This is particularly evident in terms of generating higher returns, thereby validating the fundamental principle of high risk leading to high reward. The experiment highlights the substantial advantages of employing machine learning and neural network models in constructing better-performing investment portfolios. It further reinforces the effectiveness of machine learning and neural network models in portfolio optimization.

VI. Conclusion and Future Work

Machine learning algorithms and neural network models have become essential tools in various fields, including financial market investment, to make more advantageous and informed decisions. This study applies machine learning algorithms and neural network models to portfolio construction, introducing a prediction-based weighted scoring method to build optimized investment portfolios. The research enriches portfolio theory by offering novel insights. The key findings are as follows:

Firstly, this study provides a detailed introduction to six models: K-Nearest Neighbors (KNN), Support Vector Regression (SVR), XGBoost, Long Short-Term Memory (LSTM), Transformer, and Informer. These models are evaluated based on their stock return prediction capabilities on a test dataset. Secondly, this paper presents a portfolio construction method based on return prediction and weighted scoring (RPWS). The effectiveness of this method is validated through the combination with the aforementioned predictive models. Among the models tested, SVR-RPWS and Transformer-RPWS perform well in terms of both maximizing returns and controlling risk and drawdowns. Overall, their performance outperforms mainstream funds and traditional portfolio models. Finally, in practical backtesting, the training time of the models also requires attention. For instance, the Transformer model takes nearly a week to complete a backtest on the US stock market, whereas the SVR model only requires about one hour. In the future, further validation of the RPWS technique will require expanding the dataset, making the SVR model's faster convergence speed a notable advantage for further exploration.

Previous studies typically divide machine learning algorithms and portfolio optimization into two separate stages, which means that the modeling capabilities of machine learning are only utilized in the stock screening phase and not fully exploited during weight optimization. The method proposed in this study, which uses RPWS to construct portfolios, provides a new perspective for the interdisciplinary field of machine learning and portfolio optimization. The key aspect of this method lies in mapping the prediction results of machine learning algorithms to weight biases, successfully integrating the predictive capabilities of machine learning into the weight optimization phase, ensuring its full utilization throughout the portfolio construction process.

In the future, we will explore more weighting methods to reasonably and effectively map the prediction results of machine learning algorithms to weight biases. We have considered several directions to further optimize the initial weight biases, including logarithmic weighting and enforcing log uniform distribution. In particular, enforcing log uniform distribution adjusts the initial weight biases of stocks based on their rankings to conform to a log uniform distribution, and its actual performance will be tested in future backtesting experiments. Additionally, this study uses historical data and technical indi-

cators as input features for predicting future returns. However, many external factors, such as government policies, interest rates, and public events, also impact financial markets, so future work may consider incorporating more sentiment factors into the dataset. We believe that by continuously optimizing weight mapping methods and integrating more influencing factors, the stability and practical value of portfolio strategies will be further enhanced.

REFERENCES

[1] H. Markowitz, "Portfolio selection," *J. Finance*, vol. 7, pp. 77–91, 1952, doi: 10.2307/2975974.

[2] T. Björk, A. Murgoci, and X. Y. Zhou, "Mean-variance portfolio optimization with state-dependent risk aversion," *Math. Finance*, vol. 24, pp. 1–24, 2014, doi: 10.1111/j.1467-9965.2011.00515.x.

[3] K. Liagkouras and K. Metaxiotis, "Multi-period mean–variance fuzzy portfolio optimization model with transaction costs," *Eng. Appl. Artif. Intell.*, vol. 67, pp. 260–269, 2018, doi: 10.1016/j.engappai.2017.10.010.

[4] D. Rosadi, E. P. Setiawan, M. Templ, and P. Filzmoser, "Robust covariance estimators for mean-variance portfolio optimization with transaction lots," *Oper. Res. Perspect.*, vol. 7, p. 100154, 2020, doi: 10.1016/j.orp.2020.100154.

[5] V. N. Katsikis, S. D. Mourtas, P. S. Stanimirović, S. Li, and X. Cao, "Time-varying mean-variance portfolio selection under transaction costs and cardinality constraint problem via beetle antennae search algorithm (BAS)," *Oper. Res. Forum*, vol. 2, 2021, doi: 10.1007/s43069-021-00060-5.

[6] B. Beheshti, "Effective stock selection and portfolio construction within US, international, and emerging markets," *Front. Appl. Math. Stat.*, vol. 4, 2018, doi: 10.3389/fams.2018.00017.

[7] G. S. M. Thakur, R. Bhattacharyya, and S. Sarkar (Mondal), "Stock portfolio selection using Dempster–Shafer evidence theory," *J. King Saud Univ.- Comput. Inf. Sci.*, vol. 30, pp. 223–235, 2018, doi: 10.1016/j.jksuci.2016.07.001.

[8] Y. Zhang, X. Li, and S. Guo, "Portfolio selection problems with Markowitz's mean–variance framework: a review of literature," *Fuzzy Optim. Decis. Making*, pp. 125–158, 2018, doi: 10.1007/s10700-017-9266-z.

[9] P. N. Kolm, R. Tütüncü, and F. J. Fabozzi, "60 years of portfolio optimization: Practical challenges and current trends," *Eur. J. Oper. Res.*, pp. 356–371, 2014, doi: 10.1016/j.ejor.2013.10.060.

[10] Y. Chen and Y. Hao, "Integrating principle component analysis and weighted support vector machine for stock trading signals prediction," *Neurocomputing*, vol. 321, pp. 381–402, 2018, doi: 10.1016/j.neucom.2018.08.077.

[11] F. D. Paiva, "Decision-making for financial trading: A fusion approach of machine learning and portfolio selection," *Expert Syst. Appl.*, vol. 115, pp. 635–655, 2019, doi: 10.1016/j.eswa.2018.08.003.

[12] E. Chong, C. Han, and F. C. Park, "Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies," *Expert Syst. Appl.*, pp. 187–205, 2017, doi: 10.1016/j.eswa.2017.04.030.

[13] H. Liu and M. Loewenstein, "Optimal portfolio selection with transaction costs and finite horizons," *Rev. Financ. Stud.*, vol. 15, pp. 805–835, 2002, doi: 10.1093/rfs/15.3.805.

[14] O. D. Anderson, G. E. P. Box, and G. M. Jenkins, "Time series analysis: Forecasting and control," *The Statistician*, p. 265, 1978, doi: 10.2307/2988198.

[15] R. F. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation," *Econometrica*, vol. 50, pp. 987–1007, 1982, doi: 10.2307/1912773.

[16] N. Shephard, "Statistical aspects of ARCH and stochastic volatility," in *Time Series Models in Econometrics, Finance and Other Fields*, pp. 1–67, 1996.

[17] T. Bollerslev, "Generalized autoregressive conditional heteroskedasticity," *J. Econom.*, pp. 307–327, 1986, doi: 10.1016/0304-4076(86)90063-1.

[18] R. C. Garcia, J. Contreras, M. van Akkeren, and J. B. C. Garcia, "A GARCH forecasting model to predict day-ahead electricity prices," *IEEE Trans. Power Syst.*, vol. 20, pp. 867–874, 2005, doi: 10.1109/TPWRS.2005.846044.

[19] M. A. Villegas, D. J. Pedregal, and J. R. Trapero, "A support vector machine for model selection in demand forecasting applications," *Comput. Ind. Eng.*, vol. 121, pp. 1–7, 2018, doi: 10.1016/j.cie.2018.04.042.

[20] T. Kavzoglu and A. Teke, "Predictive performances of ensemble machine learning algorithms in landslide susceptibility mapping using random forest, extreme gradient boosting (XGBoost) and natural gradient boosting (NGBoost)," *Arab. J. Sci. Eng.*, vol. 47, pp. 7367–7385, 2022, doi: 10.1007/s13369-022-06560-8.

[21] Y. L. Ma, R. Z. Han, and W. Z. Wang, "Portfolio optimization with return prediction using deep learning and machine learning," *Expert Syst. Appl.*, p. 113973, 2021, doi: 10.1016/j.eswa.2020.113973.

[22] A.-L. Boulesteix and M. Schmid, "Machine learning versus statistical modeling," *Biom. J.*, vol. 56, pp. 588–593, 2014, doi: 10.1002/bimj.201300226.

[23] J. Zhang, Z. Li, Z. Pu, and C. Xu, "Comparing prediction performance for crash injury severity among various machine learning and statistical methods," *IEEE Access*, vol. 6, pp. 60079–60087, 2018, doi: 10.1109/ACCESS.2018.2874979.

[24] F. D. Freitas, A. F. de Souza, and A. R. de Almeida, "Prediction-based portfolio optimization model using neural networks," *Neurocomputing*, vol. 72, pp. 2155–2170, 2009, doi: 10.1016/j.neucom.2008.08.019.

[25] Z. Jiang, R. Ji, and K.-C. Chang, "A machine learning integrated portfolio rebalance framework with risk-aversion adjustment," *J. Risk Financ. Manag.*, vol. 13, p. 155, 2020, doi: 10.3390/jrfm13070155.

[26] T. Kaczmarek and K. Perez, "Building portfolios based on machine learning predictions," *Econ. Res.-Ekon. Istraž.*, pp. 19–37, 2022, doi: 10.1080/1331677X.2021.1875865.

[27] D. K. Padhi, N. Padhy, A. K. Bhoi, J. Shafi, and S. H. Yesuf, "An intelligent fusion model with portfolio selection and machine learning for stock market prediction," *Comput. Intell. Neurosci.*, pp. 1–18, 2022, doi: 10.1155/2022/7588303.

[28] S. Deng and X. Min, "Applied optimization in global efficient portfolio construction using earning forecasts," *J. Invest.*, vol. 22, pp. 104–114, 2013, doi: 10.3905/joi.2013.22.4.104.

[29] M. Ballings, D. Van den Poel, N. Hespeels, and R. Gryp, "Evaluating multiple classifiers for stock price direction prediction," *Expert Syst. Appl.*, pp. 7046–7056, 2015, doi: 10.1016/j.eswa.2015.05.013.

[30] A. Chaweewanchon and R. Chaysiri, "Markowitz mean-variance portfolio optimization with predictive stock selection using machine learning," *Int. J. Financ. Stud.*, vol. 10, no. 3, Art. no. 64, 2022, doi: 10.3390/ijfs10030064.

[31] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Appl. Soft Comput.*, pp. 525–538, 2018, doi: 10.1016/j.asoc.2018.04.024.

[32] C.-F. Tsai and Y.-C. Hsiao, "Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches," *Decis. Support Syst.*, pp. 258–269, 2010, doi: 10.1016/j.dss.2010.08.028.

[33] N. N. Vo, X. He, S. Liu, and G. Xu, "Deep learning for decision making and the optimization of socially responsible investments and portfolio," *Decis. Support Syst.*, vol. 124, p. 113097, 2019, doi: 10.1016/j.dss.2019.113097.

[34] W. Wang, W. Li, N. Zhang, and K. C. Liu, "Portfolio formation with preselection using deep learning from long-term financial data," *Expert Syst. Appl.*, vol. 143, p. 113042, 2020, doi: 10.1016/j.eswa.2019.113042.

[35] Y. L. Ma, R. Z. Han, and W. Z. Wang, "Portfolio optimization with return prediction using deep learning and machine learning," *Expert Syst. Appl.*, vol. 165, p. 113973, 2021, doi: 10.1016/j.eswa.2020.113973.

[36] O. Ustun and R. Kasimbeyli, "Combined forecasts in portfolio optimization: A generalized approach," *Comput. Oper. Res.*, vol. 39, pp. 805–819, 2012, doi: 10.1016/j.cor.2010.09.008.

[37] J. R. Yu, W. J. P. Chiou, W. Y. Lee, and S. J. Lin, "Portfolio models with return forecasting and transaction costs," *Int. Rev. Econ. Finance*, vol. 66, pp. 118–130, 2020, doi: 10.1016/j.iref.2019.11.002.

[38] J. Behera, A. K. Pasayat, H. Behera, and P. Kumar, "Prediction based mean-value-at-risk portfolio optimization using machine learning regression algorithms for multi-national stock markets," *Eng. Appl. Artif. Intell.*, vol. 120, pp. 105843, 2023, doi: 10.1016/j.engappai.2023.105843.

[39] F. D. Freitas, A. F. de Souza, and A. R. de Almeida, "Autoregressive neural network predictors in the Brazilian stock markets," in *IEEE Latin American Robotics Symposium*, Brazil, 2005.

[40] C. Y. Hao, J. Q. Wang, and W. Xu, "Prediction-based portfolio selection model using support vector machines," in *6th International Conference on Business Intelligence and Financial Engineering*, pp. 567–571, 2013, doi: 10.1109/BIFE.2013.118.

[41] Y. L. Ma, R. Z. Han, and W. Z. Wang, "Prediction-based portfolio optimization models using deep neural networks," *IEEE Access*, vol. 8, pp. 115393–115405, 2020, doi: 10.1109/ACCESS.2020.3003819.

[42] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 785–794, 2016, doi: 10.1145/2939672.2939785.

[43] G. Lin, A. Lin, and D. Gu, "Using support vector regression and k-nearest neighbors for short-term traffic flow prediction based on maximal information coefficient," *Inf. Sci.*, pp. 517–531, 2022, doi: 10.1016/j.ins.2022.06.090.

[44] Y. Qiu, D. Garg, S.-M. Kim, I. Mudawar, and C. R. Kharangate, "Machine learning algorithms to predict flow boiling pressure drop in mini/micro-channels based on universal consolidated data," *Int. J. Heat Mass Transf.*, p. 121607, 2021, doi: 10.1016/j.ijheatmasstransfer.2021.121607.

[45] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, pp. 273–297, 1995, doi: 10.1023/A:1022627411411.

[46] S. Emir, "Predicting the Istanbul stock exchange index return using technical indicators," *Int. J. Finance Bank. Stud.*, vol. 2, pp. 111–117, 2016, doi: 10.20525/ijfbs.v2i3.158.

[47] J. M. Matías and J. C. Reboredo, "Forecasting performance of nonlinear models for intraday stock returns," *J. Forecast.*, vol. 31, pp. 172–188, 2012, doi: 10.1002/for.1218.

[48] R. I. Rasel, N. Sultana, and P. Meesad, "An efficient modelling approach for forecasting financial time series data using support vector regression and windowing operators," *Int. J. Comput. Intell. Stud.*, vol. 4, p. 134, 2015, doi: 10.1504/IJCISTUDIES.2015.071180.

[49] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[50] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, pp. 654–669, 2018, doi: 10.1016/j.ejor.2017.11.054.

[51] Y. Gao, R. Wang, and E. Zhou, "Stock prediction based on optimized LSTM and GRU models," *Sci. Program.*, pp. 1–8, 2021, doi: 10.1155/2021/4055281.

[52] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Adv. Neural Inf. Process. Syst.*, 2017, doi: 10.5555/3295222.3295349.

[53] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[54] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proc. AAAI Conf. Artif. Intell.*, pp. 11106–11115, 2021, doi: 10.1609/aaai.v35i12.17325.

[55] Y. Lu, H. Zhang, and Q. Guo, "Stock and market index prediction using informer network," *arXiv preprint arXiv:2305.14382*, 2023, doi: 10.48550/arXiv.2305.14382.

[56] D. P. Gandhmal and K. Kumar, "Systematic analysis and review of stock market prediction techniques," *Comput. Sci. Rev.*, vol. 34, p. 100190, 2019, doi: 10.1016/j.cosrev.2019.08.001.

[57] W. Chen, M. Jiang, W.-G. Zhang, and Z. Chen, "A novel graph convolutional feature based convolutional neural network for stock trend prediction," *Inf. Sci.*, vol. 556, pp. 67–94, 2021, doi: 10.1016/j.ins.2020.12.068.

[58] Evans, J.; Archer, S. Diversification and the reduction of dispersion: An empirical analysis. J. Financ. 1968, 23, 761–767.

[59] Alexeev, V.; Dungey, M. Equity portfolio diversification with high-frequency data. Quant. Financ. 2015, 15, 1205–1215.