MILLION: A General Multi-Objective Framework with Controllable Risk for Portfolio Management

Liwei Deng
University of Electronic Science and Technology of China
deng_liwei@std.uestc.edu.cn

Tianfu Wang
University of Science and Technology of China
tianfuwang@mail.ustc.edu.cn

Yan Zhao
University of Electronic Science and Technology of China✉
yanz@cs.aau.dk

Kai Zheng*
University of Electronic Science and Technology of China✉
zhengkai@uestc.edu.cn

ABSTRACT

Portfolio management is an important yet challenging task in AI for FinTech, which aims to allocate investors' budgets among different assets to balance the risk and return of an investment. In this study, we propose a general **M**ulti-object**I**ve framework with contro**LL**able r**I**sk for p**O**rtfolio ma**N**agement (**MILLION**), which consists of two main phases, i.e., return-related maximization and risk control. Specifically, in the return-related maximization phase, we introduce two auxiliary objectives, i.e., return rate prediction, and return rate ranking, combined with portfolio optimization to remit the overfitting problem and improve the generalization of the trained model to future markets. Subsequently, in the risk control phase, we propose two methods, i.e., portfolio interpolation and portfolio improvement, to achieve fine-grained risk control and fast risk adaption to a user-specified risk level. For the portfolio interpolation method, we theoretically prove that the risk can be perfectly controlled if the to-be-set risk level is in a proper interval. In addition, we also show that the return rate of the adjusted portfolio after portfolio interpolation is no less than that of the min-variance optimization, as long as the model in the reward maximization phase is effective. Furthermore, the portfolio improvement method can achieve greater return rates while keeping the same risk level compared to portfolio interpolation. Extensive experiments are conducted on three real-world datasets. The results demonstrate the effectiveness and efficiency of the proposed framework.

*Liwei Deng and Tianfu Wang are equally contributed to this work. ✉Yan Zhao and Kai Zheng are corresponding authors. They are with School of Computer Science and Engineering, and Shenzhen Institute for Advanced Study, UESTC.

1 INTRODUCTION

Portfolio management is an essential component of a trading system, which allocates a budget among different possible financial assets according to different objectives, such as maximizing returns at a given risk level [16, 24, 29, 41]. In 1952, Markowitz introduced a pioneer work, called Modern Portfolio Theory (MPT) [29]. MPT aims to construct a portfolio by solving a combinational optimization problem, leading to a higher return per risk than trading an individual asset [46]. Recently, the benefit of the portfolio compared with investing a single asset is further confirmed [46, 51], e.g., Eric Zivot [51] shows that the risk of a long-only portfolio is always lower than that of an individual asset, for a given expected return, as long as assets are not perfect correlated. Due to the desirable property of investing a bucket of assets, portfolio management has drawn much attention over the past decades.

Existing studies on portfolio management can be roughly divided into three main lines, i.e., predict-then-optimize, reinforcement learning (RL) based methods, and deep learning (DL) based approaches, based on different optimization ways. The methods along the first line [4, 5, 12, 13, 16, 20, 50] first estimate future price or return rate of each asset, and then solve a combinational optimization problem, e.g., mean-variance model, to obtain the final portfolio. For example, Huang et al. [16] forecast the future price of each asset through a sliding window-based moving average and then optimize the robust median reversion problem with the estimated prices to obtain the portfolio. Despite its ability of great ease of use, the performance of these methods is strongly related to the accuracy of the forecasting model. Unfortunately, the accurate asset price or return rate cannot be accessible due to the volatility of the dynamic market. Thus, rather than focusing on accurate price prediction, RL-based methods [1, 14, 17, 24, 25, 35] aim to directly obtain a portfolio from the observed market state through maximizing the defined reward function. For example, Liu et al. [25] propose a general RL-based framework to achieve automated trading, in which they adopt classical RL methods, such as Proximal Policy Optimization (PPO), to optimize the neural networks using the cumulative return as their reward. However, these methods ignore the fact that the portfolio optimization problem is different from others, such as video gaming and cheesing, where the reward from many investing goals, such as Sharpe ratio [32] and cumulative return, is differentiable. Therefore, optimizing from a surrogate loss in RL is inefficient. The methods in the last line [42, 44–46]

overcome this issue through directly optimizing the objective. For example, Zhang et al. [46] develop an end-to-end DL-based model and directly adopt Sharpe ratio as their objective.

Despite growing interest in the last direction, existing studies are still limited to the following aspects. First, the market is highly dynamic and the historical information of each asset has a relatively low signal-to-noise ratio, which causes a poor generalization of the trained model to the future market. Current studies on this problem either adopt simple architecture [45, 46], or incorporate outsourcing information [37], such as financial news [7, 8, 22, 28, 44, 48], to enrich the information ratio in raw data. The former approaches may not be effective enough to extract the necessary features, while the outsourcing information may not be always available in the latter approaches. Second, different investors may have different tastes in taking risks. For example, an investor with limited wealth usually prefers to allocate their budget to low-risk bonds and stocks, while an investor with enough spare money prefers to take more risks to achieve higher expected returns. Current learning-based approaches only return a fixed portfolio for a given market state, which cannot satisfy different investors' demands with different risk levels. Moreover, existing DL-based methods cannot achieve fine-grained risk control. For example, Zhang et al. [45] combine the return objective and risk term using a Lagrange form, where the risk multiplier is hard to preset, i.e., higher weights risk underfitting the return while lower weights weaken risk control.

To tackle these problems, we propose a general **M**ulti-object**I**ve framework with contro**LL**able r**I**sk for p**O**rtfolio ma**N**agement, named MILLION. In summary, we decompose the mean-variance optimization [29] into two phases, i.e., return-related maximization and risk control. Specifically, in the first phase, rather than introducing outsourcing data to improve the data quantity, we only leverage the assets' historical price and volume information despite our framework also being easy to modify to incorporate other source information. To improve the generalization of the trained model to the future market, we introduce two auxiliary objectives, i.e., assets return rate forecasting and assets return rate ranking, which are highly related to portfolio construction. Furthermore, to control risk to a user-specified risk level, we propose a simple but effective portfolio interpolation method in the risk control phase, in which the constructed portfolio is obtained through interpolation between the portfolios from the reward maximization phase and the min-variance optimization. We theoretically prove that the risk can be perfectly controlled as long as the given risk level is in a risk interval. Besides, we also theoretically show that the expected return of the portfolio after interpolation is always greater than the portfolio from the min-variance optimization if the reward maximization is effective. In addition, based on the idea of the interpolation approach, we further propose a portfolio improvement method to achieve higher portfolio return with the same risk level compared with the interpolation method. It should be emphasized that the risk control components can be appended to any existing portfolio construction approaches to help them control the risk.

In summary, our contributions are as follows:

- We propose a general DL-based framework with multi-objective learning to improve the generalization of the trained model to perform better in the future market.

- We develop two risk control approaches, i.e., portfolio interpolation and portfolio improvement, which can be used to fit investors' personalized risk preferences. According to the best of our knowledge, this is the first attempt to achieve fine-grained risk control in learning based portfolio construction.
- We conduct extensive experiments to demonstrate the effectiveness of MILLION and its components.

2 PRELIMINARIES

We present necessary concepts and define the problem addressed.

Definition 2.1 (Holding Period). *A holding period is the minimum time unit to invest an asset. We follow previous studies that divide the whole investment period into multiple non-overlapped holding periods with fixed length, such as one day or one month.*

Definition 2.2 (Asset Prices). *The price of an asset is defined as a time series $\boldsymbol{p}^i = \{p_1^i, p_2^i, ..., p_t^i\}$, where p_t^i is the price of asset i at t.*

Definition 2.3 (Return Rate). *The return rate of an asset i at time t is defined as $r_t^i = p_{t+1}^i/p_t^i - 1$, which presents that if an investor spent n cash to buy asset i at time t, he can get profit $n * r_t^i$.*

Definition 2.4 (Risk). *Following the risk definition in MPT [29], we define the risk as volatility (i.e., variance) of return rate (i.e., $\sigma_i^2 = E[(r_t^i - \bar{r}^i)^2]$, where $\bar{r}^i$ is the expectation of r^i). The idea behind this definition is that the return rate of an asset has a lower variance, and the certainty of investing in this asset is higher, which induces lower risk. In this study, σ_i^2 is calculated from a sliding window of historical asset return rate, in which the window size is consistent with w in temporal modeling (cf. Section 3.1.2).*

Definition 2.5 (Long Position). *The long position is to buy an asset i at time t_1 and then sell it at time t_2, aiming for a profit derived from an increase in the asset's price over this period. The produced profit can be formulated as $n * r_{t_1,t_2}^i$, where n and r_{t_1,t_2}^i denotes the investment amount and return rate from t_1 to t_2 of asset i, respectively.*

Conversely, a short position operation represents the opposite strategy, where investors profit from the decline in an asset's price. In this study, we prohibit the short position operation, i.e., investors can only get profits when the prices of invested assets increase.

Definition 2.6 (Portfolio). *Given N assets to be invested, a portfolio is defined as a vector $\boldsymbol{b} = (b^1, b^2, ..., b^N)$, where b^i presents the proportion of the investment on asset i and $\sum_{i=1}^N b^i = 1$. The return rate of portfolio $\boldsymbol{b}$ is $\boldsymbol{b}^T \boldsymbol{r}$, where $\boldsymbol{r}$ indicates the return rates of N assets. The risk of portfolio is $\boldsymbol{b}^T \Sigma \boldsymbol{b}$, where $\Sigma \in \mathcal{R}^{N \times N}$ is the return rate covariance matrix of N assets.*

Definition 2.7 (Portfolio Management). *Portfolio management is a sequential investment, which determines portfolio $\boldsymbol{b}$ at the end of each holding period. We pursue to achieve two goals in this study: (1) maximizing the portfolio return; (2) controlling the portfolio risk to a user-specific risk level.*

3 METHODOLOGY

We propose a general multi-objective framework with controllable risk for portfolio management, named MILLION, as shown in Figure 1, which consists of two main phases, i.e., return-related maximization and risk control. In the return-related maximization phase,
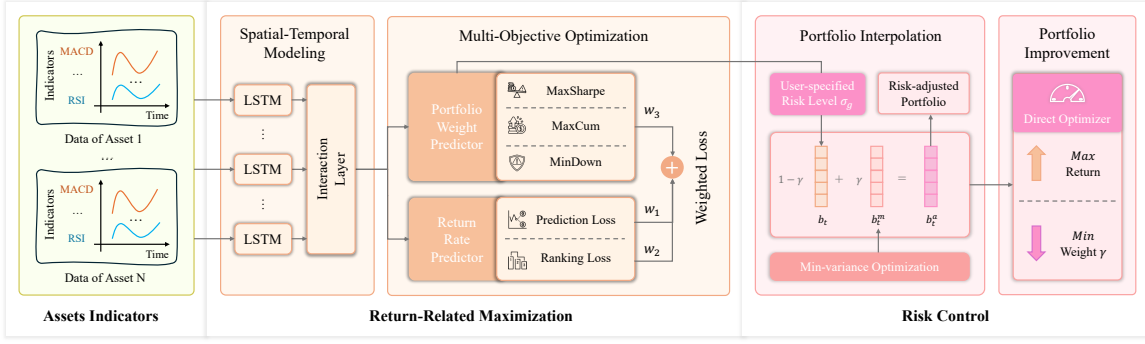
Figure 1: The proposed MILLION framework

we adopt a DL-based spatio-temporal model to construct a portfolio for each time step to maximize the objectives (e.g., Sharpe ratio) that are related to the portfolio return, which provides a significant signal to get profits from the investment behaviors. Then, in the risk control phase, we propose two novel methods (i.e., portfolio interpolation and portfolio improvement) to adjust the portfolio obtained from the return-related maximization phase to fulfill a user-specified risk level. Each component is elaborated in the following sections.

3.1 Return Maximization

Earning money is the primary goal for most investors, in which understanding the current market state is a fundamental and critical step to provide instructions to guide the investors' decision process. In this study, rather than designing a new DL-based model to effectively extract more powerful and predictive representations from raw features, we focus on developing a general framework that can fit various models. Thus, we directly adopt existing models, i.e., LSTMHA [9], with slight modifications to encode the market state. It should be noted that another model with the ability of spatial-temporal modeling can also be adopted to encode the state of assets [6, 26, 38, 40, 49]. As shown in the left panel of Figure 1, we model the spatial (i.e., relations among assets) and temporal (i.e., relations along timestamp) information with attention technique and LSTM, respectively, in which the covariance matrix between assets' historical return rate is incorporated into the attention module. After the representation of each asset is obtained, a multi-objective optimization module is appended, in which two portfolio-related objectives are adopted to improve the model generalization.

3.1.1 Assets Indicators.
Following previous studies, e.g., FinRL [25], we incorporate the eight indicators derived from the prices and volumes as our model input. These indicators include Moving Average Convergence/Divergence (MACD), Bollinger Bands (BOLL) (i.e., lower bound and upper bound of BOLL), Relative Strength Index (RSI), Commodity Channel Index (CCI), Directional Movement Index (DMI), and Simple Moving Average (SMA) (i.e., 30-days and 60-days). Given the disparate scales across these financial metrics, we employ Z-score normalization to standardize them.

3.1.2 Spatio-Temporal Modeling.
To construct an effective portfolio, we necessitate insight into the future market, particularly regarding the performance (e.g., return rate) of individual assets.

To achieve this goal, we differentiate the modeling of each asset into temporal and spatial relations.

Temporal Modeling. We use the vector x_t^i to denote the history state of asset i at time t, which consists of eight indicators as stated in section 3.1.1. Thus, the current state of asset i at time t is presented by a window size of x_t^i (i.e., $X_t^i = \{x_{t-w}^i, \cdots, x_t^i\}$). A one-layer *LSTM* is adopted to recursively encode X^i into a vector.

$$h_t^i = LSTM(X_t^i) \tag{1}$$

where $h_t^i \in \mathcal{R}^d$ is the representation of asset i at time t, which models the intra-correlation along the timestamps.

Spatial Modeling. Despite the temporal dependency is encoded into h_t^i, the inter-correlation among assets are not included. Therefore, we leverage the attention technique to model the dynamic relations among assets, in which the covariance matrix between assets' historical return rates is integrated to remit the burden in the learning process.

$$\hat{h}_t^i = \frac{1}{\beta+1} \sum_{k=1}^{N} \alpha_k \cdot h_t^k + \frac{\beta}{\beta+1} \sum_{k=1}^{N} c_{i,k} \cdot h_t^k$$

$$\alpha_k = \frac{\exp(W h_t^k)}{\sum_{j=1}^{N} \exp(W h_t^j)} \tag{2}$$

where $W_a \in \mathcal{R}^{d_1 \times d}$ is the to-be-learned parameters, β is a scalar to balance the weight between attention weights and covariance matrix, which is updated with the training goes on, $c_{i,k}$ presents the covariance between asset i and k, and $\hat{h}_t^i$ is the encoded representation for asset i at time t.

Portfolio Construction. Based on the extracted feature from previous modules, we adopt a two-layer *MLP* with *ReLU* activation to evaluate each asset and construct a portfolio as follows:

$$v_t^i = MLP_p(\hat{h}_t^i)$$

$$b_t^i = \frac{exp(v_t^i)}{\sum_{k=1}^{N} exp(v_t^k)} \tag{3}$$

where v_t^i indicates the estimated valuation of investing on asset i, and b_t^i is the portfolio weight of asset i.

3.1.3 Multi-Objective Optimization.
After obtaining the portfolio weight at each time step, we can directly optimize the Sharpe ratio or the cumulative return as the same with previous studies [45, 46]. However, while neural networks offer powerful representational ability, they often face challenges in generalizing to future market conditions effectively. To remit this problem, we propose a

shift from a singular objective to a multiple objective optimization framework, which incorporates two auxiliary loss functions: asset return rate prediction and asset return rate ranking. In the latter, we first elaborate the portfolio optimization and then the auxiliary optimization, separately.

Portfolio Optimization. To optimize the constructed portfolio, there are two common objectives [46] as follows:

$$(MaxCum)\,\mathcal{L}_{mc} = \prod_{t=1}^{T}(r_t^p + 1)$$
$$(MaxSharpe)\,\mathcal{L}_{ms} = \frac{Mean(\{r_t^p\}_{t=1}^T)}{Std(\{r_t^p\}_{t=1}^T)} \tag{4}$$
$$r_t^p = b_t^T r_t - c_t |b_t^T - b_{t-1}^T|_1$$

where r_t represents the return rate of each assets at time t, r_t^p is the corresponding portfolio return rate, c_t is the transaction cost rate, and T is the total number of holding period. *MaxCum* focuses on maximizing the cumulative return, which will drive the model to construct a centralized portfolio (i.e., investing in a single asset that may achieve the highest return rate among other assets). *MaxSharpe* not only focuses on maximizing the portfolio return rate at each time but also aims to minimize its standard deviation, which will impose the model to construct a relatively conservative portfolio. Except for the two commonly used objectives, we also develop another objective as follows:

$$(MinDown)\,\mathcal{L}_{md} = -\sum_{t=1}^{T}\max(-r_t^p + \delta_d, 0) \tag{5}$$

where δ_d indicates a threshold, which presents the investors' expected return rate in each holding period. The goal of *MinDown* is to construct portfolios that can achieve a return rate larger than the given threshold δ_d. The lower δ_d is, the more conservative the constructed portfolio is, which will endow the model with roughly risk-control ability. Moreover, δ_d is unnecessary to be fixed, which can be dynamic according to the situation of the current market state. For example, a simple implementation is to replace δ_d with the return rate of a benchmark such as NAS100 index.

Auxiliary Optimization. Directly optimizing a single objective in portfolio optimization, the trained model may suffer from the overfitting problem (i.e., it has poor generalization in the future market) due to the highly dynamic market and low signal-to-noise ratio in historical information. To remit this problem, we introduce two auxiliary objectives [8, 10, 11, 34, 47], which are optimized combining with the objective in portfolio optimization.

$$(Prediction)\,\mathcal{L}_p = \sum_{t=1}^{T}||\hat{r}_t - r_t||_2$$
$$(Ranking)\,\mathcal{L}_r = \sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{j=1}^{N}\max(-(\hat{r}_t^i - \hat{r}_t^j)(r_t^i - r_t^j), 0) \tag{6}$$

where $\hat{r}_t = [\hat{r}_t^1, \hat{r}_t^2, \cdots, \hat{r}_t^N]$ and $r_t = [r_t^1, r_t^2, \cdots, r_t^N]$ is the predicted and the ground-truth return rate of N assets. $\hat{r}_t$ is obtained through a *MLP* with $\hat{h}_t^i$ as inputs. This neural networks share the same architecture with MLP_p in portfolio construction but with different parameters. With these objectives, we define our final optimization objective as follows:

$$\mathcal{L} = -\zeta_m \mathcal{L}_* + \zeta_p \mathcal{L}_p + \zeta_r \mathcal{L}_r \tag{7}$$

where $*$ is in $\{mc, ms, md\}$, and ζ_m, ζ_p, and ζ_r are the weights to balance the contributions of different objectives. In our experiments, we notice the performance of the constructed portfolio is sensitive to the weights. To reduce the efforts to search an optimal weights, we follow Kendal et al. [18] to set the weights adaptively.

$$\mathcal{L} = -\frac{1}{\zeta_m^2}\mathcal{L}_* + \frac{1}{\zeta_p^2}\mathcal{L}_p + \frac{1}{\zeta_r^2}\mathcal{L}_r + \sum_{i\in\{m,p,r\}}\log\zeta_i \tag{8}$$

where ζ_m, ζ_p, and ζ_r are to-be-learned parameters, which are adaptively updated in the training phase. The parameters of the neural networks are optimized through minimizing $\mathcal{L}$ in Equation 8.

3.2 Risk Control

Despite risk management being a vital component in portfolio management, current DL-based and RL-based approaches are hard to achieve fine-grained risk control. For example, Zhang et al. [45] utilize neural networks to optimize the Lagrange formulation mean-variance model, in which the risk term is weighted and added to the portfolio return. It can achieve rough risk control in the training phase but without guaranteeing the unseen data. Moreover, existing studies always construct one portfolio for all investors at each holding period, which cannot satisfy investors' different preferences for risk-taking. Thus, we propose two methods, i.e., portfolio interpolation and portfolio improvement, to deal with these issues, which are elaborated as follows.

3.2.1 Portfolio Interpolation. We denote b_t^m as the portfolio at time t obtained from the min-variance optimization, which has the lowest risk compared to any other portfolios. To control the risk of the constructed portfolio from the return maximization phase, we obtain the risk-adjusted portfolio b_t^a with interpolation as follows:

$$b_t^a = (1 - \gamma_t)b_t + \gamma_t b_t^m \tag{9}$$

where $\gamma_t \in [0, 1]$ is the weight to control the amount of interpolation. With the above interpolation method, we have the following proposition, in which we denote σ_t^a, σ_t, and σ_t^m as the risk of the three portfolios b_t^a, b_t, and b_t^m, respectively (e.g., $\sigma_t = b_t^T \Sigma_t b_t$).

PROPOSITION 3.1. σ_t^a is a decreasing monotone function in terms of γ_t if $\delta_t \neq \delta_t^m$, whose value is in the interval $[\sigma_t^m, \sigma_t]$.
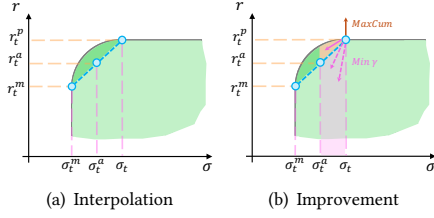PROOF.
$$\begin{aligned}\sigma_t^a &= b_t^{aT}\Sigma_t b_t^a \\ &= [(1-\gamma_t)b_t + \gamma_t b_t^m]^T \Sigma_t [(1-\gamma_t)b_t + \gamma_t b_t^m] \\ &= (b_t^T \Sigma_t b_t - 2b_t^T \Sigma_t b_t^m + b_t^{mT}\Sigma_t b_t^m)\gamma_t^2 \\ &\quad + 2(b_t^T \Sigma_t b_t^m - b_t^T \Sigma_t b_t)\gamma_t + b_t^T \Sigma_t b_t\end{aligned} \tag{10}$$

We represent the portfolios with the eigenvectors of Σ_t (i.e., $b_t = \sum_{i=1}^N c_i x_i$ and $b_t^m = \sum_{i=1}^N d_i x_i$). The eigenvalue of x_i is denoted as λ_i. It should be noted that Σ_t is symmetric and semi-positive whose eigenvalue is non-negative (i.e., $\lambda_i \geq 0, \forall i \in [1, N]$).

$$\begin{aligned}b_t^T \Sigma_t b_t - 2b_t^T \Sigma_t b_t^m + b_t^{mT}\Sigma_t b_t^m &= \sum_{i=1}^N \lambda_i c_i^2 - 2\sum_{i=1}^N \lambda_i c_i d_i + \sum_{i=1}^N \lambda_i d_i^2 \\ &= \sum_{i=1}^N \lambda_i (c_i - d_i)^2 \geq 0\end{aligned} \tag{11}$$

When σ_t^a is a quadratic function in terms of γ_t, we can know that this function is an upward opening and it gets the lowest when γ_t equals 1. Thus, this function is decreasing monotone when γ_t varies from 0 to 1 (i.e., $\sigma_t^a \in [\sigma_t^m, \sigma_t]$). Next, when the quadratic

(a) Interpolation (b) Improvement

Figure 2: An illustration of portfolio interpolation.

term equals 0, this function degenerates to a linear function. It should be noted that $\sigma_t^m \leq b_t^{*T} \Sigma_t b_t^*$ from the definition of b_t^m, where b_t^* represents any portfolio. We analyze the coefficient of the primary term as follows:

$$2(b_t^T \Sigma_t b_t^m - b_t^T \Sigma_t b_t) = 2b_t^T \Sigma_t b_t^m - b_t^T \Sigma_t b_t - b_t^{mT} \Sigma_t b_t^m -$$
$$b_t^T \Sigma_t b_t + b_t^{mT} \Sigma_t b_t^m \quad (12)$$
$$= -\sum_{i=1}^N \lambda_i (c_i - d_i)^2 - (\sigma_t - \sigma_t^m) \leq 0$$

We can see that when the function is linear, it is also decreasing monotone when γ_t in $[0, 1]$. Thus, combining these two situations, we can conclude that $\sigma_t^a \in [\sigma_t^m, \sigma_t]$ and with the increase of γ_t, the portfolio risk will be gradually decreased if $\sigma_t^m \neq \sigma_t$. □

With proposition 3.1, we can control the portfolio risk to a user-specified risk level σ_g by replacing the left of Equation (10) with σ_g and solving γ_t as long as $\sigma_g \in [\sigma_t^m, \sigma_t]$. Since this equation is a quadratic equation with only γ_t unknown, it is easy to calculate the exact value of γ_t when the expected risk is given. Therefore, the portfolio can be fast adapted to satisfy different investors' requests in terms of risk level. After analysing the effect of risk through interpolation, we continue to study the effect on the portfolio return rate. We have the following proposition, in which we denote r_t^a, r_t^p, and r_t^m as the return rate of the three portfolios b_t^a, b_t, and b_t^m, respectively (e.g., $r_t^p = b_t^T r_t$).

PROPOSITION 3.2. *r_t^a is always no less than r_t^m if the model is effective (i.e., $r_t^p \geq r_t^m$).*
PROOF.
$$r_t^a = [(1 - \gamma_t) b_t + \gamma_t b_t^m]^T r_t = (1 - \gamma_t) b_t^T r_t + \gamma_t b_t^{mT} r_t$$
$$= (1 - \gamma_t) r_t^p + \gamma_t r_t^m \quad (13)$$

From above equation, we can see that the portfolio return rate is a linear function in terms of γ_t, which achieves the lowest r_t^m when γ_t equals to 1. Thus, r_t^a is no less than r_t^m. □

Proposition 3.2 demonstrates that no matter how we interpolate, the return rate of b_t^a is always bounded by it of b_t (i.e., upper bound) and b_t^m (i.e., lower bound), which means that the portfolio interpolation method is safe and will not generate a portfolio that causes dramatic loss.

3.2.2 Portfolio Improvement. Despite the advantages of the portfolio interpolation method, the portfolio after interpolation may be far away from the efficient frontier, which means there is an opportunity to improve the portfolio return while keeping the user-specified risk unchanged. We provide an illustration of this situation as shown in Figure 2(a). The horizontal and vertical axis represent the risk and return, respectively. The blue dash line presents the portfolio interpolation . From this figure, we can see that there may exist an orange area that cannot be obtained through interpolation,

in which there exist points that have the same risk as the interpolated point but have higher returns. To reach these points, we propose a portfolio improvement approach, which is to optimize the portfolio from return maximization to push it to the orange area as shown in Figure 2(b). Then, the portfolio interpolation is adopted to control risk, in which the interpolated point is expected to have a higher return with the same risk compared with the portfolio from the pure portfolio interpolation method. Before introducing our approach, we first present a proposition as follows:

PROPOSITION 3.3. *Assume we have a set of portfolios $\{b_t^1, b_t^2, \cdots, b_t^i, \cdots\}$ whose return rates are the same. We apply the portfolio interpolation method to control their risks to a given risk σ_g. The portfolio with the highest return after interpolation has the lowest γ_t.*

PROOF. Since this proposition can be directly deduced from proposition 3.2, the details of the proof are omitted. □

From this proposition, the portfolio improvement approach is to minimize the interpolation weight γ_t for a user-specified risk level σ_g. Specifically, we solve Equation 10 for a given σ_g to get γ_t, in which γ_t can be presented as a function of σ_g, Σ_t, b_t, and b_t^m. Since γ_t is the root of a quadratic equation with one unknown, the function f is differentiable. Thus, we can directly minimize γ_t to optimize v_t (i.e., the portfolio before *Softmax*, cf. Equation 3) to compel the portfolio constraint (i.e., $\sum_{i=1}^N b^i = 1$).

$$\mathcal{L}_{imp} = \sum_{t=1}^T \gamma_t \quad (14)$$

Moreover, since the goal of personalized risk control is to fit different investors' requests, it is time-consuming to finetune the whole model for each investor. Therefore, the model's parameters are fixed at the portfolio improvement, which can save the amount of calculation.

However, only minimizing γ_t may cause unexpected damage to the portfolio return as shown in Figure 2(b), where the pink area will produce a smaller γ_t but also a smaller portfolio return. To deal with this situation, we incorporate a return objective when optimizing. The return-added objective can be formulated as follows:

$$\mathcal{L}_{imp+ret} = -\zeta \prod_{t=1}^T (b_t^T \hat{r}_t) + \sum_{t=1}^N \gamma_t \quad (15)$$

where $\hat{r}_t$ is the predicted assets' return rate, and ζ is a weight to balance the loss of the two components. It should be noted that the predicted return rate is not constrained to obtained from our predictor in the return-related maximization phase but can be accessed from any predictors, such as LSTM or more advanced models, and training separately.

4 EXPERIMENTS

Dataset. The data from both the U.S. stock market and cryptocurrency market is obtained using FinRL[1] and CCXT[2], and then preprocessed with the FinRL library to extract the indicators detailed in section 3.1.1. In the case of the stock market, we specifically focus on stocks from two prominent U.S. stock indexes, namely NAS100 and DOW30. For the cryptocurrency market, we select the top 10 cryptocurrencies by market occupancy. Table 2 provides an

---
[1]https://github.com/AI4Finance-Foundation/FinRL
[2]https://github.com/ccxt/ccxt

Table 1: Performance comparison between MILLION and competitors on three real-world datasets.

Methods	DOW30					NAS100					Crypto10				
	APR↑	AVOL↓	ASR↑	ACR↑	MDD↑	APR↑	AVOL↓	ASR↑	ACR↑	MDD↑	APR↑	AVOL↓	ASR↑	ACR↑	MDD↑
Market	0.2182	0.1133	1.7994	3.6065	-0.0605	0.3566	0.3704	1.0101	1.1099	-0.3213	-0.0494	0.5612	0.1922	-0.0656	-0.7530
MVM	0.0941	0.1138	0.8472	1.6677	**-0.0565**	0.3907	**0.2914**	1.2796	1.3885	-0.2814	0.1406	**0.4560**	0.5180	0.1939	-0.7252
DT	0.0948	0.1229	0.7979	1.1003	-0.0861	0.9011	0.4100	_1.7731_	_4.4232_	**-0.2037**	-0.1786	0.6937	0.0627	-0.2116	-0.8438
LR	0.2125	0.1416	1.4318	2.5832	-0.0823	0.7128	0.4149	1.5057	2.2923	-0.3109	-0.1489	0.6744	0.0757	-0.1774	-0.8395
RF	0.2326	0.1562	1.4173	2.6315	-0.0884	0.5249	0.3772	1.3082	1.7548	-0.2991	-0.1698	0.7046	0.0830	-0.2175	-0.7806
SVM	0.0047	**0.1105**	0.0980	0.0545	-0.0871	0.8349	_0.3617_	1.6598	3.1439	_-0.2656_	0.0025	0.7448	0.3526	0.0028	-0.8955
LSTM-PTO	0.1808	0.1194	1.4516	2.6715	-0.0677	-0.4360	0.9906	-0.0831	-0.7148	-0.6099	0.0334	0.6555	0.3759	0.0432	-0.7730
LSTMHAM-PTO	0.1044	0.1143	0.9254	1.8236	-0.0572	-0.4640	0.8107	-0.3547	-0.7971	-0.5821	-0.0297	0.7656	0.3403	-0.0436	_-0.6799_
FinRL-A2C	0.2186	0.1134	1.8001	3.5902	-0.0609	0.3630	0.3718	1.0203	1.1285	-0.3217	-0.0451	0.5605	0.2000	-0.0602	-0.7488
FinRL-PPO	0.2379	_0.1128_	1.9492	4.1736	_-0.0570_	0.3572	0.3686	1.0143	1.1170	-0.3198	-0.0835	_0.5521_	0.1204	-0.1095	-0.7624
LSTMHAM-S	0.2731	0.1212	2.0532	3.8952	-0.0701	0.4442	0.5038	0.9794	1.5705	-0.2828	0.1837	0.6858	0.5684	0.2525	-0.7278
LSTMHAM-C	0.2915	0.1198	2.1949	4.1515	-0.0702	1.2153	0.8541	1.3296	2.7640	-0.4396	0.0838	0.6444	0.4357	0.1138	-0.7365
LSTMHAM-M	0.3214	0.1279	2.2432	4.2906	-0.0749	0.1846	0.4949	0.5888	0.5421	-0.3404	0.0532	0.5797	0.3790	0.0714	-0.7449
MILLION-S	**0.3936**	0.1417	_2.4132_	_4.6153_	-0.0806	**1.9763**	0.7243	**1.8528**	**6.9817**	-0.2830	**0.3871**	0.7405	**0.7914**	**0.5821**	**-0.6650**
MILLION-C	_0.3857_	0.2103	1.6572	3.8870	-0.0992	_1.3172_	0.8578	1.3759	3.1930	-0.4125	_0.1996_	0.7279	_0.5888_	_0.2786_	-0.7165
MILLION-M	0.3389	0.1288	**2.5306**	**4.6412**	-0.0736	1.2206	0.5638	1.6917	4.0832	-0.2989	0.0581	0.5797	0.3870	0.0783	-0.7440

overview of the statistics for each dataset. In the case of NAS100 and DOW30 datasets, the data from the last year of the training period is utilized as a validation set for conducting model selection. **Competitors.** We compare three types of competitors. (1) The predict-then-optimize methods are: Decision Tree (DT) [3], Linear Regression (LR), Random Forest (RF) [2], Support Vector Machine (SVM) [30], LSTM-PTO [15], and LSTMHAM-PTO, where they focus on predicting the assets' return rate of next holding period and then solve the classical mean-variance problem through maximizing Sharpe ratio. It should be noted that LSTMHAM-PTO shares the same model architecture with ours. (2) The RL-based methods: A2C and PPO, in which we adopt a RL-based financial lib, i.e., FinRL [24, 25], to implement. (3) The DL-based methods: LSTMHAM-S, LSTMHAM-C, and LSTMHAM-M, which keep the same model architecture as ours but with only one objective to optimize, where -S, -C, and -M indicates the model is trained with *MaxSharpe*, *MaxCum*, and *MinDown*, respectively. We also include two classical methods, i.e., Market and min-variance model (MVM), as competitors.

Evaluation Metrics. We include six commonly-used metrics [5, 44, 46] in portfolio management to evaluate the proposed framework, i.e., Cumulative Wealth (CW), Annualized Percentage Rate (APR), Annualized Volatility (AVOL), Annualized Sharpe Ratio (ASR), Maximum DrawDown (MDD), and Annualized Calmar Ratio (ACR).

Parameter Settings. The holding period is fixed at one day, with a window size (w) of 20 for temporal modeling. Transaction cost (c_t) in Equation 4 is set to 0, while the threshold (δ_d) in Equation 5 is set to 0.005. Optimization is conducted using the *AdamW* optimizer with a learning rate of 1e-4. The neural networks' hidden size (d) is set to 64 for the DOW30 and Crypto10 datasets and 128 for the NAS100 dataset. In subsequent sections, we independently assess the efficacy of our proposed components: return-related maximization and risk control.

4.1 Return-Related Maximization Performance

Overall. Table 1 displays the backtesting outcomes of various models across three datasets, while Figure 3 illustrates cumulative wealth curves. It's evident that among the predict-then-optimize approaches, no single model outperforms others consistently in terms of APR or ASR. This suggests that different market conditions favor different prediction models, making it challenging to

Table 2: Dataset Statistics

Dataset	Training Period	Testing Period	#Assets
NAS100	2009/01/01-2020/02/01	2020/02/01-2021/01/01	78
DOW30	2009/01/01-2019/02/01	2019/02/01-2020/01/01	28
Crypto10	2018/08/02-2021/07/01	2021/07/01-2023/10/32	10

devise a universal model applicable to all scenarios. Additionally, the performance of these methods varies significantly across datasets, indicating the sensitivity of portfolio construction to predicted asset returns. Furthermore, Market and MVM exhibit relatively lower AVOL and MDD yet competitive APR compared to predict-then-optimize methods, underscoring the validity of diversified investment strategies. Notably, RL-based methods excel in DOW30 but not in other datasets. Our experiments reveal that RL-based methods often demand extensive interactions to learn effective policies on training data but struggle with generalization to unseen test data, highlighting training efficiency and generalizability issues. Moreover, LSTMHAM with diverse objectives generally outperforms other benchmarks in most cases. Ultimately, our proposed MILLION framework consistently achieves the best performance in terms of return-related metrics such as APR, ASR, and ACR.

Effect of different frameworks. Comparing the same model architecture with different portfolio construction methods (e.g., LSTMHAM-PTO, FinRL-PPO, LSTMHAM-S), we can find that the DL-based framework always achieves better while the predict-then-optimize framework is comparable with RL-based framework.

Effect of different objectives. Our investigation into the impact of different optimization objectives uncovers distinct advantages associated with each. For instance, the *MaxSharpe* objective typically minimizes MDD to a greater extent compared to other objectives. On the other hand, *MaxCum* tends to achieve superior APR, while *MinDown* consistently minimizes AVOL across most scenarios.

Effect of multiple objectives. Compared with single objective DL-based methods (e.g., LSTMHAM-S), our proposed framework MILLION (e.g., MILLION-S) can always perform better in terms of return-related metrics, which shows the effectiveness of using multiple objectives to train the DL-based models.

4.2 Risk Control Performance

We conduct a comparative analysis of the proposed risk control methodologies alongside predict-then-optimize strategies, as illustrated in Figure 5. The horizontal axis represents user-defined risk
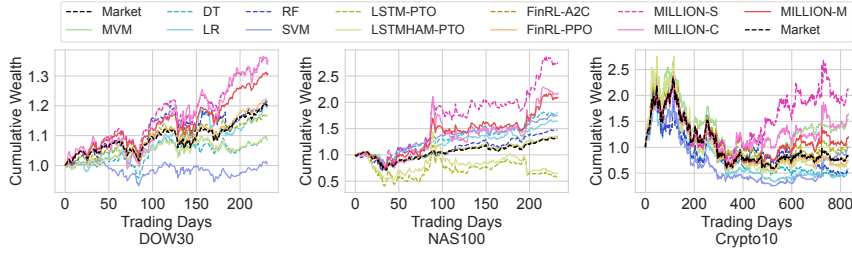
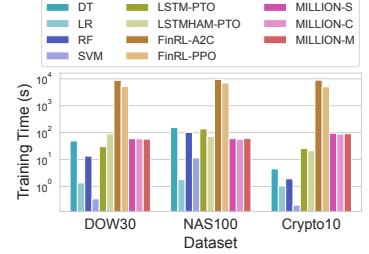Figure 3: Backtest results in terms of CW on three datasets

Figure 4: Training efficiency

levels, while the vertical axis denotes the corresponding backtesting metrics. "Interpolation" and "Improvement" refer to portfolio interpolation (cf. Equation 9) and portfolio improvement (cf. Equation 14) methodologies, respectively. "Improvement-LSTM" and "Improvement-SVM" denote portfolio enhancement utilizing Equation 15, where return rates are forecasted using LSTM and SVM algorithms, respectively.

From Figure 5, it is evident that predict-then-optimize methodologies yield varied outcomes when risk is constrained to specific values across different datasets. Notably, for LR, ASR in the DOW30 dataset demonstrates a gradual increase with risk levels ranging from 1e-5 to 1e-4, while in NAS100, it exhibits an inverse trend. Furthermore, the construction of risk-constrained portfolios may outperform those optimized solely for maximizing the Sharpe ratio, as evidenced by Table 1. For example, in the DOW30 dataset, constraining LR's risk to 5e-5 results in an ASR exceeding 1.5, surpassing the 1.43 ASR achieved through maximizing the Sharpe ratio alone (cf. Table 1). Additionally, our proposed risk control methodologies consistently demonstrate an anticipated pattern: as specified risk increases, both realized ASR and AVOL simultaneously increase. Moreover, it is noteworthy that Improvement consistently outperforms Interpolation, even in scenarios where there are no predicted return rates to guide optimization. Incorporating predicted return rates into portfolio enhancement tends to yield superior performance by guiding the portfolio towards regions with higher return rates.

4.3 Efficiency

In Figure 4, we present the training time required for each algorithm to reach convergence. Traditional machine learning methods, such as DT and RF, are executed on the CPU, while other models are run on an A100 GPU. Notably, MILLION demonstrates comparable training times to predict-then-optimize algorithms and outperforms RL-based methods in terms of speed. This is attributed to the stability of training in DL compared to RL, which typically requires over 100K steps to converge. Additionally, the interaction between RL agents and the environment is slower, further hampering the training efficiency of RL-based methods.

4.4 Case Study

To further understand the effectiveness of the proposed risk control approaches, we conduct a case study on DOW30 dataset.
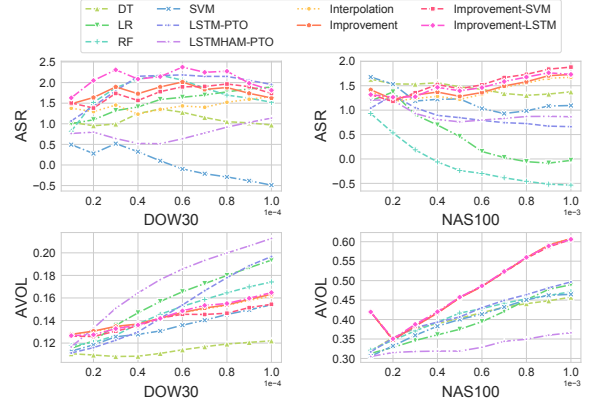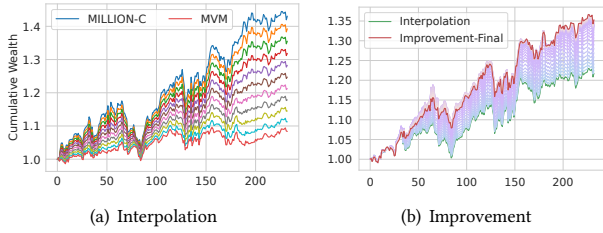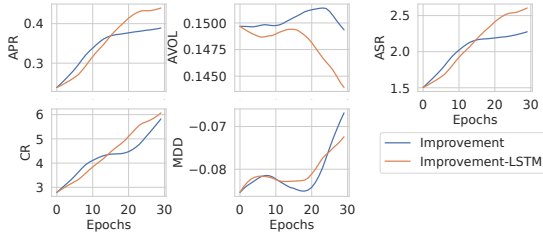

Figure 5: Comparison of the proposed risk control methods with predict-then-optimize approaches

Effect of uniform interpolation. In Figure 6(a), the performance of uniform portfolio interpolation is depicted, showcasing the variation of γ_t from 0 to 1 with a step size of 0.1. The bottom curve represents the cumulative wealth attained from min-variance optimization, where γ_t is fixed at 1 for all t. Notably, as γ_t increases, both CW and AVOL gradually decrease. Furthermore, CW consistently remains bounded by the cumulative wealth obtained from min-variance optimization (MVM), underscoring the empirical validity of Proposition 3.2.

Effect of portfolio improvement. Apart from the portfolio interpolation method, we demonstrate the effectiveness of portfolio improvement in Figure 6(b). Here, we maintain the portfolio risk at 5e-5 and iterate Equation (14) optimization 30 times. Each line in the figure represents a single iteration, with the green line depicting the outcome of portfolio interpolation and the red line representing the final result of portfolio improvement. Notably, all lines exhibit identical risk levels. From this visualization, we observe a notable enhancement in the CW value, rising from approximately 1.2 to 1.35 by the conclusion of the test period, underscoring the effectiveness of the proposed portfolio improvement. Additionally, we examine the performance of portfolio improvement with varying numbers of optimization epochs (refer to Equation 14 and Equation 15), showcased in Figure 7, where each data point corresponds to the same risk level. Incorporating the predicted return rate into the portfolio improvement yields a final portfolio with superior returns and reduced risk, despite potential inaccuracies in the predicted return rate. For instance, the predict-then-optimize framework yields a

Figure 6: The effect of the proposed risk control approaches in terms of CW on DOW30 dataset

Figure 7: The effect of portfolio improvement on metrics

0.1808 APR, as demonstrated in Table 1. This observation underscores the robustness of our proposed portfolio improvement to the accuracy of predicted return rates, owing to the provision of a solid initial portfolio through the return-related maximization model.

5 RELATED WORKS

In this section, we survey the related studies on predict-then-optimize and direct portfolio optimization.

Predict-then-optimize Portfolio Optimization. Mean-variance model [29] is a classical method to construct a portfolio through solving a combinational optimization problem as follows:

$$\max \sum_{i=1}^{N} b_t^i r_t^i \quad \text{s.t.} \quad \boldsymbol{b}_t^T \Sigma \boldsymbol{b}_t \leq \sigma_g, \quad \sum_{i=1}^{N} b^i = 1, \quad 1 \geq b^i \geq 0, \forall_i \quad (16)$$

in which the return rate of each asset, i.e., $\boldsymbol{r}_t$, and its covariance, i.e., Σ_t, are supposed to be already known or simply estimated using the sample mean and sample covariance of historical assets' return rate. However, since the market is dynamic and volatile, the simple estimation may not reflect the future market. To construct a more effective portfolio, plenty of researchers are devoted to developing more powerful return rate prediction models. For example, Li et al. [21] propose Confidence Weighted Mean Reversion (CWMR) to estimate the next price relative as the inverse of the last of it. Huang et al. [16] exploit the reversion phenomenon via robust L_1-median estimators to predict the next price relative. With the fast development of machine learning, a number of advanced models are proposed. For example, Li et al. [22] propose a multimodal event-driven LSTM model using online news to predict stocks' return rates. Besides improving the accuracy of return rate prediction, there are lots of efforts in formulating a different portfolio optimization problem. For example, Rockafellar et al. [31] adopt conditional value-at-risk (CVaR) as the risk metric for portfolio construction. Furthermore, Lai et al. [19] propose a multi-trend CVaR as the risk

metric to optimize the constructed portfolio. Despite the methods in this line that can easily control risk to a user-specified risk level, i.e., σ_g, their effectiveness will be strongly influenced by the estimation of assets' return rate. Unfortunately, the accurate return rate prediction is difficult or even impossible, which may be due to the incorrectness of the model construction or the less predictability of the market states.

Direct Portfolio Optimization. Instead of predicting the return rate of assets, RL-based [23, 27, 33, 36, 39, 42, 43] and DL-based approaches aim to directly output a portfolio for each holding period, which usually can achieve higher investment profits compared with methods in the predict-then-optimize framework. In RL-based methods, they focus on learning a policy to map the market state to an action to maximize the discounted cumulative reward, in which the action is defined as the portfolio weight $\boldsymbol{b}_t$. For example, Liu et al. [24, 25] develop a general deep RL-based framework to enable investors to automate trading, in which investors can flexibly incorporate their prior knowledge. Lien et al. [23] adopt contrastive learning technique and reward smoothing to indicate the stock relationships and maximize a long-term profit, respectively. For the DL-based methods [45, 46], the only difference to RL-based methods is the way of optimization, where RL-based methods optimize the parameters through gradients backward from a surrogate loss while DL-based methods directly optimize the portfolio objective through gradient ascent since this objective is differentiable. Zhang et al. [46] propose to directly optimize Sharpe ratio in different model architectures such as MLP, CNN, and LSTM. Although optimizing Sharpe ratio can control risk to some extent, they are hard to achieve fine-grained risk control and impossible to fit investors' personality in terms of risk preference with only one constructed portfolio. Moreover, the strong ability of representation of neural networks will degenerate the generalization of the trained model to be applied to the future market.

6 CONCLUSION

In this paper, we propose a general multi-objective framework with controllable risk for portfolio management called MILLION, in which we decompose the portfolio management into two main phases, i.e., return-related maximization and risk control. In the first phase, we follow the DL-based portfolio framework and demonstrate that the multi-objective design is useful in improving return-related metrics, such as APR, and ASR, through backtesting on three real-world datasets. In the risk control phase, we propose two approaches to adjust the risk of the constructed portfolio to fit different investors' preferences in terms of risk-taking. Compared with methods in the predict-then-optimize framework, MILLION performs better in terms of ASR under the same risk level.

REFERENCES

[1] Zeng-Liang Bai, Ya-Ning Zhao, Zhigang Zhou, Wen-Qin Li, Yang Gao, Ying Tang, Long-Zheng Dai, and Yi-You Dong. 2023. Mercury: A Deep Reinforcement Learning-Based Investment Portfolio Strategy for Risk-Return Balance. *IEEE Access* 11 (2023), 78353–78362.

[2] L. Breiman. 2001. Random Forests. *Machine Learning* 45 (2001), 5–32.

[3] L. Breiman and Richard A. Olshen. 2017. Points of Significance: Classification and regression trees. *Nature Methods* 14 (2017), 757–758.

[4] W. Chen, Haoyu Zhang, Mukesh Kumar Mehlawat, and Lifen Jia. 2021. Mean-variance portfolio optimization using machine learning-based stock price prediction. *Appl. Soft Comput.* (2021).

[5] Tuoyuan Cheng and Kan Chen. 2023. A General Framework for Portfolio Construction Based on Generative Models of Asset Returns. *The Journal of Finance and Data Science* (2023).

[6] Liwei Deng, Yan Zhao, Jin Chen, Shuncheng Liu, Yuyang Xia, and Kai Zheng. 2024. Learning to Hash for Trajectory Similarity Computation and Search. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 4491–4503.

[7] Shumin Deng, Ningyu Zhang, Wen Zhang, Jiaoyan Chen, Jeff Z. Pan, and Huajun Chen. 2019. Knowledge-Driven Stock Trend Prediction and Explanation via Temporal Convolutional Network. *Companion Proceedings of The 2019 World Wide Web Conference* (2019).

[8] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2015. Deep Learning for Event-Driven Stock Prediction. In *International Joint Conference on Artificial Intelligence*.

[9] Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. 2018. Enhancing Stock Movement Prediction with Adversarial Training. In *International Joint Conference on Artificial Intelligence*.

[10] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2018. Temporal Relational Ranking for Stock Prediction. *ACM Transactions on Information Systems (TOIS)* 37 (2018), 1 – 30.

[11] Jianliang Gao, Xiaoting Ying, Cong Xu, Jianxin Wang, Shichao Zhang, and Zhao Li. 2021. Graph-Based Stock Recommendation by Time-Aware Relational Attention Network. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 16 (2021), 1 – 21.

[12] Haoyu Geng, Hang Ruan, Runzhong Wang, Yang Li, Yang Wang, Lei Chen, and Junchi Yan. 2023. Rethinking and Benchmarking Predict-then-Optimize Paradigm for Combinatorial Optimization Problems. *ArXiv* (2023).

[13] Li Han, Nan Ding, Guoxuan Wang, Da Zhao Cheng, and Yuqi Liang. 2023. Efficient Continuous Space Policy Optimization for High-frequency Trading. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (2023).

[14] Zheng Hao, Haowei Zhang, and Yipu Zhang. 2023. Stock Portfolio Management by Using Fuzzy Ensemble Deep Reinforcement Learning Algorithm. *Journal of Risk and Financial Management* (2023).

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9 (1997), 1735–1780.

[16] Dingjiang Huang, Junlong Zhou, Bin Li, Steven C. H. Hoi, and Shuigeng Zhou. 2016. Robust Median Reversion Strategy for Online Portfolio Selection. *TKDE* 28, 9 (2016), 2480–2493.

[17] Junkyu Jang and Nohyoon Seong. 2023. Deep reinforcement learning for stock portfolio optimization by connecting with modern portfolio theory. *Expert Syst. Appl.* 218 (2023), 119556.

[18] Alex Kendall, Yarin Gal, and Roberto Cipolla. 2017. Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2017), 7482–7491.

[19] Zhaoyang Lai, Cheng Li, Xiaotian Wu, Quanlong Guan, and Liangda Fang. [n.d.]. Multitrend Conditional Value at Risk for Portfolio Optimization. *IEEE transactions on neural networks and learning systems* ([n. d.]).

[20] Bin Li, Steven C. H. Hoi, Doyen Sahoo, and Zhiyong Liu. 2015. Moving average reversion strategy for on-line portfolio selection. *Artif. Intell.* 222 (2015), 104–123.

[21] Bin Li, Steven C. H. Hoi, Peilin Zhao, and Vivekanand Gopalkrishnan. 2011. Confidence Weighted Mean Reversion Strategy for Online Portfolio Selection. *ACM Trans. Knowl. Discov. Data* (2011).

[22] Qing Li, Jinghua Tan, Jun Wang, and Hsinchun Chen. 2021. A Multimodal Event-Driven LSTM Model for Stock Prediction Using Online News. *IEEE Transactions on Knowledge and Data Engineering* 33 (2021), 3323–3337.

[23] Yun-Hsuan Lien, Yuan kui Li, and Yu-Shuen Wang. 2023. Contrastive Learning and Reward Smoothing for Deep Portfolio Management. In *International Joint Conference on Artificial Intelligence*.

[24] Xiao-Yang Liu, Ziyi Xia, Jingyang Rui, Jiechao Gao, Hongyang Yang, Ming Zhu, Christina Dan Wang, Zhaoran Wang, and Jian Guo. 2022. FinRL-Meta: Market Environments and Benchmarks for Data-Driven Financial Reinforcement Learning. In *NeurIPS*.

[25] Xiao-Yang Liu, Hongyang Yang, Jiechao Gao, and Chris Wang. 2021. FinRL: deep reinforcement learning framework to automate trading in quantitative finance. *Proceedings of the Second ACM International Conference on AI in Finance* (2021).

[26] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2024. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations*.

[27] Yang Liu, Qi Liu, Hongke Zhao, Zhen Pan, and Chuanren Liu. 2020. Adaptive Quantitative Trading: An Imitative Deep Reinforcement Learning Approach. In *AAAI Conference on Artificial Intelligence*.

[28] Yu Ma, Rui Mao, Qika Lin, Peng Wu, and E. Cambria. 2022. Multi-source aggregated classification for stock price movement prediction. *Inf. Fusion* 91 (2022), 515–528.

[29] Harry M. Markowitz. 1952. Portfolio Selection. *The Journal of Finance*.

[30] John Platt. 1999. Probabilistic Outputs for Support vector Machines and Comparisons to Regularized Likelihood Methods.

[31] R. Tyrrell Rockafellar and Stanislav Uryasev. 2000. Optimization of conditional value-at risk. *Journal of Risk* (2000).

[32] William F. Sharpe. 1994. The Sharpe Ratio.

[33] Shuo Sun, Xu He, R. Wang, Junlei Zhu, Jian Li, and Bo An. 2021. DeepScalper: A Risk-Aware Reinforcement Learning Framework to Capture Fleeting Intraday Trading Opportunities. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (2021).

[34] Heyuan Wang, Tengjiao Wang, Shun Li, Jiayi Zheng, Shijie Guan, and Wei Chen. 2022. Adaptive Long-Short Pattern Transformer for Stock Investment Selection. In *International Joint Conference on Artificial Intelligence*.

[35] Jingyuan Wang, Yang Zhang, Ke Tang, Junjie Wu, and Zhang Xiong. 2019. AlphaStock: A Buying-Winners-and-Selling-Losers Investment Strategy using Interpretable Deep Reinforcement Attention Networks. *SIGKDD* (2019).

[36] R. Wang, Hongxin Wei, Bo An, Zhouyan Feng, and Jun Yao. 2020. Commission Fee is not Enough: A Hierarchical Reinforced Framework for Portfolio Management. In *AAAI Conference on Artificial Intelligence*.

[37] Tianfu Wang, Liwei Deng, Chao Wang, Jianxun Lian, Yue Yan, Nicholas Jing Yuan, Qi Zhang, and Hui Xiong. 2024. COMET: NFT Price Prediction with Wallet Profiling. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*. 5893–5904.

[38] Tianfu Wang, Li Shen, Qilin Fan, Tong Xu, Tongliang Liu, and Hui Xiong. 2024. Joint Admission Control and Resource Allocation of Virtual Network Embedding Via Hierarchical Deep Reinforcement Learning. *IEEE Transactions on Services Computing* 17, 03 (2024), 1001–1015.

[39] Zhicheng Wang, Biwei Huang, Shikui Tu, Kun Zhang, and Lei Xu. 2021. Deep-Trader: A Deep Reinforcement Learning Approach for Risk-Return Balanced Portfolio Management with Market Conditions Embedding. In *AAAI Conference on Artificial Intelligence*.

[40] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems* 34 (2021), 22419–22430.

[41] MuEn Wu, JiaHao Syu, Jerry Chunwei Lin, and JanMing Ho. 2021. Portfolio management system in equity market neutral using reinforcement learning. *Applied Intelligence* 51, 8119 – 8131.

[42] Ke Xu, Yifan Zhang, Deheng Ye, Peilin Zhao, and Mingkui Tan. 2020. Relation-Aware Transformer for Portfolio Policy Learning. In *International Joint Conference on Artificial Intelligence*.

[43] Mengyuan Yang, Xiaolin Zheng, Qianqiao Liang, Bing Han, and Mengying Zhu. 2022. A Smart Trader for Portfolio Management based on Normalizing Flows. In *International Joint Conference on Artificial Intelligence*.

[44] Yunan Ye, Hengzhi Pei, Boxin Wang, PinYu Chen, Yada Zhu, Jun Xiao, and Bo Li. 2020. Reinforcement-Learning based Portfolio Management with Augmented Asset Movement Prediction States. *ArXiv*.

[45] Chao Zhang, Zihao Zhang, Mihai Cucuringu, and Stefan Zohren. 2021. A Universal End-to-End Approach to Portfolio Optimization via Deep Learning.

[46] Zihao Zhang, Stefan Zohren, and Stephen J. Roberts. 2020. Deep Learning for Portfolio Optimization. In *The Journal of Financial Data Science*.

[47] Zetao Zheng, Jie Shao, Jia Zhu, and Heng Tao Shen. 2023. Relational Temporal Graph Convolutional Networks for Ranking-Based Stock Prediction. *2023 IEEE 39th International Conference on Data Engineering (ICDE)* (2023), 123–136.

[48] Dawei Zhou, Lecheng Zheng, Yada Zhu, Jianbo Li, and Jingrui He. 2020. Domain Adaptive Multi-Modality Neural Attention Network for Financial Forecasting. *Proceedings of The Web Conference 2020* (2020).

[49] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 11106–11115.

[50] Mengying Zhu, Yan Wang, Fei Wu, Mengyuan Yang, Cheng Chen, Qianqiao Liang, and Xiaolin Zheng. 2022. WISE: Wavelet based Interpretable Stock Embedding for Risk-Averse Portfolio Management. *Companion Proceedings of the Web Conference 2022* (2022).

[51] Eric Zivot. 2017. *Introduction to Computational Finance and Financial Econometrics*. Chapman & Hall Crs.