Staged Pre-Training with Frozen Backbone and Parameter-Efficient Fine-Tuning for Financial RL Portfolio Optimization

Anonymous Author(s)

Abstract

Reinforcement learning (RL) has shown promise in portfolio management, particularly in handling customizable stock pools (CSPs). However, prevailing joint end-to-end training approaches, exemplified by EarnMore, suffer from severe gradient interference in financial environments characterized by extremely sparse, noisy, and delayed rewards. This often leads to unstable representation learning, poor generalization, and degraded performance under market regime shifts.To address these challenges, we propose **SPF**(Staged Pre-training with Frozen backbone), a decoupled, staged framework tailored for low-signal financial RL. First, we pre-train a robust, mask-aware representation via multi-task self-supervision, combining masked reconstruction with latent-space contrastive learning to capture both local price patterns and inter-stock semantic relationships. The pre-trained backbone is then fully frozen to preserve financial inductive biases and prevent destructive interference from weak RL gradients. Finally, we perform highly parameter-efficient fine-tuning using a hierarchical residual lightweight adapter chain (inspired by Lossless Adaptation) at multiple levels of abstraction, while confining RL updates to only the adapters, SAC critic, and entropy coefficient (with optional actor adaptation).Extensive experiments on the DJ30 and SP500 datasets, including Global Stock Pool (GSP) and multiple CSP benchmarks, demonstrate that **SPF** significantly outperforms joint-training baselines, achieving substantial improvements in risk-adjusted metrics (e.g., Sharpe ratio, cumulative return, Calmar ratio) and enhanced robustness across out-of-sample periods and market regimes. The proposed approach also exhibits lower variance and better adaptation to transaction costs. Our code is publicly available at https://github.com/Flying932/PFRL.

CCS Concepts

• **Computing methodologies** → **Reinforcement learning**; *Neural networks*; *Machine learning*; • **Applied computing** → Economics.

Keywords

Reinforcement Learning, Portfolio Management, Customizable Stock Pools, Self-Supervised Pre-training, Frozen Representations, Parameter-Efficient Fine-Tuning, Financial Time Series

ACM MM '26, Rio de Janeiro, Brazil

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Anonymous Author(s). 2026. Staged Pre-Training with Frozen Backbone and Parameter-Efficient Fine-Tuning for Financial RL Portfolio Optimization. In *Proceedings of Proceedings of the 34th ACM International Conference on Multimedia (ACM MM '26).* ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Portfolio management under customizable stock pools (CSPs) constitutes a challenging sequential decision-making task in which the available asset universe changes dynamically[cite: 58, 79]. Recent reinforcement learning (RL) methods, particularly EarnMore [**?**], have advanced this domain by introducing maskable representations that enable a single model to adapt to varying stock pools[cite: 60, 80]. Despite this progress, such approaches rely predominantly on joint end-to-end training of representation learning and policy optimization[cite: 59, 81].

Financial RL environments exhibit extreme characteristics: rewards are sparse, noisy, and significantly delayed, while the underlying data distribution is highly non-stationary[cite: 59, 82, 114]. **A fundamental challenge arises from the deep misalignment between the dense signals in self-supervised learning and the sparse, high-variance rewards in RL**. While pre-training objectives like price reconstruction provide a dense supervisory signal across all tokens—enabling the model to capture the intricate "market physics"—RL rewards are often dominated by idiosyncratic market noise. **Under joint training, early excessive exploration involves stochastic actions that generate high-variance gradients; these gradients, when backpropagated, can effectively "reset" the encoder's weights**. Consequently, the model loses its painstakingly learned sensitivity to critical price trends and inter-stock relationships, leading to unstable representations and substantial performance degradation during market regime shifts.

To address these issues, we adopt a staged pre-training and fine-tuning paradigm that deliberately decouples representation learning from policy optimization[cite: 60, 86, 370]. **By treating the pre-training as a dense knowledge-acquisition stage, we establish a robust and stable initialization that captures temporal dynamics and market patterns from unlabeled data**[cite: 61, 87]. This stage remains largely unaffected by the sparsity and noise of subsequent RL rewards[cite: 88, 119]. During fine-tuning, the pre-trained Transformer backbone is fully frozen to protect these high-quality representations from the destructive interference of sparse RL gradients[cite: 62, 89, 192]. **This "freeze-and-adapt" strategy ensures that the RL agent builds upon a fixed, reliable financial foundation rather than corrupting it through noisy updates**.

Adaptation to the downstream RL task occurs through a hierarchical residual lightweight adapter chain (inspired by Lossless

Adaptation [?]), inserted at multiple levels of abstraction[cite: 63, 193, 451]. RL updates are strictly confined to the adapters, SAC critic networks, and entropy coefficient, ensuring minimal disruption to the frozen financial priors[cite: 63, 194, 619].

The remainder of this paper is organized as follows: Section 2 reviews related work, Section 3 presents the proposed framework, Section 4 reports experimental results, and Section 5 concludes the paper.

2 Related Work

2.1 Joint Representation Learning with RL

Joint optimization of state representations and RL policies has been a dominant approach in recent portfolio management research. Yang et al. (2023) highlighted that many existing RL methods treat assets in isolation and optimize solely for episodic rewards, neglecting the global portfolio context. They introduced the Task-Context Mutual Actor–Critic (TC-MAC) algorithm, which simultaneously learns a task-context representation encoding both individual asset features and the overall dynamic portfolio context, along with a policy optimized through a combined RL and mutual-information loss. By maximizing mutual information between local asset embeddings and the global context, TC-MAC achieves more coherent portfolio representations and improved policy optimality, outperforming traditional methods and prior RL baselines with enhanced transferability to unseen market data.

EarnMore [?] represents a key advancement in this direction. To address the limitation of fixed asset universes, it performs one-shot training on a large global stock pool and proposes maskable stock representations that adapt to arbitrary customizable subsets. The method employs a self-supervised autoencoder for masked reconstruction and uses learnable mask tokens to suppress irrelevant stocks, combined with a re-weighting mechanism to concentrate the portfolio on promising assets. The authors adopt a three-stage joint end-to-end training pipeline and argue that freezing pretrained embeddings may restrict RL exploration and task alignment. EarnMore achieved substantial profit gains (over 40%) over 14 baseline methods on U.S. stock datasets. However, joint training exposes the representation module to weak and noisy RL gradients, potentially leading to sample inefficiency and instability in reward-sparse financial environments.

2.2 Incorporating Financial Knowledge and Hybrid Training

Several works integrate domain knowledge or supervised signals to improve sample efficiency and alignment with financial objectives. Hu and Gu (2024) combined Modern Portfolio Theory with deep RL in a two-stage framework. The first stage distills the agent on signals from Markowitz optimal portfolios (balancing return and risk), while the second stage performs RL fine-tuning. This knowledge-distillation approach (KDD) provides a strong warm start and yields higher returns and risk-adjusted performance (Sharpe ratio of 2.03) compared to pure RL and traditional baselines.

Li et al. (2025) proposed Logic-Q, a neural-symbolic framework that augments deep RL with rule-based trend detection for quantitative trading. By injecting lightweight program sketches encoding human-defined logic (e.g., macro trend indicators), Logic-Q enables the agent to adjust risk exposure or actions based on market regimes. Evaluations demonstrated significant reductions in catastrophic drawdowns during downtrends and improved exploitation in uptrends, outperforming prior DRL strategies.

2.3 Representation Learning for Stability and Generalization

Representation learning has become central to enhancing RL stability and generalization in financial environments. Wu et al. (2023) addressed overfitting and volatility by combining contrastive learning with reward smoothing. The contrastive objective encourages trend-aware state representations that cluster similar long-term dynamics and separate dissimilar ones, while reward smoothing mitigates short-term noise. This approach improved consistency and robustness on both stock and cryptocurrency markets.

Other efforts leverage graph neural networks to capture cross-asset correlations and sector dynamics, or attention mechanisms to dynamically focus on relevant assets and market signals. These representation-oriented techniques consistently demonstrate improved robustness and generalization compared to raw price-based state representations.

In summary, recent advances have significantly improved performance through richer representations, domain knowledge integration, auxiliary tasks, and hybrid training schemes. Nevertheless, most methods rely on joint end-to-end optimization or partial pre-training, leaving room for fully decoupled paradigms that protect high-quality representations from sparse RL interference while enabling efficient task-specific adaptation. Our work builds on these insights by proposing a staged, frozen-backbone approach with multi-task self-supervision and hierarchical adapters, specifically tailored to the low-signal, non-stationary nature of financial portfolio optimization.

3 Method

We propose **SPF** (Staged Pre-training with Frozen backbone), a decoupled framework that separates dense self-supervised representation learning from sparse RL optimization for robust portfolio management in customizable stock pools (CSPs). The framework consists of three main stages: (1) multi-task self-supervised pre-training, (2) complete backbone freezing, and (3) parameter-efficient fine-tuning with RL adaptation.

3.1 Stage 1: Multi-Task Self-Supervised Pre-Training

The goal of the pre-training stage is to learn high-quality, mask-aware stock representations that capture both local temporal patterns and inter-stock semantic relationships, providing a strong and stable initialization for downstream RL.

We build directly on the self-supervised tasks introduced in EarnMore [?]. Let $X \in \mathbb{R}^{B\times T\times N\times D}$ denote a batch of stock time-series sequences, where B is the batch size, T is the look-back window, N is the number of stocks in the global pool, and D is the feature dimension (OHLCV + technical indicators).

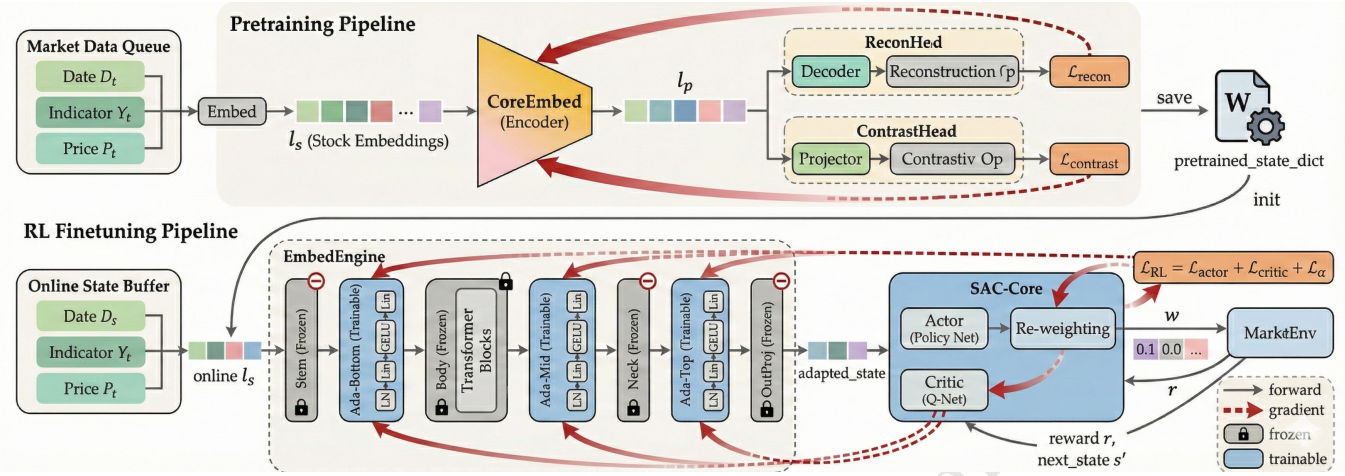

Figure 1: Overview of the SPF (Staged Pre-training with Frozen backbone) framework. Stage I (top) illustrates the multi-task self-supervised pre-training pipeline, where the encoder (CoreEmbed) learns from both reconstruction and contrastive signals. Stage II (bottom) shows the RL fine-tuning process. The pre-trained backbone is frozen to preserve financial semantic knowledge, while hierarchical residual adapters (Ada-Bottom, Ada-Mid, and Ada-Top) are optimized alongside the SAC agent to adapt to specific portfolio management tasks in customizable stock pools.

3.1.1 Masked Reconstruction (Inherited from EarnMore). We randomly mask a portion (mask ratio $\rho \in [0.15, 0.3]$) of the input tokens along the time and stock dimensions. Let M be the binary mask matrix with the same shape as X, where $M_{b,t,n,d} = 1$ indicates a masked position. The encoder $E(\cdot)$ (a Transformer-based architecture) processes the masked input $X \odot (1 - M) + m \cdot M$, where m is a learnable mask token. The decoder $D(\cdot)$ reconstructs the original values at masked positions:

$$\hat{X} = D(E(X \odot (1 - M) + m \cdot M)) \tag{1}$$

The reconstruction loss is the mean squared error over masked positions only:

$$\mathcal{L}_{\text{recon}} = \frac{1}{\sum M} \sum_{b,t,n,d} M_{b,t,n,d} \cdot (\hat{X}_{b,t,n,d} - X_{b,t,n,d})^2 \tag{2}$$

This task encourages the encoder to learn local temporal dynamics and stock-specific patterns.

3.1.2 Latent Contrastive Learning (Our Extension). To enrich the representations with robust semantic consistency, we introduce a dual-view contrastive learning objective in the latent space. Unlike traditional stock-pair contrast, our approach constructs views through stochastic perturbations. Specifically, for each mini-batch of market state windows $X \in \mathbb{R}^{B \times T \times N \times D}$, we perform two independent forward passes through the encoder $E(\cdot)$ with different random masking seeds (maintaining the same mask ratio ρ). This yields two distinct views of the same batch: $Z^{(1)} = E(X; \text{seed}_1)$ and $Z^{(2)} = E(X; \text{seed}_2)$, where $Z^{(k)} \in \mathbb{R}^{B \times C}$ represents the global latent embedding after temporal and pool-level pooling.

Positive and Negative Pairs. For each anchor embedding $z_i^{(1)}$ in $Z^{(1)}$, we define its corresponding embedding from the same market state, $z_i^{(2)}$, as the unique positive pair. The remaining $2B - 2$ embeddings within the combined set $\{Z^{(1)}, Z^{(2)}\}$ (excluding the anchor and its positive pair) are treated as negative samples, representing different time windows or market regimes.

Contrastive Objective. Following the NT-Xent (Normalized Temperature-scaled Cross Entropy) framework, we apply L2-normalization to all embeddings and compute similarity via dot products[cite: 378, 382]. The loss for a single anchor i is formulated as:

$$\ell_i = -\log \frac{\exp(z_i^{(1)\top} z_i^{(2)} / \tau)}{\sum_{k=1}^{B} [\exp(z_i^{(1)\top} z_k^{(1)} / \tau) + \exp(z_i^{(1)\top} z_k^{(2)} / \tau)] - \exp(z_i^{(1)\top} z_i^{(1)} / \tau)} \tag{3}$$

where $\tau = 0.2$ is the temperature coefficient. The total contrastive loss $\mathcal{L}_{\text{contrast}}$ is the average over all $2B$ samples in the dual-view batch[cite: 412, 443].

This task encourages the model to maintain semantic invariance under masking noise—emphasizing that the "same market state" should have a consistent representation regardless of partial occlusion—while distinguishing between different market conditions.

Discussion. While effective for robust feature extraction, we acknowledge that this masking-based contrast primarily captures stochastic invariance. Current negative samples from the same batch may contain "false negatives" (e.g., temporally adjacent or semantically similar windows)[cite: 191]. Future extensions could incorporate stronger financial inductive biases, such as industry-consistent or correlation-aware positive sampling[cite: 61, 92].

The total pre-training loss is thus:

$$\mathcal{L}_{\text{pre}} = \lambda_1 \mathcal{L}_{\text{recon}} + \lambda_2 \mathcal{L}_{\text{contrast}} \tag{4}$$

where we set $\lambda_1 = \lambda_2 = 1.0$[cite: 443, 444].

Table 1: Performance comparison on SP500 and DJ30 with Global Stock Pool. Results in red, yellow and green show the best, second best and third best results on each dataset.

2*Categories	2*Strategies	SP500				DJ30			
		ARR↑	SR↑	CR↑	SOR↑	ARR↑	SR↑	CR↑	SOR↑
3*Rule-based	Market	9.320	0.556	0.702	17.120	6.710	0.458	0.776	15.960
	BLSW	11.630	0.696	0.894	21.560	7.610	0.512	0.858	16.560
	CSMBoost	5.070	0.329	0.434	9.840	5.930	0.400	0.643	12.950
2*ML-based	XGBoost	10.690	0.377	0.575	13.650	10.340	0.343	0.599	14.220
	LightGBM	16.330	0.575	0.744	13.110	10.320	0.591	0.703	14.220
2*DL-based	ALSTM	43.560	1.157	1.044	22.550	15.930	1.186	0.732	8.890
	TCN	13.560	0.875	0.744	16.350	6.980	0.186	0.269	8.490
5*RL-based	PG	12.580	0.431	0.519	24.370	7.920	0.321	0.435	8.040
	PPO	15.130	0.538	0.743	15.470	22.900	0.689	1.465	8.350
	SAC	15.034	0.538	0.673	14.570	20.900	0.326	0.445	8.670
	EIIE	21.240	0.756	0.970	25.230	27.192	0.786	1.109	27.320
	DeepTrader	50.320	1.162	0.949	35.230	27.420	0.909	1.593	27.230
	SPF (Ours)	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**

3.2 Stage 2: Complete Backbone Freezing

After pre-training, the entire Transformer encoder $E(\cdot)$ is frozen (parameters set to require no gradient). This step protects the high-quality financial representations learned from dense self-supervised signals from being corrupted by the sparse, noisy gradients of RL during fine-tuning.

3.3 Stage 3: Parameter-Efficient Fine-Tuning with RL Adaptation

In the fine-tuning stage, the frozen encoder produces representations $Z = E(X)$ for each input sequence. To adapt these representations to the RL task (optimal portfolio allocation) with minimal interference, we insert a hierarchical residual lightweight adapter chain at multiple levels of abstraction.

The adapter chain consists of three sequential residual adapters (bottom, mid, top):

- **Bottom Adapter**: operates on token-level embeddings
- **Mid Adapter**: operates on pooled stock-level representations
- **Top Adapter**: operates on global sequence representation

Each adapter applies a residual transformation: it first normalizes the input z via **Layer Normalization**, then processes it through a **low-rank bottleneck linear layer** (with **GELU activation**), followed by a second linear layer and **dropout**. Finally, the output is added back to the original input z through a **residual connection** to ensure stable information flow. This process is compactly represented as $\text{Adapter}(z) = z + \text{Dropout}(\text{Linear}_2(\text{GELU}(\text{Linear}_1(\text{LayerNorm}(z)))))$.where both linear layers use a low-rank projection (bottleneck dimension $r \ll C$, e.g., $r = 64$). The residual connection ensures that the adapter only learns task-specific corrections without overwriting the pre-trained features.

The adapted representation is then fed into the SAC agent. RL updates are strictly confined to: - all parameters of the hierarchical adapters - the SAC critic networks (dual Q-functions) - the entropy coefficient α (automatic temperature tuning) - the actor network (optional; our main experiments freeze the actor to further minimize interference)

This parameter-efficient fine-tuning (PEFT) strategy ensures stable convergence in the low-signal financial environment while adapting the pre-trained representations to the specific allocation task.

The overall objective in the fine-tuning stage is the standard SAC loss:

$$\mathcal{L}_{\text{SAC}} = \mathcal{L}_{\text{critic}} + \mathcal{L}_{\text{actor}} + \mathcal{L}_{\alpha} \tag{5}$$

where $\mathcal{L}_{\text{critic}}$ is the Bellman error on the dual Q-functions, $\mathcal{L}_{\text{actor}}$ maximizes $Q - \alpha \log \pi$, and $\mathcal{L}_{\alpha}$ enforces target entropy.

Through this three-stage design, SPF achieves robust, efficient, and generalizable RL-based portfolio management tailored to the unique challenges of financial domains.

4 Experimental Evaluation

4.1 Datasets

We conduct experiments on two widely-used financial benchmarks: DJ30 and SP500. The DJ30 dataset consists of the 30 constituent stocks of the Dow Jones Industrial Average, while SP500 includes the constituents of the S&P 500 index. Both datasets cover the period from September 26, 2007 to June 26, 2022, with input features comprising 102 dimensions. The prediction targets are ret1 (one-day return) and mov1 (movement label).

The data is split chronologically as follows:

- **Pre-training stage**: Training set: September 26, 2007 – July 22, 2019; Validation set: July 22, 2019 – June 26, 2022.
- **Fine-tuning stage**: Training set: September 26, 2007 – July 22, 2019; Validation set: July 22, 2019 – January 8, 2021; Test set: January 8, 2021 – June 26, 2022.

4.2 Experimental Settings

All experiments are conducted on an NVIDIA RTX 4090 GPU with 24GB memory. To ensure reliability and reduce variance, we perform 3 independent data splits and 5 random seeds for each setting, reporting the average and standard deviation.

Key hyperparameters are as follows: Pre-training uses a mask ratio $\rho = 0.25$, contrastive temperature $\tau = 0.2$, and reconstruction weight $\lambda_1 = 1.0$. Fine-tuning employs the SAC algorithm with a learning rate of 3×10^{-5} and a buffer size of 10^6. Other settings follow the default configurations of ElegantRL and Qlib.

4.3 Comparison with Baselines

We compare SPF against a range of baselines including rule-based, ML-based, and RL-based methods. Performance comparison under the Global Stock Pool (GSP) setting is detailed in Table 1. For Customizable Stock Pools (CSP), results are presented in Table 2.

Table 2: Performance comparison on SP500 and DJ30 with Customizable Stock Pools. Underlined metrics indicate the best-performing index results.

Pool	Strategies	SP500		DJ30	
		ARR↑	SR↑	ARR↑	SR↑
CSP1	SARL (Baseline)	34.333	0.820	24.141	0.638
	IMIT (Baseline)	20.973	0.860	20.741	0.928
	SPF (Ours)	122.610	2.278	53.990	1.810
CSP2	SARL (Baseline)	17.000	0.578	20.020	0.820
	DeepTrader (Baseline)	34.030	0.793	38.470	0.955
	SPF (Ours)	110.110	2.279	43.400	1.549
CSP3	SARL (Baseline)	18.090	0.760	10.910	0.480
	IMIT (Baseline)	61.320	1.489	16.840	0.601
	SPF (Ours)	93.670	2.120	43.460	1.572

4.4 Ablation Study

To verify the effectiveness of each component, we conduct an ablation study with variants: (1) w/o Pre-training, (2) w/o Contrastive, and (3) w/o Reconstruction. Table 3 reports the results, where red and green indicate performance changes.

Table 3: Ablation Study of SPF on SP500 and DJ30. Red, green and underline indicate improvement, performance decrease and best results, respectively.

2*Model	SP500			DJ30		
	ARR↑	SR↑	MDD↓	ARR↑	SR↑	MDD↓
w/o Pre-training	xxx	xxx	xxx	xxx	xxx	xxx
w/o Contrastive	xxx	xxx	xxx	xxx	xxx	xxx
w/o Reconstruction	xxx	xxx	xxx	xxx	xxx	xxx
Full SPF (Ours)	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**	**xxx**

5 Conclusion

We introduced a staged pre-training and frozen representation framework for robust RL in customizable portfolio management. Experiments show significant improvements over joint-training baselines in financial environments with weak signals. Future work includes multi-modal extensions and real-world deployment.

References