Transformer-Based Reinforcement Learning for Portfolio Optimization

Ramiro Javier Valdes Jara
Department of Industrial and Systems Engineering
University of Miami
Coral Gables, FL, USA
rjv71@miami.edu

***Abstract*—We propose T-RLPM, a transformer-based reinforcement learning (RL) framework for dynamic financial portfolio optimization. While recent RL approaches like EarnMore already support customizable stock pools through masking and self-supervised representation learning, they typically rely on shallow MLP architectures with limited capacity to model complex temporal and cross-asset relationships.**

T-RLPM builds directly on EarnMore by introducing transformer-style encoders for both actor and critic networks, enabling richer representations of spatiotemporal financial patterns. Our method retains one-shot training capabilities on a global stock pool and supports masked asset handling, but leverages the expressive power of transformers to better capture asset dependencies.

Empirical results on the Dow Jones Industrial Average (DJIA) dataset show that T-RLPM achieves a 59.87% annual return and outperforms prior methods in the Global Stock Pool (GSP) setting. However, the model underperforms in Customizable Stock Pools (CSPs), indicating challenges in generalization and training stability under masked conditions. The source code for T-RLPM is available for reproducibility https://github.com/ramirovaldesjara/T-RLPM.

I. Introduction

Portfolio management (PM) is a crucial aspect of financial trading aimed at optimally allocating and periodically reallocating capital across various financial assets to achieve maximum returns with balanced risks. Traditional approaches, such as mean-variance optimization and capital asset pricing models, often miss to capture the complex, nonlinear interactions between financial assets, particularly under rapidly evolving market conditions [1]. Reinforcement learning (RL) has recently emerged as a powerful framework for addressing these shortcomings by modeling adaptive sequential decision-making processes, capable of dynamically responding to changing market environments [2], [3].

Deep reinforcement learning (DRL) methods, which integrate deep neural networks with traditional RL algorithms, have significantly advanced portfolio management strategies by learning effective asset allocation policies directly from interactions with financial markets. A notable example is DeepTrader [3], which utilizes a two-module system combining graph-attention networks for asset interactions and market-condition embeddings to dynamically balance risk and returns. Such models have demonstrated remarkable robustness and superior performance compared to classical methods.

Nevertheless, conventional RL models typically assume fixed stock pools, limiting their flexibility and practical applicability since investors frequently alter their portfolios based on evolving market conditions or individual preferences. Addressing this issue, the EarnMore framework [4] introduces maskable stock representations to handle customizable stock pools (CSPs) through a one-shot training process on a global stock pool. EarnMore leverages self-supervised mask-and-reconstruct learning to derive meaningful stock representations, enabling efficient adaptability to varying investor-specific asset selections without the computational cost of retraining. This strategy significantly improves profitability and adaptability, outperforming numerous prior methods.

Building upon previous developments, our research integrates the pool adaptability of EarnMore with advanced transformer-based mechanisms. Specifically, we propose a unified RL framework, T-RLPM, that combines dynamic asset masking, adaptive representation learning, and transformer-driven decision-making, thereby enhancing overall performance in dynamic market conditions.

The remainder of the document is structured as follows. Section II provides an overview of related RL-based portfolio management approaches. Section IV details the architecture and mechanisms of the proposed framework. Section III formally defines the portfolio optimization problem. Experimental results are presented in Section V, showcasing the superior performance of our proposed approach. Lastly, Section VI summarizes our contributions and outlines potential directions for future research.

II. Related Work

Portfolio management is a critical area within financial investment, entailing strategic asset allocation to maximize returns while simultaneously mitigating risks. Traditional rule-based methodologies primarily fall into two categories: mean reversion [5] and momentum strategies [6]. Mean reversion strategies involve purchasing undervalued stocks and selling overvalued ones, relying on the assumption that asset prices will revert to their historical averages. Conversely, momentum strategies anticipate that past asset performance trends will persist in the future. Two prominent momentum-based approaches include Cross-Sectional Momentum and Time-Series Momentum [7]. However, these rule-based methods often struggle to adapt to rapidly evolving market dynamics and typically perform optimally only under specific market conditions [8].

In recent years, prediction-based approaches leveraging advanced machine learning techniques have substantially improved portfolio management outcomes compared to traditional methods. These techniques approach portfolio management as supervised learning tasks, either predicting future asset returns through regression or forecasting price movement directions via classification, subsequently utilizing heuristic strategies to determine asset allocations based on predictive outcomes. Prediction-based methodologies generally fall into two groups: classical machine learning models such as XGBoost [9] and LightGBM [10], and deep learning architectures including Attention-based Long Short-Term Memory networks (ALSTM) [11] and Temporal Convolutional Networks (TCN) [12]. Nevertheless, the inherent volatility, noise, and unpredictability of financial markets significantly challenge the accuracy of such predictive models. Moreover, translating predictive signals into actionable trading strategies remains inherently challenging, often resulting in suboptimal financial performance [13].

More recently, reinforcement learning (RL) techniques have gained prominence in portfolio management due to their capacity for sequential decision-making in uncertain and dynamic environments. Examples include Ensemble of Identical Independent Evaluators (EIIE), which utilizes convolutional neural networks (CNN) for feature extraction combined with RL for asset allocation [2]. The Investor-Imitator framework introduces imitation learning within RL to emulate professional investor behaviors, enhancing decision-making reliability [14]. State-Augmented Reinforcement Learning (SARL) incorporates price predictions into the RL state representation to improve decision-making capabilities, utilizing deterministic policy gradient methods [15]. DeepTrader further advances this field by dynamically balancing risk-return profiles, employing unique graph structures for portfolio generation [3]. Hierarchical Reinforcement Portfolio Management (HRPM) integrates hierarchical decision-making processes aimed at maximizing long-term profits while incorporating transaction costs, such as price slippage, into the trading strategy [16]. Lastly, DeepScalper merges intraday trading strategies with risk-aware considerations to exploit short-term investment opportunities effectively [17].

Despite significant progress, existing RL-based methods typically address fixed asset pools, limiting their practical applicability. Recent advancements like EarnMore have specifically targeted customizable stock pools by employing maskable stock representations and self-supervised learning techniques to enhance adaptability and robustness in portfolio management [4]. Our work builds upon these developments, integrating sophisticated attention mechanisms to further enhance decision-making precision, flexibility, and overall performance under diverse market conditions.

A. Attention and Transformers in Finance

Attention mechanisms and transformer architectures have revolutionized the field of deep learning by providing highly efficient ways to model long-range dependencies in sequential data [18]. These models initially demonstrated extraordinary success in natural language processing tasks, notably in language translation and text generation. The inherent capacity of transformers to capture intricate relationships over varying temporal scales and their interpretability have driven researchers to explore their applicability to financial domains, particularly in portfolio optimization, asset pricing, and risk management.

The financial markets are characterized by complex interactions across different assets and temporal patterns. Traditional sequence models such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks have shown effectiveness to some extent, but often suffer from limitations such as difficulties in modeling very long sequences, susceptibility to vanishing gradients, and inefficiencies in parallel computation [12]. Attention mechanisms address these challenges by allowing models to dynamically weigh information from different parts of a sequence, enabling better learning of both long-term dependencies and cross-sectional interactions among assets [11].

Several studies have adapted attention-based architectures to financial prediction tasks. For instance, Qin et al. [11] proposed dual-stage attention-based recurrent networks that significantly improved forecasting accuracy on financial time series. Meanwhile, attention mechanisms have also been employed effectively in capturing trading signals from heterogeneous data sources, including market indicators and textual data.

More recently, transformer-based models have gained significant popularity within financial applications due to their capability for modeling sequential data with varying intervals and irregular patterns, common in financial markets. Li and Tam [19] further enhanced portfolio management through an attention-based ensemble framework, highlighting the importance of attention in emphasizing critical features across diverse market scenarios.

Despite their potential, transformer-based approaches have not been integrated into reinforcement learning frameworks for portfolio optimization. To bridge this gap, our work leverages transformer-based encoders for both actor and critic networks within an RL-based portfolio management framework. We introduce explicit positional embeddings and a learnable cash token mechanism to better handle dynamic stock pools.

III. Problem Formulation

We model the task of portfolio management as a Markov Decision Process (MDP), denoted by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$, where:

- $\mathcal{S}$ is the set of environment states.
- $\mathcal{A}$ is the set of available actions (portfolio allocations).
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function, defining the probability of transitioning between states.

- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function that evaluates the immediate return.
- $\gamma \in [0, 1)$ is the discount factor.

A. State Representation s_t

At each time step t, the environment provides a comprehensive view of the financial market through a multi-dimensional state tensor: $X_t = [x_{t-D+1}, x_{t-D+2}, \ldots, x_t] \in \mathbb{R}^{N \times D \times F}$ where N is the number of assets in the Global Stock Pool (GSP), D is the length of the historical lookback window (e.g., past 30 days) and F is the number of features per asset per time step, such as OHLCV prices, technical indicators, and temporal embeddings.

To support personalized preferences, we define a Customizable Stock Pool (CSP) as a subset of the GSP for each investor. Let N^* denote the number of excluded stocks. These stocks are substituted with a learnable masked token $[M]$, resulting in a padded and fixed-dimension input for the agent $s_t = [x_t^{(1)}, \ldots, x_t^{(N-N^*)}, [M], \ldots, [M]] \in \mathbb{R}^{N \times D \times F}$.

B. Action Representation a_t

The action a_t corresponds to the portfolio weight vector at time step t, which determines the proportion of capital allocated to each asset and cash $a_t = W_t = [w_0, w_1, \ldots, w_{N-N^*}, w_{[M]}^{(1)}, \ldots, w_{[M]}^{(N^*)}] \in \mathbb{R}^{N+1}$, where w_0 is the proportion of portfolio held as cash (risk-free asset), w_i represents the allocation weight for each asset in CSP and $w_{[M]}^{(j)}$, the allocation for each masked stock. All weights must sum to one to ensure the portfolio is fully invested $\sum_{i=0}^{N} w_i = 1$.

C. Reward Function r_t

The reward at time t is defined as the increment in the total portfolio value after reallocation $r_t = V_t - V_{t-1}$. The portfolio value V_t is computed based on the weighted sum of asset returns:

$$V_t = w_0 V_{t-1} + (1 - w_0) V_{t-1} \left(1 + \sum_{i=1}^{N} w_i \cdot \frac{p_{i,t}^c - p_{i,t-1}^c}{p_{i,t-1}^c} \right)$$

where $p_{i,t}^c$ is the closing price of asset i at time t.

IV. Methodology

We propose a transformer-based reinforcement learning framework that builds on the previous literature EarnMore [4], while overcoming their limitations. Existing framework, use shallow MLP architectures and inter-dependent actor-critic designs. While efficient, these approaches fall short on modelling complex market dynamics. T-RLPM introduces the following key enhancements:

- **Transformer-Based Encoders:** We use multi-head attention encoders to capture richer temporal and cross-asset relationships.
- **Decoupled Actor and Critic Networks:** Separate attention-based encoders are used for the actor and critic to improve task specialization.

A. Maskable Stock Representation for CSPs

To support the one-shot training paradigm for customizable stock pools (CSPs), we use from previous literature the maskable stock representation mechanism. This approach ensures that each investor's subset of assets (CSP) can be processed within the same model trained on the full global stock pool (GSP).

Given the global pool $\mathcal{U}$ with N assets, a CSP $\mathcal{C}_t$ is generated at time t by masking N^* stocks. The resulting masked input includes $[M]$ tokens to maintain fixed input dimensionality. We construct this representation in two stages

1) **Stock-Level Encoding:** Raw price and technical features for each stock are embedded using 1D convolutions and temporal encodings.
2) **Pool-Level Representation:** Masked tokens replace masked stocks and the entire sequence is passed through a transformer encoder, preserving latent relationships between stocks.

By simulating various masking patterns during training, the model learns to generalize across different CSPs while retaining the full GSP structure.

B. Reinforcement Learning Optimization with Masked CSPs

Our reinforcement learning algorithm is based on Soft Actor-Critic (SAC) from previous literature (CITE). Both the actor and critic networks are constructed using attention-based encoders, which receive masked stock embeddings and are trained jointly with a dynamically evolving policy.

Let $\rho(s_t, m)$ denote the masked stock representation used as input state, where m encodes investor-specific masking over the global stock pool. We abbreviate $\rho(s_t, m)$ as ρ_t for clarity. The policy $\pi_\phi(a_t|\rho_t)$ and Q-function $Q_\theta(\rho_t, a_t)$ are jointly optimized using the following objectives.

The critic network is optimized to minimize the Bellman residual

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t)\sim\mathcal{D}} \Big[\frac{1}{2} \big(Q_\theta(\rho_t, a_t) - \big(r(s_t, a_t) + \gamma \, \mathbb{E}_{s_{t+1}\sim p} \left[V_{\bar{\theta}}(\rho_{t+1}) \right] \big) \big)^2 \Big]$$

where the soft value function is defined as

$$V_{\bar{\theta}}(\rho_t) = \mathbb{E}_{a_t\sim\pi_\phi} \left[Q_{\bar{\theta}}(\rho_t, a_t) - \alpha \log \pi_\phi(a_t|\rho_t) \right]$$

Here, $\bar{\theta}$ represents the parameters of the target Q-network updated via an exponential moving average.

The actor network is trained using the reparameterization trick. Let $\epsilon_t \sim \mathcal{N}(0, I)$ be a standard Gaussian sample, and define the action as $a_t = f_\phi(\epsilon_t; \rho_t)$. The actor loss is given by

$$J_\pi(\phi) = \mathbb{E}_{s_t\sim\mathcal{D},\epsilon_t\sim\mathcal{N}} \left[\alpha \log \pi_\phi(f_\phi(\epsilon_t; \rho_t)|\rho_t) - Q_\theta(\rho_t, f_\phi(\epsilon_t; \rho_t)) \right]$$

The entropy regularization coefficient α is automatically adjusted by minimizing

$$J(\alpha) = \mathbb{E}_{a_t\sim\pi_t} \left[-\alpha \log \pi_t(a_t|\rho_t) - \alpha \bar{H} \right]$$

where $\bar{H}$ is the target entropy, treated as a tunable hyperparameter.

To prevent allocation to masked stocks, we use the following penalty strategy. A penalty IS added to the TD error if any masked asset receives nonzero allocation.

The maskable stock representation is trained using a reconstruction loss computed only on the masked assets

$$J(\theta_e, \theta_c, \theta_{enc}, \theta_{dec}) = \frac{1}{N^*} \sum_{i=1}^{N^*} (p_{i,t} - \tilde{p}_{i,t})^2$$

where N^* is the number of masked stocks, $p_{i,t}$ the true price, and $\tilde{p}_{i,t}$ the reconstructed output. This objective is optimized simultaneously with actor-critic training but not via a unified weighted loss to avoid destabilizing the sampling distribution.

Our training loop sequentially updates the critic, entropy coefficient, actor, and masked representation module using mini-batches sampled from a replay buffer.

C. Attention-Based Actor and Critic Networks

Our architecture introduces transformer-style attention mechanisms for both the actor and critic networks.

1) Actor Network: The actor network is responsible for generating portfolio allocation weights a_t given a masked stock representation s_t. The input tensor is first projected into a latent embedding space via a linear transformation, followed by the addition of positional encodings. A learnable cash token `[CLS]` is used, prepended to the sequence, yielding an augmented input tensor of shape $(N + 1) \times d$.

Formally, the embedded input is

$$Z_t = \text{Transformer}([\texttt{[CLS]}] \oplus (E s_t + P))$$

where $E \in \mathbb{R}^{F \times d}$ is the embedding matrix, $P \in \mathbb{R}^{N \times d}$ denotes the learnable positional embeddings, and $Z_t \in \mathbb{R}^{(N+1) \times d}$ is the output of the transformer encoder.

The output Z_t is then linearly mapped to a pair of vectors representing the mean and log standard deviation of a Gaussian distribution $(\mu_t, \log \sigma_t) = W Z_t + b, \quad W \in \mathbb{R}^{d \times 2}, \quad b \in \mathbb{R}^2$.

Portfolio scores are sampled using the reparameterization trick $\hat{a}_t \sim \mathcal{N}(\mu_t, \sigma_t^2), \quad \sigma_t = \exp(\log \sigma_t)$.

To promote sparse and interpretable allocations, a log-ranking softmax is applied

$$\alpha_{t,i} = \hat{a}_{t,i} \cdot \log(i + 1)$$

$$a_{t,i} = \frac{\exp(\alpha_{t,i})}{\sum_{j=1}^{N+1} \exp(\alpha_{t,j})}$$

The resulting vector $a_t \in \mathbb{R}^{N+1}$ represents normalized portfolio weights over the asset pool, with $a_{t,0}$ corresponding to the allocation assigned to the cash token.

2) Critic Network: The critic network estimates the value of a state-action pair $Q(s_t, a_t)$ by integrating masked state representations with projected action embeddings. The input state tensor s_t is first embedded into a latent space using a linear transformation. Simultaneously, the action vector a_t is projected into the same latent space via a learned linear mapping $A \in \mathbb{R}^{1 \times d}$, resulting in $A(a_t) \in \mathbb{R}^{(N+1) \times d}$.

The embedded state features and projected action embeddings are combined additively

$$Z_t = E(s_t) + A(a_t)$$

A learnable cash token `[CLS]` is prepended, and positional encodings $P \in \mathbb{R}^{(N+1) \times d}$ are added to the sequence. The resulting input is passed through a transformer encoder

$$H_t = \text{Transformer}([\texttt{[CLS]}] \oplus (Z_t + P))$$

The output $H_t \in \mathbb{R}^{(N+1) \times d}$ is aggregated via average pooling $h_t = \frac{1}{N+1} \sum_{i=1}^{N+1} H_{t,i}$. Finally, a feedforward head outputs a pair of Q-value estimates $Q_1, Q_2 \in \mathbb{R}$, used in twin Q-learning: $(Q_1, Q_2) = W h_t + b, \quad W \in \mathbb{R}^{d \times 2}, \quad b \in \mathbb{R}^2$. These values are used to estimate the value of the policy and to update both the critic and the actor using soft actor-critic (SAC).

V. Numerical Experiments

We evaluate the performance of our T-RLPM framework through empirical experiments on real-world financial data. Specifically, we use the Dow Jones Industrial Average (DJIA) dataset, which consists of daily stock price observations for 28 assets. The evaluation follows standard portfolio management benchmarks, and we compare our method against several baselines.

A. Experimental Setup

We divide the dataset into a training set and a non-overlapping testing set. All models are trained using the same historical lookback window and evaluated on unseen market data. Our training period spans multiple market conditions to ensure robustness and generalizability.

TABLE II
Dataset and Date Splits for DJ30

Dataset	**GSP**	**CSP1**	**CSP2**	**CSP3**
Stocks	28	10	7	10
Train Dates		2007-09-26	2021-01-07	
Test Dates		2021-01-07	2022-06-26	

B. Evaluation Metrics

To evaluate the performance of our portfolio management strategies, we adopt a set of standard risk-return metrics used in the financial literature. These metrics provide a view of both profitability and risk exposure.

The *Annual Rate of Return (ARR)* captures the average compounded return achieved per year. It is computed as

$$\text{ARR} = \frac{V_T - V_0}{V_0} \times \frac{C}{T}$$

where V_0 and V_T denote the initial and final portfolio values, T is the number of trading days in the evaluation period, and $C = 252$ represents the typical number of trading days in a year.

TABLE I

PERFORMANCE COMPARISON ON DJ30 WITH GLOBAL STOCK POOL. RESULTS IN RED, YELLOW AND GREEN SHOW THE BEST, SECOND BEST AND THIRD BEST RESULTS ON EACH DATASET.

Categories	Strategies	ARR% ↑	SR ↑	CR ↑	SOR ↑	MDD% ↓	VOL ↓
Rule-based	Market	6.710	0.458	0.776	15.560	22.200	0.013
	BLSW	7.610	0.512	0.857	16.930	21.540	0.012
	CSM	5.930	0.400	0.643	12.950	20.770	0.012
ML-based	XGBoost	10.260	0.343	0.599	10.420	14.760	0.013
	LightGBM	13.420	0.591	0.703	14.220	20.900	0.014
DL-based	ALSTM	15.030	1.186	0.590	14.890	28.070	0.013
	TCN	6.980	0.732	0.269	8.280	37.400	0.018
RL-based	PG	7.970	0.321	0.435	8.430	21.570	0.012
	PPO	9.240	0.385	0.512	10.140	20.810	0.012
	SAC	9.150	0.326	0.448	8.830	20.600	0.012
	EIIE	22.900	0.689	1.465	23.450	16.770	0.014
	SARL	21.920	0.786	1.109	23.020	20.400	0.012
	IMIT	27.640	0.909	1.593	27.380	20.050	0.014
	DeepTrader	32.230	1.335	1.440	27.110	21.190	0.013
	EarnMore	47.290	1.454	1.692	28.040	21.650	0.018
	T-RLPM	59.870	1.760	1.678	27.31	29.09	0.020
Improvement over SOTA		26.60%	21.04%	-0.83%	-2.67%	-	-

The *Sharpe Ratio (SR)* measures the risk-adjusted return by dividing the expected return by the standard deviation of returns. It is defined as

$$\text{SR} = \frac{\mathbb{E}[r]}{\sigma[r]}$$

where the return sequence r is given by

$$r = \left[\frac{V_1 - V_0}{V_0}, \frac{V_2 - V_1}{V_1}, \ldots, \frac{V_T - V_{T-1}}{V_{T-1}}\right]$$

The *Volatility (VOL)* corresponds to the standard deviation of daily returns and is simply expressed as

$$\text{VOL} = \sigma[r]$$

To assess downside risk, we include the *Maximum Drawdown (MDD)*, which quantifies the largest observed drop from a historical peak. Let

$$R_i = \prod_{j=1}^{i} \frac{V_j}{V_{j-1}}, \quad P_i = \max_{1 \le j \le i} R_j$$

Then the MDD is computed as

$$\text{MDD} = \max_i \left(\frac{P_i - R_i}{P_i}\right)$$

The *Calmar Ratio (CR)* is another risk-adjusted return metric that relates the expected return to the maximum drawdown

$$\text{CR} = \frac{\mathbb{E}[r]}{\text{MDD}}$$

Finally, the *Sortino Ratio (SoR)* offers a variation of the Sharpe Ratio that focuses specifically on downside risk. It uses the standard deviation of negative returns r_{neg} instead of total volatility

$$\text{SoR} = \frac{\mathbb{E}[r]}{\sigma[r_{\text{neg}}]}$$

where r_{neg} includes only the elements of r that are less than zero.

C. Results

We begin by evaluating the performance of the proposed T-RLPM model on the Global Stock Pool (GSP), which includes all assets in the Dow Jones Industrial Average (DJIA). As reported in Table I, T-RLPM achieves the highest Annualized Return of 59.87%, substantially outperforming all baseline methods. Additionally, it obtains the highest Sharpe Ratio (1.76), indicating strong risk-adjusted returns. These results confirm that transformer-based attention mechanisms enable more expressive temporal and cross-asset modeling compared to traditional MLP-based designs.

TABLE III

PERFORMANCE COMPARISON ON DJ30 ACROSS DIFFERENT CSP POOLS.

Pool	**Strategies**	**ARR% ↑**	**SR ↑**
CSP1	SARL	24.140	0.638
	IMIT	20.071	0.920
	DeepTrader	27.740	0.757
	EarnMore	53.990	1.810
	T-RLPM	23.93	0.943
CSP2	SARL	20.020	0.820
	IMIT	11.841	0.751
	DeepTrader	38.470	0.955
	EarnMore	43.400	1.549
	T-RLPM	6.49	0.430
CSP3	SARL	10.910	0.480
	IMIT	6.851	0.496
	DeepTrader	16.840	0.601
	EarnMore	43.460	1.572
	T-RLPM	-3.92	0.041

To assess generalization, we evaluate T-RLPM on three Customizable Stock Pools (CSPs), simulating realistic investor scenarios with personalized subsets of assets. As shown in Table III, T-RLPM maintains strong performance in CSP1, achieving an ARR of 23.93% and a leading Sharpe Ratio of 0.943. However, its performance deteriorates significantly in CSP2 and CSP3, where it achieves an ARR of only 6.49% and -3.92%, respectively. In contrast, the EarnMore baseline exhibits consistently strong results across all CSPs.

These results suggest that although T-RLPM is highly effective when operating on the full stock universe, its ability to generalize to unseen or smaller asset pools is limited. Notably, our implementation did not achieve meaningful learning in the masked CSP setting. Despite the theoretical benefits of transformer architectures for handling dynamic inputs, the model struggled to adapt when stock selections were customized, and its performance was highly sensitive to asset composition.

We hypothesize that this issue comes from insufficient learning dynamics in masked settings, potentially caused by suboptimal training hyperparameters. Reassessing the design of the masking strategy, penalty functions, and actor supervision may lead to more robust performance under CSPs.

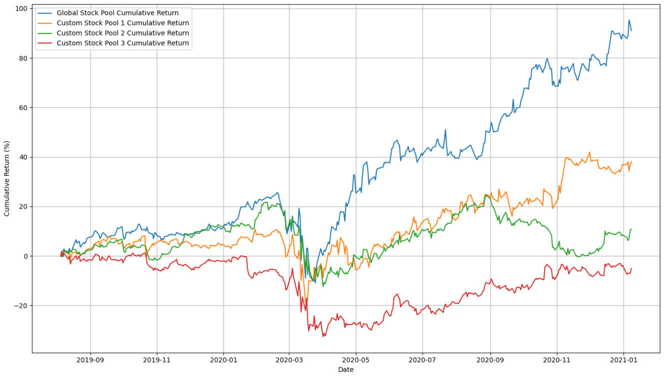

Fig. 1. Cumulative return curves for the Global Stock Pool and Custom Stock Pools 1–3, as generated by the T-RLPM agent during the test period.

The plot in Figure 1 illustrates the cumulative return trajectories over time . The T-RLPM model demonstrates strong performance when operating on the full stock universe (GSP), achieving a final cumulative return close to 100%. However, performance significantly degrades across CSPs, particularly CSP3, which ends the test period with negative overall return. This result reinforces the numerical findings in Table III, where T-RLPM exhibits the highest ARR and Sharpe Ratio in GSP but fails to generalize effectively in CSP2 and CSP3.

In conclusion, the experimental results clearly demonstrate that T-RLPM benefits from transformer-based encoders, outperforming prior methods in GSP settings. However, learning in customizable stock pools did not occur effectively, highlighting the need for further refinement of the masked reinforcement learning framework.

VI. Future Work

In future research, we aim to extend our framework by incorporating multi-agent decision-making for portfolio management and exploring more advanced attention mechanisms.

References

[1] J. Moody, L. Wu, Y. Liao, and M. Saffell, "Performance functions and reinforcement learning for trading systems and portfolios," *Journal of Forecasting*, vol. 17, no. 5-6, pp. 441–470, 1998.

[2] Z. Jiang, D. Xu, and J. Liang, "A deep reinforcement learning framework for the financial portfolio management problem," *arXiv preprint arXiv:1706.10059*, 2017.

[3] Z. Wang, B. Huang, S. Tu, K. Zhang, and L. Xu, "Deeptrader: A deep reinforcement learning approach for risk-return balanced portfolio management with market conditions embedding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1. AAAI Press, 2021, pp. 643–650.

[4] W. Zhang, Y. Zhao, S. Sun, J. Ying, Y. Xie, Z. Song, X. Wang, and B. An, "Reinforcement learning with maskable stock representation for portfolio management in customizable stock pools," in *Proceedings of the ACM Web Conference 2024*, ser. WWW '24, 2024, p. 187–198. [Online]. Available: https://doi.org/10.1145/3589334.3645615

[5] J. M. Poterba and L. H. Summers, "Mean reversion in stock prices: Evidence and implications," *Journal of Financial Economics*, vol. 22, no. 1, pp. 27–59, 1988.

[6] N. Jegadeesh and S. Titman, "Returns to buying winners and selling losers: Implications for stock market efficiency," *The Journal of Finance*, vol. 48, no. 1, pp. 65–91, 1993.

[7] T. J. Moskowitz, Y. H. Ooi, and L. H. Pedersen, "Time series momentum," *Journal of Financial Economics*, vol. 104, no. 2, pp. 228–250, 2012.

[8] L. K. Chan, N. Jegadeesh, and J. Lakonishok, "Momentum strategies," *The Journal of Finance*, vol. 51, no. 5, pp. 1681–1713, 1996.

[9] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 785–794.

[10] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 3146–3154.

[11] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," *arXiv preprint arXiv:1704.02971*, 2017.

[12] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[13] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018.

[14] Y. Ding, P. Liu, and X. S. Wang, "Investor-imitator: A framework for trading knowledge extraction and decision support," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, 2018, pp. 1310–1319.

[15] Y. Ye, Y. Pei, B. Wang, H. Zhou, and P. Liu, "Reinforcement-learning based portfolio management with augmented asset movement prediction states," *AAAI Conference on Artificial Intelligence*, pp. 1112–1119, 2020.

[16] Z. Wang, B. Huang, K. Zhang, and L. Xu, "Hierarchical reinforcement portfolio management with risk control," *Expert Systems with Applications*, vol. 179, p. 115013, 2021.

[17] S. Sun, X. Wang, Y. Zhao, W. Zhang, and B. An, "Deepscalper: A risk-aware reinforcement learning framework for high-frequency trading," *arXiv preprint arXiv:2202.02795*, 2022.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 30, 2017, pp. 5998–6008.

[19] Z. Li and V. Tam, "Developing an attention-based ensemble learning framework for financial portfolio optimisation," in *2024 International Joint Conference on Neural Networks (IJCNN)*, 2024, pp. 1–8.

[20] J. Wang, Y. Zhang, K. Tang, J. Wu, and Z. Xiong, "Alphastock: A buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 2019, pp. 1900–1908.

[21] Y. Ye, H. Pei, B. Wang, P.-Y. Chen, Y. Zhu, J. Xiao, and B. Li, "Reinforcement-learning based portfolio management with augmented asset movement prediction states," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*, 2020, pp. 1112–1119.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2019, pp. 4171–4186.