Reward Function Design and Evaluation in Reinforcement Learning for Portfolio Allocation

Bian Yuqi

National University of Singapore

Supervisor: Dr Akshay Narayan

National University of Singapore

Acknowledgments

I would like to express my sincere gratitude to Dr. Akshay Narayan for his invaluable guidance, support, and mentorship throughout this project. His insights, expertise, and thoughtful feedback have been instrumental in shaping the direction and quality of my work. I am deeply appreciative of the time, patience, and encouragement he has provided, which have not only helped me overcome challenges but also inspired me to grow academically and professionally.

Contents

1 Introduction

1.1 Background

Algorithmic trading has become increasingly dominant in financial markets. Recent estimates suggest that approximately 73% of U.S. equity trading volume is executed via algorithmic strategies [1], reflecting the pervasive role of automated, data-driven trading. This growing prevalence underscores the demand for advanced techniques that can offer a competitive edge in high-speed market environments.

Traditionally, machine learning (ML) methods in finance have focused on price prediction, but they face notable limitations. Financial market data exhibit a low signal-to-noise ratio and are highly non-stationary, causing ML models to latch onto spurious patterns that fail to generalize [2]. As a result, purely predictive models often show unstable performance out-of-sample[3]. Moreover, ML models struggle with extrapolation–they cannot reliably predict values outside the range of their training data, a fundamental limitation inherent to neural networks and tree-based methods[4]. These limitations constrain the effectiveness of conventional supervised learning approaches in trading.

Reinforcement learning (RL) offers an alternative by modeling trading as a sequential decision-making process. In an RL framework, an agent interacts with the market environment and learns a policy (trading strategy) through trial and error, receiving rewards (e.g., profits or risk-adjusted returns) for its actions [5]. Unlike static predictive models, an RL trading agent continuously adjusts its decisions (buy, sell, hold) based on feedback, potentially handling dynamic market conditions and delayed effects more robustly.

Deep reinforcement learning (DRL) techniques have demonstrated remarkable success in other complex domains. DRL agents have achieved superhuman performance in games – DeepMind's AlphaGo defeated a Go world champion [6], and OpenAI Five beat the world champion Dota 2 team [7]. These achievements highlight how an agent learning from reward feedback in a challenging environment can surpass human experts. Notably, the same agent-environment-reward structure of DRL readily applies to financial trading, where the "game" is the market and the objective is to maximize returns.

Consequently, there is growing interest in applying RL and DRL across different asset classes. In equity and cryptocurrency markets, deep RL has been used to learn trading policies and manage portfolios; for example, DRL-based cryptocurrency trading agents have demonstrated significant portfolio gains [8]. In the derivatives domain, advanced "deep hedging" approaches employ DRL to optimize options trading and dynamic hedging under real-world market frictions [9]. These early successes across stocks, crypto, and options underscore the potential of sequential decision algorithms to enhance trading strategies in complex, volatile markets.

1.2 Motivation

A widespread methodological weakness in RL trading research is the dependence on a narrow, pre-selected group of securities for both training and evaluation. Many studies showcase the performance of a new algorithm using only a small set of strong technology stocks—often the well-known FAANG names (Facebook, Amazon, Apple, Netflix, Google). Although this setup simplifies the learning task, it inevitably introduces substantial selection bias.[10]

A fundamental component of Reinforcement Learning is the definition of the Action Space ($\mathcal{A}$)—the set of all valid moves the agent can make in a given state. In portfolio optimization problems, the action space is typically defined as a continuous vector of portfolio weights $\mathbf{w} = \{w_1, w_2, ..., w_n\}$ such that $\sum_{i=1}^{n} w_i = 1$, where n is the number of risky assets. This formulation imposes a critical constraint: the agent must be fully invested in the market at all times. [11] While the agent can rotate capital between Stock A and Stock B, it cannot retreat from the market entirely. In favorable market regimes (bull markets), this constraint is non-binding. However, in unfavorable regimes (bear markets, financial crashes), this "forced trading" exposes the agent to unavoidable systematic risk (beta). Even the "safest" stock in a crashing market often correlates to 1.0 with the index, leading to significant drawdowns. [10]

A further limitation is that most studies adopt a single experimental configuration—such as one reward function or one set of market conditions—without systematically examining how RL behavior changes under different design choices.

This paper addresses these methodological limitations in three ways. First, it employs the Dow Jones 30 component stocks as the asset universe, providing broader market representation and reducing selection bias.

Second, it explicitly incorporates cash as an available action, allowing the agent to adopt risk-off positioning during unfavorable market conditions. Third, it systematically examines how different reward formulations shape trading behavior and model performance. Understanding these reward dynamics is essential not only for building robust RL trading agents but also for facilitating educational applications and enabling future research extensions within existing RL trading frameworks.

Moreover, the motivation of this study is to systematically evaluate a range of reward functions so that a robust suite of them can be developed for use as a teaching aid and seamlessly integrated into the reinforcement-learning stock-trading framework implemented by Wu Hanyu under the supervision of Dr. Akshay Narayan. In doing so, this project complements Hanyu's work by providing well-tested benchmark results, a curated library of reward functions, and a set of tunable parameters to support future research and instructional use.

1.3 Problem Statement

This paper conducts a comprehensive empirical study on how the choice of reward function influences the performance, stability, and risk profile of reinforcement-learning-based portfolio allocation. Specifically, we evaluate a broad set of reward formulations—ranging from profit-driven metrics to risk-adjusted and drawdown-aware objectives—within a continuous-action allocation framework. To ensure robustness across algorithmic families, we benchmark these rewards using three widely adopted actor–critic algorithms: Proximal Policy Optimization (PPO), Twin Delayed Deep Deterministic Policy Gradient (TD3), and Soft Actor–Critic (SAC). By systematically comparing their behavior under identical market conditions, we aim to uncover the sensitivity of each algorithm to reward design, provide insights into which reward structures lead to more stable and realistic trading behavior, and highlight the interactions between reward shaping and policy learning in continuous financial decision-making.

2 Literature Review

2.1 RL Algorithm for Trading - Major Developments

Reinforcement Learning (RL) has been applied to stock trading and portfolio management by framing the problem as a sequential decision process (trading policy that adapts to market data [12]. Moody and Saffel [13] were among the first to apply reinforcement learning in finance to optimize trading policies directly by maximizing financial metrics like profit, Sharpe ratio, and other utility functions. The advent of deep RL enabled more powerful function approximation for trading strategies. Jiang et al. (2017) introduced a model-free Reinforcement Learning (RL) framework for cryptocurrency portfolio management [8]. The framework uses an Ensemble of Identical Independent Evaluators (EIIE) topology, a Portfolio-Vector Memory (PVM), and an Online Stochastic Batch Learning (OSBL) scheme, realized through CNN, RNN, and LSTM networks. The RL approach significantly outperformed traditional strategies in back-tests, achieving high returns despite high transaction costs. Subsequent studies built on these advances, Liu et al. (2018) adopted the Deep Deterministic Policy Gradient (DDPG) algorithm (an actor-critic method for continuous actions) and achieved remarkable performance improvement over earlier policy-gradient approaches.[14] This highlighted the benefit of deep actor-critic networks in portfolio management tasks.

Following DDPG, researchers explored a range of reinforcement learning (RL) algorithms. Gao et al. [15] applied Deep Q-Networks (DQN) with improvements such as dueling Q-networks and prioritized experience replay to a discrete-action portfolio problem, improving sample efficiency and performance. However, pure discrete-action methods are typically infeasible for large portfolios due to the exponential action space. This limitation motivated the shift to continuous-action methods. Newer actor–critic algorithms such as Twin Delayed DDPG (TD3), which mitigates DDPG's overestimation bias using twin critics and delayed policy updates, and Soft Actor-Critic (SAC), which optimizes a stochastic policy with entropy regularization for improved stability, have been increasingly applied in trading research [16, 17]. These off-policy methods enable efficient experience reuse and have demonstrated strong results in financial environments [12].

On-policy policy-gradient methods like Advantage Actor-Critic (A2C/A3C) and Proximal Policy Optimization (PPO) have also been investigated. PPO, known for its stable learning via a clipped surrogate

objective, has been widely adopted in financial RL toolkits such as FinRL [18] and has shown potential for profitable strategies [19].

Beyond single-agent approaches, advanced architectures have emerged. Hierarchical reinforcement learning has been used to separate long-term strategic planning from short-term execution: for example, a high-level agent optimizes long-run profitability while a low-level agent minimizes trading costs [20]. Multi-agent and parallel-agent models have also been explored; Ma et al. [21] used dual agents—one focusing on short-term asset signals, the other leveraging LSTMs for long-term trends—to enhance performance. Domain-specific knowledge has been integrated in models like AlphaStock [22], which combines momentum-based heuristics ("buy winners, sell losers") with dual RL agents (long and short) optimized via a Sharpe-ratio objective.

To address non-stationary market conditions, hybrid ensemble methods have been proposed. Yang et al. [19] introduced an adaptive framework that dynamically switches between PPO, A2C, and DDPG based on market regime indicators, improving robustness under volatility. These innovations illustrate a broader trend toward combining multiple RL paradigms and neural architectures to better align with the complexities of real-world trading environments.

2.2 Market-Tracking Behavior in RL Trading Agents

Several studies have observed that deep reinforcement learning (DRL) trading agents often converge toward strategies that mirror broad market movements (i.e., index-like behavior) rather than generating unique alpha:

- A study on portfolio management demonstrated that DRL agents failed to statistically outperform market indices, with alpha generation not significantly different from zero. The trading policies exhibited high-risk opportunistic behavior focused on individual trades rather than consistent market outperformance, suggesting that even sophisticated neural network architectures struggle to beat simple buy-and-hold strategies [23].

- Research on deep reinforcement learning for portfolio allocation found that agents using A2C, PPO, and DDPG algorithms exhibited similar correlation patterns with market benchmarks. When comparing against buy-and-hold strategies across multiple metrics, several DRL agents showed comparable or lower risk-adjusted returns than passive investment approaches, particularly when transaction costs were considered [24].

- An ensemble trading strategy study evaluated A2C, PPO, and DDPG agents on Dow Jones stocks during the Q1 2020 COVID-19 market crash. The A2C agent demonstrated the lowest maximum drawdown of -10.2% during this period, while still experiencing significant portfolio value reductions concurrent with the broader market decline. The study noted that agents performed better when incorporating turbulence indices to detect extreme market conditions, suggesting they were vulnerable to systemic market movements [19].

- Analysis of DRL trading behavior across multiple algorithms revealed that agents exhibit unique but often convergent trading patterns. While A2C emerged as a top performer in cumulative rewards, multiple algorithms showed tendencies toward either concentrated trading in limited stocks or extended holding periods that resembled passive strategies. This homogeneity in strategic outcomes suggests agents learn similar market-following behaviors despite architectural differences [25].

- A comprehensive study applying actor-critic algorithms to the Australian stock market found that the A2C agent exhibited strong buy-and-hold tendencies, making only two sell transactions over a two-year testing period while accumulating a portfolio of 17 stocks. The agent's trading behavior was characterized by extended holding periods and minimal rebalancing, closely mimicking passive index-tracking approaches rather than active trading strategies [26].

- Research on DRL portfolio optimization using twin-delayed deep deterministic policy gradient (TD3) algorithms demonstrated superior performance against benchmarks when explicitly embedding risk aversion and transaction cost constraints. However, the study acknowledged that without careful

reward function design and cost considerations, DRL agents tend to adopt strategies resembling those of traditional mean-variance portfolios, indicating a natural convergence toward established market-tracking methodologies [27].

This convergence toward market-tracking likely reflects the sparse, noisy nature of financial reward signals, which pushes agents toward simpler policies that follow market momentum rather than discovering genuine alpha. The prevalence across diverse algorithms suggests this represents a fundamental challenge in financial RL rather than algorithm-specific limitations.

2.3 Technical Indicators in Reinforcement Learning

Technical indicators condense price and volume series into engineered signals that capture trend, momentum, volatility, and participation. In discretionary and rules-based trading, indicators are used via fixed thresholds (e.g., crossovers, overbought/oversold). In reinforcement learning (RL), they are typically provided as state features; the agent learns context-dependent policies that weigh these signals with the reward objective, often outperforming static rules when well-regularized and evaluated out-of-sample [28, 29, 18].

Moving Averages (MA). Simple and exponential moving averages smooth noise and define trend; classic golden/death cross rules have documented historical efficacy and limitations such as lag and whipsaws [28]. RL systems feed multiple MA horizons (e.g., 5/20/50/200) as features so that agents learn when to trust crossovers and how to size positions by regime instead of hardcoding thresholds [29, 18].

Relative Strength Index (RSI). RSI measures normalized momentum over a typical 14-period window with heuristic zones at 70/30 [30]. Traditional traders fade extremes or use divergences; RL agents exploit RSI as a bounded, scale-stable momentum channel, learning asymmetric behaviors (e.g., ignore "overbought" in strong uptrends, act on mean reversion in ranges) through reward feedback [18].

MACD The Moving Average Convergence Divergence computes the spread of fast/slow EMAs with a signal EMA and histogram; practitioners use zero-line and signal crossovers for trend-momentum timing [28, 31]. In RL, MACD (line, signal, histogram) enriches the state with momentum curvature, letting agents distinguish between small crossovers during low volatility and wide, expanding separations [29].

Bollinger Bands. A mid simple moving average with $\pm k\sigma$ envelopes estimates relative price extremes and a volatility "squeeze" precursor to breakouts [32]. Rules typically involve mean reversion at bands and breakout on expansion. RL leverages band width and percent-B features to condition actions on regime: fade extremes in ranges, follow when breaks occur after sustained compression [18].

Stochastic Oscillator. %K compares the close within the recent high–low range while %D smooths %K. Thresholds (80/20) and %K/%D crosses echo RSI-like uses [28]. In RL, the stochastic oscillator provides a fast, bounded range-position feature helpful for short-horizon timing; policies learn to discount frequent false signals during persistent trends.

Traditional vs. RL Use. Fixed-rule indicator systems are interpretable but brittle across regimes. RL treats indicators as informative features, learning nonlinear, context-aware mappings from indicator configurations to actions while optimizing risk-adjusted rewards [29]. Cautions include look-ahead bias, transaction costs, and overfitting; robust evaluation requires walk-forward windows and risk-aware objectives [18]. Early RL trading systems (e.g., [13]) and recent libraries [18] show that embedding well-understood indicators accelerates learning and improves explainability relative to raw-price-only inputs.

Across both academia and industry, a compact set of indicators—moving averages, RSI, MACD, Bollinger Bands and stochastic oscillators remain useful. In RL systems, they serve as domain-informed scaffolding that helps agents learn regime-dependent policies, bridging human interpretability with data-driven adaptability [28, 29, 18].

2.4 Reward Function Design in Financial Reinforcement Learning

A reinforcement learning agent's behavior in trading is entirely driven by the reward signal, so choosing an appropriate reward function is critical to shaping the desired strategy and performance [29]. Early applications of RL to trading optimized straightforward objectives like profit or wealth gain, reflecting the intuitive goal of maximizing returns [8]. Raw returns or profit-based rewards (e.g., cumulative P&L or portfolio value increase) have been used extensively in the literature [19, 33]. Such reward formulations do encourage high profits, but they ignore risk—often yielding agents that pursue unstable, high-variance

strategies. This limitation has led researchers to incorporate various risk-adjusted reward structures that balance return and risk in the objective function [12].

Common reward formulations in financial RL include the following (often in combination):

- **Cumulative Return or Profit:** Many studies use raw return (or log-return) of the portfolio or episodic profit as the reward signal [33, 19]. This directly drives the agent to maximize wealth but fails to account for transaction costs, capital preservation, or risk control [34].
- **Sharpe Ratio:** To encourage risk-adjusted performance, numerous works optimize Sharpe ratio (return divided by volatility) [35, 36]. Using this reward typically results in more stable strategies with lower volatility. Differential Sharpe Ratio (DSR), a variant for online updating, is also used for finer control.
- **Information Ratio:** The information ratio (excess return over benchmark normalized by tracking error) encourages outperforming the market with consistency. Wang et al. [37] successfully combine this with turnover-aware terms to enhance alpha generation and reduce churn.
- **Drawdown Penalties:** Reward structures that penalize large drawdowns help the agent avoid steep capital losses. These are commonly implemented as penalties on maximum drawdown or rolling drawdown [34, 12].
- **CVaR-Based Rewards:** Conditional Value-at-Risk (CVaR) focuses on tail-risk. Reward functions based on CVaR penalize large losses and have been used to guide policies toward downside protection [38].

Empirical studies confirm that reward design substantially influences the behavior and success of RL traders. Profit-only rewards often lead to unstable learning and volatile outcomes [34], whereas Sharpe- or CVaR-based rewards tend to improve both convergence and out-of-sample performance [35]. Composite or multi-objective reward functions are increasingly used to align agent behavior with investor objectives across return, risk, and cost dimensions [39, 37].

In summary, reward function design is central to the success of financial RL systems. As highlighted in surveys like Zhai et al. [33] and reviews such as Bai et al. [12], properly engineered reward signals—especially those incorporating risk controls—are the strongest determinant of whether an agent learns effective and deployable trading strategies.

3 Methodology

3.1 Portfolio Environment

We consider a discrete-time allocation problem over the Dow 30 universe augmented with a cash asset. At decision time t, the environment provides an observation vector

$$o_t = \big[\phi_t^{(1)}, \ldots, \phi_t^{(N)},\, w_{t-1},\, c_{t-1},\, \mathbf{1}^\top w_{t-1},\, \tfrac{V_{t-1}}{V_0}\big],$$

where $\phi_t^{(i)} \in \mathbb{R}^5$ encodes engineered features for stock i comprising the daily simple return $r_t^{(i)} = \frac{P_t^{(i)} - P_{t-1}^{(i)}}{P_{t-1}^{(i)}}$, two moving-average divergences, a 20-day realized volatility, and a scaled 14-day RSI. The portfolio state (w_{t-1}, c_{t-1}) captures the previous allocations to risky assets and cash, respectively, and V_t denotes the portfolio value.

The policy outputs an unconstrained action $a_t \in \mathbb{R}^{N+1}$ drawn from a Box space with component-wise bounds $[-10, 10]$. These limits are exposed to the learning algorithms prior to the softmax, capping the magnitude of each logit while still permitting confident preferences. Smaller limits drove the policy toward near-equal allocations that shadowed the buy-and-hold benchmark, whereas materially larger limits caused numerical overflow in the softmax and encouraged brittle, one-hot allocations. Even with the $[-10, 10]$ choice

the agents occasionally saturate all components at the boundary, but such situations are rare and typically short-lived. The bounded logits are subsequently mapped to budget-feasible allocations via a softmax layer

$$\pi_t = \sigma(a_t), \qquad \pi_t^{(k)} = \frac{\exp\{a_t^{(k)}\}}{\sum_{j=1}^{N+1} \exp\{a_t^{(j)}\}},$$

with the last component representing the cash weight. Executed allocations produce the next-step gross return

$$R_t = \pi_t^{(\text{cash})} r_f + \sum_{i=1}^{N} \pi_t^{(i)} r_{t+1}^{(i)},$$

where r_f is the fixed daily risk-free rate derived from an annual 3% benchmark. The reward $g(o_t, a_t)$ is supplied by a configurable library encompassing risk-adjusted (e.g., Sharpe-like), downside-aware, momentum, and diversification objectives, allowing a consistent comparison of reward sculpting strategies across agents.

3.2 Algorithm Exclusion Rationale

Discrete-Action Methods (DQN)

Canonical discrete-action methods such as DQN[40] are excluded. With $N \approx 30$ tradable assets and three primitive decisions per asset (buy, hold, sell), the joint action space is the Cartesian product

$$\mathcal{A} = \{\text{buy}, \text{hold}, \text{sell}\}^N, \qquad |\mathcal{A}| = 3^N,$$

which yields an exponentially large discrete action space. Maintaining Q-values for such a large number of discrete actions is computationally infeasible [41], and even coarse discretizations of trade size produce exponential blow-ups. Extending the action space to specify continuous trade sizes further exacerbates this combinatorial explosion, making pure value-based Q-learning algorithms impractical for the portfolio allocation problem considered here.

Pure Policy-Based Methods

The study also omits pure policy-gradient methods such as REINFORCE [42] because several reward functions explored—including Sharpe ratio objectives and drawdown-penalized criteria—are non-additive and depend on cumulative performance over an episode rather than on stepwise rewards. Policy gradients assume an additive objective of the form

$$J(\theta) = \mathbb{E}_\theta \left[\sum_{t=0}^{T-1} \gamma^t r_t \right],$$

where unbiased gradient estimates hinge on per-step rewards r_t [5]. In contrast, metrics such as the portfolio Sharpe ratio

$$\mathcal{R}_{\text{Sharpe}} = \frac{\mathbb{E}[R_{1:T}] - r_f}{\text{Std}[R_{1:T}]},$$

and drawdown-penalized objectives $\mathcal{R}_{\text{DD}} = \mathbb{E}[R_{1:T}] - \lambda \max_{t \le T} \text{DD}_t$ depend on the entire return path $R_{1:T}$. Applying REINFORCE to such non-additive functionals introduces bias in gradient estimation [43]. Consequently, the analysis centers on actor–critic algorithms (PPO, A2C, TD3, SAC), whose critic components can propagate these delayed signals through value function estimation while still supporting the continuous-allocation parameterization described above.

3.3 Selected Algorithms

Proximal Policy Optimization

The PPO agent parameterizes a Gaussian policy with state-dependent mean $\mu_\theta(o_t)$ and diagonal covariance $\Sigma_\theta(o_t)$ and trains alongside a value function $V_\psi(o_t)$ [44]. Policy updates maximize the clipped surrogate

objective

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t\Big[\min\Big(r_t(\theta)\,\hat{A}_t,\ \text{clip}\big(r_t(\theta), 1-\epsilon, 1+\epsilon\big)\,\hat{A}_t\Big)\Big],$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|o_t)}{\pi_{\theta_{\text{old}}}(a_t|o_t)}$ represents the probability ratio, and $\hat{A}_t$ denotes the generalized advantage estimate (GAE) [45] computed with discount factor γ and GAE parameter λ. The clipping mechanism with threshold ϵ prevents excessively large policy updates that could destabilize training [44]. Entropy regularization is retained to encourage exploration in the continuous allocation simplex, a standard technique for maintaining adequate exploration in policy gradient methods [46]. Training utilizes vectorized environments to provide batched trajectories, following established best practices for efficient on-policy learning [47].

Soft Actor-Critic

The SAC agent optimizes a stochastic policy $\pi_\theta(a_t \mid o_t)$ by maximizing the entropy-regularized objective

$$J(\pi_\theta) = \sum_t \mathbb{E}\big[Q_\phi(o_t, a_t) - \alpha \log \pi_\theta(a_t \mid o_t)\big],$$

where the temperature parameter α controls the trade-off between exploration and exploitation. SAC employs automatic entropy tuning to maintain a target entropy level, eliminating the need for manual temperature scheduling [48]. Twin Q-functions Q_{ϕ_1}, Q_{ϕ_2} mitigate overestimation bias through clipped double Q-learning [16], while target networks implement Polyak averaging with coefficient τ for stable value function updates [49]. The policy network outputs parameters of a squashed Gaussian distribution, with actions projected through a tanh function before the final softmax transformation to enforce allocation feasibility, ensuring that performance differences arise from the learning dynamics rather than environment handling.

Twin Delayed DDPG (TD3)

TD3 augments the deterministic policy gradient framework [50] with three critical modifications to address function approximation error [16]: (1) twin critics Q_{ϕ_1}, Q_{ϕ_2} using the minimum value for Bellman targets, (2) delayed policy updates occurring less frequently than critic updates, and (3) target policy smoothing. Actor updates employ the deterministic gradient

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\nabla_a Q_{\phi_1}(o_t, a)\big|_{a=\mu_\theta(o_t)} \nabla_\theta \mu_\theta(o_t)\right],$$

while target policy smoothing samples a noisy action $\tilde{a} = \mu_{\bar{\theta}}(o_{t+1}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ is clipped within bounds before computing the Bellman target. This regularization prevents exploitation of sharp peaks in the Q-function that could lead to brittle policies [16]. The shared softmax allocator keeps the action within the allocation simplex, aligning TD3 with the other selected algorithms and ensuring valid portfolio allocations.

3.4 Reward Function Design

Let $r_t^{(p)}$ denote the portfolio's simple return at step t, V_t the portfolio value, V_0 the initial balance, w_t the vector of risky-asset allocations, c_t the cash weight, Δw_t the aggregate turnover, $r_t^{(i)}$ the simple return of asset i, $\bar{r}_t = \frac{1}{N}\sum_i r_t^{(i)}$ the equal-weight benchmark, and $\sigma(r_t^{(i)})$ the cross-sectional volatility of constituent returns. Rolling statistics such as $\hat{\sigma}_{t,20}$ (20-step volatility), D_t (current drawdown), $D_t^{\max}$ (running max drawdown), $\tilde{r}_t$ (mean of the past ten $\bar{r}_t$ values), and $e_t = r_t^{(p)} - \bar{r}_t$ are tracked per episode, with $\varepsilon = 10^{-8}$ preventing division by zero. The sets $\mathcal{T}_t$ and $\mathcal{B}_t$ capture the top and bottom three performers at step t.

Absolute and Relative Return Objectives

These rewards target profit without explicit risk normalization:

Raw Portfolio Return	$R_t = r_t^{(p)}$.
Log Return	$R_t = \log \frac{V_t}{V_{t-1}}$.
Normalized Profit	$R_t = \dfrac{V_t - V_{t-1}}{V_0}$, scaling profit by starting capital.
Excess Return over Benchmark	$R_t = r_t^{(p)} - \bar{r}_t$, rewarding alpha versus the equal-weight Dow basket.

Risk-Adjusted Return Objectives

These objectives scale return by volatility measures to approximate Sharpe or information ratios:

Return over Rolling Volatility	$R_t = \dfrac{r_t^{(p)}}{\hat{\sigma}_{t,20} + \varepsilon}$, mimicking a 20-day Sharpe ratio.
Online Sharpe Ratio	$R_t = \dfrac{r_t^{(p)} - \mu_{t-1}}{\sigma_t + \varepsilon}$, with μ_t, σ_t updated via Welford's method.
Information Ratio	Using tracking errors $e_t = r_t^{(p)} - \bar{r}_t$, $R_t = \dfrac{\bar{e}_t}{\sigma_e + \varepsilon}$ where $\bar{e}_t, \sigma_e$ span the latest 60 e_t values.
Risk-Adjusted by Concentration	$R_t = \dfrac{r_t^{(p)}}{\sum_i w_{t,i}^2 + c_t^2 + \varepsilon}$, penalizing concentrated positions.

Downside Risk and Drawdown Control

These formulations emphasize tail-risk mitigation through asymmetric penalties:

Sortino-like Reward	$R_t = r_t^{(p)}$ if $r_t^{(p)} \geq 0$; otherwise $R_t = 2\, r_t^{(p)}$, doubling the weight on losses.
Return minus Drawdown Penalty	$R_t = r_t^{(p)} - 0.5\, D_t$, where $D_t = \frac{\max_{s \leq t} V_s - V_t}{\max_{s \leq t} V_s}$.
Calmar-like Reward	$R_t = \dfrac{r_t^{(p)}}{D_t^{\max} + 0.01}$, emphasizing sustained drawdown suppression.
CVaR Reward	$R_t = r_t^{(p)} - \lambda \max\big(0, -\mathrm{CVaR}_{\alpha,t}\big)$ with $\alpha = 0.05$ and $\lambda = 0.5$, using the worst 5% of the past 100 returns.

Cross-Sectional and Momentum Strategies

These rewards exploit relative performance signals across assets:

Momentum Exploitation	$R_t = r_t^{(p)} + 0.05 \left(\sum_{r_t^{(i)} > 0} w_{t,i} - \sum_{r_t^{(i)} < 0} w_{t,i} \right)$, favoring winners.
Tactical Allocation	With $\mathcal{T}_t / \mathcal{B}_t$ representing top/bottom three assets, $R_t = r_t^{(p)} + 0.03 \left(\sum_{i \in \mathcal{T}_t} w_{t,i} - \sum_{i \in \mathcal{B}_t} w_{t,i} \right)$.
Relative Strength	$R_t = r_t^{(p)} + 0.1 \sum_i w_{t,i} \big(r_t^{(i)} - \bar{r}_t \big)$, aligning weights with contemporaneous outperformers.

Diversification and Portfolio Structure

These objectives encourage specific allocation patterns:

Return plus Diversification Bonus	$R_t = r_t^{(p)} + 0.01 \dfrac{\mathcal{H}(w_t)}{\log N}$ with entropy $\mathcal{H}(w_t) = -\sum_i w_{t,i} \log(w_{t,i} + \varepsilon)$.
Trading Activity Bonus	$R_t = r_t^{(p)} + 0.02\,\Delta w_t$ if $\Delta w_t > 0.01$, else $R_t = r_t^{(p)} - 0.005$, promoting active rebalancing.
Turnover-Weighted Return	$R_t = r_t^{(p)} - 0.01$ when $\Delta w_t < 0.05$, else $R_t = r_t^{(p)} + 0.02\,\Delta w_t$.
Dynamic Rebalancing	With $r_t^{(\mathrm{eq})} = \bar{r}_t$, set $R_t = r_t^{(p)} + 0.05$ if $\Delta w_t > 0.01$ and $r_t^{(p)} > r_t^{(\mathrm{eq})}$, $R_t = r_t^{(p)} - 0.02$ if $\Delta w_t > 0.01$ but $r_t^{(p)} \le r_t^{(\mathrm{eq})}$, and $R_t = r_t^{(p)} - 0.01$ when $\Delta w_t \le 0.01$.

Regime-Dependent Objectives

These rewards adapt to market conditions through conditional logic:

Market Timing	If $\bar{r}_t < -0.01$, $R_t = r_t^{(p)} + 0.03\,c_t$; otherwise $R_t = r_t^{(p)}$, rewarding dry powder in drawdowns.
Regime-Adaptive Reward	High cross-sectional volatility ($\sigma(r_t^{(i)}) > 0.02$) yields $R_t = r_t^{(p)} + 0.03\,c_t$; calmer regimes switch to $R_t = r_t^{(p)} + 0.02\,\Delta w_t$.
Risk-On Risk-Off	Using $\tilde{r}_t = \frac{1}{10}\sum_{j=0}^{9} \bar{r}_{t-j}$, set $R_t = r_t^{(p)} + 0.02(1 - c_t)$ when $\tilde{r}_t > 0$ and $R_t = r_t^{(p)} + 0.02 c_t$ otherwise.

4 Experimental Setup

4.1 Dow Jones Index

The **Dow Jones Industrial Average (DJIA)** is one of the oldest and most referenced stock indices, first calculated in 1896 with 12 industrial stocks and now comprising 30 large U.S. "blue-chip" companies. It is a *price-weighted* index, meaning that a company's stock with a higher share price exerts greater influence on the index's value than lower-priced stocks [51]. This construction is unique compared to market capitalization-weighted indices like the S&P 500 and has been noted as an inherent limitation of the DJIA [51]. The DJIA's methodology ignores dividends and other corporate actions, focusing only on price changes, which leads to understatement of total returns over time. For example, Lin et al. estimate that if dividends were accounted for, the Dow would have reached over one million points by the end of 2019 rather than about 28,500 [52].

Despite these quirks, the DJIA remains a reliable barometer of U.S. equity market performance. It generally moves in tandem with broader indices, although the Dow's price-weighting can sometimes overstate blue-chip gains. Volatility-wise, the Dow tends to be slightly less volatile than indices like the Nasdaq or Russell 2000, which include more small-cap or tech-heavy stocks. Beta coefficients for the DJIA typically sit just below 1.0, suggesting lower relative volatility [51]. Tools like the DJIA Volatility Index (VXD) have been developed to measure the market's 30-day expected volatility based on Dow index options, analogous to the S&P 500's VIX.

In financial modeling and algorithmic trading, the DJIA is frequently employed as a benchmark due to its stability and historical depth. Its constituent stocks are commonly used to train and test quantitative trading strategies. Liu et al. developed FinRL, a deep reinforcement learning library, and demonstrated its trading algorithms on DJIA component stocks [18]. Algorithmic trading's influence on DJIA behavior has also been explored, showing both volatility-dampening and amplifying effects depending on market conditions [53].

Buy-and-Hold Baseline

To benchmark the RL agents against the underlying market regime, we construct an equal-weight buy-and-hold index from the 29 Dow constituents that span the full sample (all names in the trading universe except DOW, as DOW's existence does not span the entire period). Figure 1 plots the cumulative value of investing \$1 at the start of 2015 and carrying the basket un-hedged through the two evaluation windows.

The portfolio grows from 1.0 to 2.46 by 28 Dec 2022 (+146%) during the training phase, illustrating the strong large-cap momentum that any active method must beat. For the hold-out analysis the series is re-based to 1.0 immediately after applying the 20-trading-day warm-up to the test split (the split occurs on 29 Dec 2022, so test metrics start on 30 Jan 2023); from that reset point the passive basket advances to 1.23 by 31 Dec 2024 (+23%). Using the identical split (Jan 2015–28 Dec 2022 for training, 29 Dec 2022–31 Dec 2024 for testing with a 20-day warm-up) ensures that deterministic buy-and-hold returns and RL rollouts are directly comparable.

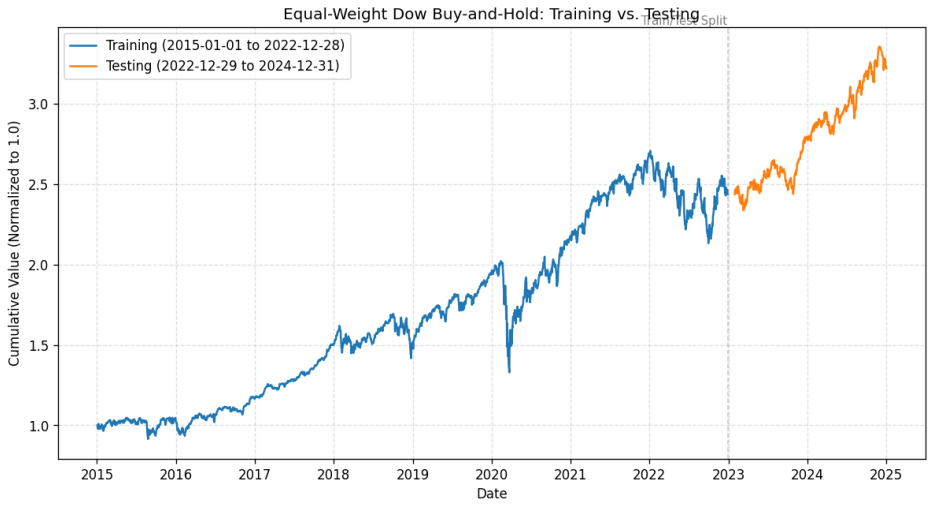


Figure 1: Equal-weight Buy and Hold Dow Jones Constituent Stocks

4.2 Data Pipeline

Historical OHLCV records (January 2015 through Dec 2024) are used. Timestamps are converted to UTC, missing values are forward-filled, and rows preceding the first complete observation are discarded. For each ticker the series is sorted chronologically and partitioned into disjoint training and testing windows using an 80/20 temporal split: the first 80% (January 2015–28 Dec 2022) supports training and on-policy evaluation, while the final 20% (29 Dec 2022–31 Dec 2024) is reserved for deterministic testing. The 20-trading-day warm-up is applied at the start of each split, so test-time evaluation begins on 30 Jan 2023 even though raw test prices start on 29 Dec 2022.

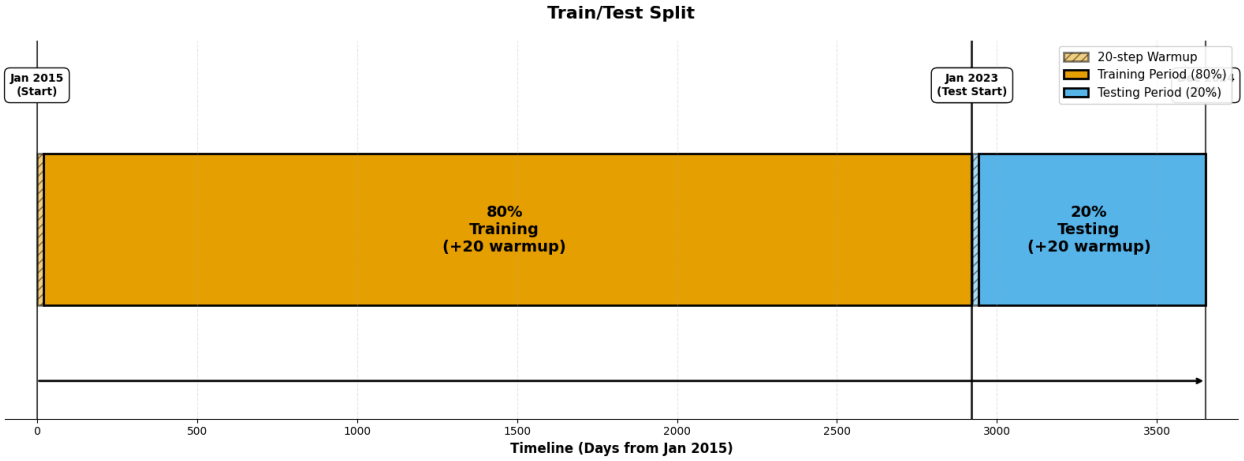


Figure 2: Train Test Split

The training segment constructs both learning and evaluation environments, whereas the hold-out segment supports deterministic rollouts for performance attribution against an equal-weight buy-and-hold benchmark whose cumulative return is computed as

$$\mathrm{BH}_{t+1} = \mathrm{BH}_t \left(1 + \frac{1}{N}\sum_{i=1}^{N} r_{t+1}^{(i)}\right), \qquad \mathrm{BH}_0 = V_0.$$

Warm-up periods of 20 trading days are applied consistently in both training and testing environments. Using the warm-up in training guards against undefined indicators early in each episode, while mirroring the same horizon in testing ensures that performance metrics align with the state features both agents experienced during learning. Experiments that omitted the warm-up on the test side produced optimistic first-step allocations driven by stale indicators; aligning the warm-up eliminates this bias and yields comparable baselines across splits.

4.3 Algorithm Configuration

We compare three actor-critic algorithms implemented via Stable-Baselines3. All agents share the same observation space and reward library. PPO trains for a maximum of 5×10^6 timesteps with evaluation checkpoints every 5×10^4 steps, while SAC and TD3 train for 2×10^6 timesteps with evaluations every 2×10^4 steps. All evaluations use five deterministic episodes under fixed random seeds. Early stopping terminates after ten consecutive evaluations without improvement in mean episodic return. Though different training timestamps were used, each reward-algorithm combination is able to converge within the schedule. Algorithm-specific hyperparameters are summarized in Table 1.

Table 1: Algorithm hyperparameters for PPO, SAC, and TD3 training.

Parameter	**PPO**	**SAC**	**TD3**
Learning rate	3×10^{-4}	3×10^{-4}	1×10^{-3}
Batch size	64	128	128
Replay buffer	—	2×10^5	2×10^5
Discount factor (γ)	0.99	0.99	0.99
Rollout horizon	2048 steps	—	—
Target update (τ)	—	0.02	0.02
Policy delay	—	—	2 steps
Learning starts	—	5,000	10,000

PPO employs an `MlpPolicy` with 2048-step rollout horizons and minibatch updates. SAC and TD3 utilize replay buffers for off-policy learning, with TD3 incorporating twin critics ($Q_{\theta_1}, Q_{\theta_2}$), delayed policy updates (every 2 critic iterations), and target policy smoothing ($\mathcal{N}(0, 0.2)$ noise clipped at 0.5) to mitigate overestimation bias. During evaluation, all algorithms execute deterministic policy rollouts to isolate learned behavior from exploration noise. Complete implementation details and training logs are available in our code repository.

5 Results and Analysis

5.1 Overall Performance Comparison

Algorithm Robustness

This section summarizes algorithm performance averaged across 22–23 reward variants and multiple random seeds. Detailed per-configuration results, including individual seed performance and convergence curves, are presented in Appendix A.

Statistical Significance. Pairwise Wilcoxon signed-rank tests confirm that both SAC and TD3 significantly outperform PPO across all metrics. SAC achieves a mean advantage of $23,071 in final portfolio value

Table 2: Average performance by algorithm

Algorithm	Final value ($)	Max drawdown (%)	Outperformance (%)	Total return (%)
PPO	104,373.71	-25.32	-18.84	4.38
SAC	127,638.17	-11.50	4.42	27.64
TD3	123,760.84	-10.41	0.54	23.76

over PPO ($p = 0.0032$) and +23.07% in outperformance ($p = 0.0032$), while also delivering substantially better drawdown control (+13.98 percentage points, $p < 0.001$). TD3 similarly dominates PPO with +$19,194 in terminal wealth ($p = 0.0115$), +19.20% outperformance ($p = 0.0115$), and +15.07 percentage points in drawdown mitigation ($p < 0.001$). In contrast, the SAC–TD3 comparison reveals no statistically significant differences: SAC's $3,877 advantage in final value ($p = 0.3209$) and +3.88% edge in outperformance ($p = 0.3209$) fall within natural variation across reward specifications. Full pairwise test results appear in Appendix B.

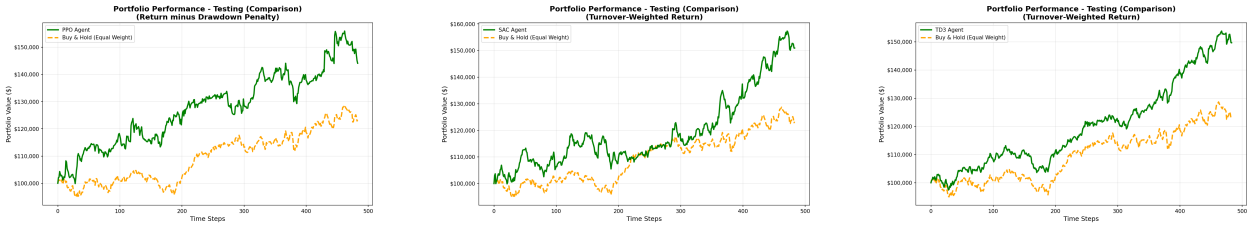
SAC (Superior) SAC not only dominates three of the global top-five portfolio values but also posts the tightest risk envelope: mean max drawdown is −11.5%. SAC's mean excess return of +4.4% is the highest among the evaluated agents, and granularly, 15 of SAC's 22 rewards finish ahead of the buy-and-hold benchmark. Even the laggards remain tightly clustered, with an interquartile range of terminal wealth spanning only $18k. This narrow dispersion confirms that practitioners can swap among turnover-, information-ratio-, or drawdown-aware objectives without destabilising the policy.

TD3 (Competitive) TD3 trails SAC only slightly on terminal wealth (mean $123,761) yet delivers marginally better average drawdown control (−10.4%). Outperformance averages +0.54%, signalling reduced but still positive alpha. Ideally, TD3 acts as a "smoother floor" than PPO; even its bottom performers keep portfolios near $106k–$107k. Furthermore, TD3 is less sensitive to reward misspecification: swapping from pure return to turnover-aware shaping only shifts terminal wealth by $11k on average, making it a pragmatic choice where deterministic policies are preferred.

PPO (Underperformer) In contrast, PPO's averages deteriorate across every metric: mean final wealth of $104,373, mean drawdown of −25.3%, and mean relative performance of −18.8%. Only six of PPO's 23 rewards beat buy-and-hold, and the dispersion is extreme. Consequently, PPO is best viewed as a stress-test baseline: it exposes reward functions that require excessive exploration noise or are highly sensitive to policy stochasticity, but it is ill-suited for production deployment without significant additional engineering.

Best Cases by Algorithm

The top-performing rewards for each algorithm are listed in Table 3. PPO's strongest configuration, *Return minus Drawdown Penalty*, finishes with $144,117 and surpasses buy-and-hold by 20.90%, owing to its explicit coupling of upside capture and drawdown control. Information-theoretic shaping (*Information Ratio Reward*) also yields competitive gains with a shallow −10.24% drawdown. SAC consistently benefits from turnover-aware and information-theoretic objectives: the *Turnover-Weighted Return* reward produces the highest excess performance at 27.71% and a terminal value of $150,926, while *Information Ratio Reward* and *Online Sharpe Ratio* offer similar upside with sub-−12% drawdowns.



PPO best returns **SAC best returns** **TD3 best returns**

Figure 3: Best return under each algorithm

Table 3: Rewards with the highest outperformance for each algorithm.

Algorithm	Reward	Final value ($)	Outperformance (%)
PPO	Return minus Drawdown Penalty	144,117.30	20.90
PPO	Information Ratio Reward	141,475.30	18.26
PPO	Return over Rolling Volatility	129,627.43	6.41
SAC	Turnover-Weighted Return	150,925.79	27.71
SAC	Information Ratio Reward	148,233.36	25.02
SAC	Online Sharpe Ratio	145,465.77	22.25
TD3	Turnover-Weighted Return	149,703.28	26.49
TD3	Normalized Profit	140,470.36	17.25
TD3	Raw Portfolio Return	138,344.38	15.13

Benchmarking Against State-of-the-Art

Our best-performing algorithm–reward combination, SAC with *Turnover-Weighted Return*, achieves a total return of 50.04% (buy-and-hold return of 22.33% plus outperformance of 27.71%) over the testing period from 30 January 2023 to 31 December 2024. Annualized over this approximately 1.92-year period, this translates to an annual return of 23.1%, computed using the compound annual growth rate formula.

Table 4: Comparison of annual returns with related works in portfolio management.

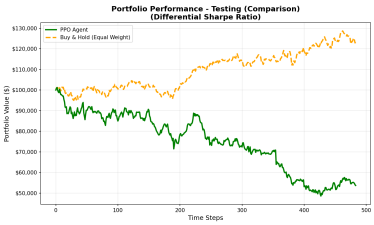
Study	Buy and Hold Benchmark (%)	Annual Return (%)	Outperformance (%)
Liu et al. [14]	16.40	25.87	9.47
Best model of this work	**11.48**	**23.1**	**11.62**
Yang et al. [54]	7.8	13.0	5.2
Wong & Liu [55]	3.08	16.24	13.16

It is important to note that direct comparison of absolute annual returns across these studies requires careful interpretation, as each work evaluates performance over different testing periods with distinct market conditions. The substantial variation in benchmark returns–ranging from 3.08% to 16.40%–reflects the differing time horizons and market environments encountered in each study. Liu et al. [14] employed a DDPG-based approach, Yang et al. [54] utilized an ensemble strategy combining multiple reinforcement learning algorithms, and Wong & Liu [55] applied a PPO-based method. This variation in benchmark returns underscores that portfolio management algorithms are inherently tested against the prevailing market dynamics of their respective evaluation periods.

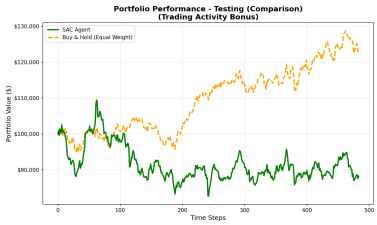
When evaluated relative to their corresponding benchmarks, our SAC-based approach achieves an outperformance of 11.62%, which is competitive with the outperformance achieved by Liu et al. (9.47%) and substantially exceeds that of Yang et al. (5.2%). Wong & Liu demonstrate the highest relative outperformance (13.16%), though this is achieved during a period with a notably lower benchmark return. These results indicate that our method delivers strong risk-adjusted performance and demonstrates robust capability in generating alpha across different market regimes.

Worst Cases by Algorithm

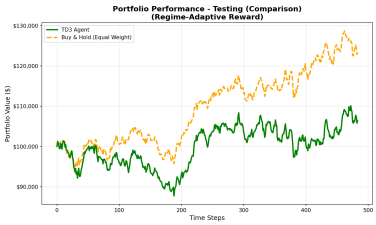
The lowest-ranked rewards (Table 5) reveal a stark contrast between algorithms. PPO suffers substantial capital erosion under *Online Sharpe Ratio*, which finishes at $53,708 and trails buy-and-hold by 69.51%, illustrating the instability of online Sharpe estimation in this on-policy setting. SAC's worst case, *Trading Activity Bonus*, remains at $88,225 with a -34.99% deficit yet still maintains shallower drawdowns relative to PPO's failures. TD3 sits between the two extremes: its weakest reward, *Regime-Adaptive Reward*, still preserves $106,372 with a -16.84% shortfall, underscoring the added stability TD3 gains from deterministic policy updates and twin critics.



PPO worst returns



SAC worst returns



TD3 worst returns

Figure 4: Worst Return Under Each Algorithm

Table 5: Rewards with the lowest outperformance for each algorithm.

Algorithm	Reward	Final value ($)	Outperformance (%)
PPO	Online Sharpe Ratio	53,707.65	-69.51
PPO	Excess Return over Benchmark	64,980.07	-58.24
PPO	Risk-On Risk-Off Reward	66,558.00	-56.66
SAC	Trading Activity Bonus	88,225.18	-34.99
SAC	Risk-Adjusted by Concentration	100,561.41	-22.66
SAC	Normalized Profit	111,950.49	-11.27
TD3	Regime-Adaptive Reward	106,372.14	-16.84
TD3	Risk-On Risk-Off Reward	107,322.91	-15.89
TD3	Relative Strength Reward	110,641.97	-12.57

Overall, SAC continues to dominate, led by *Turnover-Weighted Return* at $150,925.79 with 27.71% excess return and an −11.44% drawdown. TD3 now closes much of the gap—its best reward also exceeds $149,000 with a 26.49% edge—yet still trails SAC on downside containment. PPO remains the laggard: only two rewards exceed $140,000, and most stay negative versus buy-and-hold. On the downside, PPO's worst case erodes capital to $53,708 and lags by −69.51%, whereas TD3's weakest reward still preserves over $105,000 in value and SAC's worst retains $88,225. These updated results reinforce the benefits of off-policy methods with entropy regularisation and replay, with SAC setting the pace and TD3 providing a competitive, low-drawdown alternative.

5.2 Reward Function Impact Analysis

Tier 1: Top Performers

Averaging across algorithms pinpoints the reward functions that most reliably balance return and risk (Table 6). Turnover-aware and information-theoretic objectives provide $133k–$137k mean terminal wealth with drawdowns around −9% to −16% and double-digit excess returns. *Online Sharpe Ratio* sits at the bottom of this tier because PPO drags the average downward, reinforcing the need to pair each reward with a compatible algorithm.

Table 6: Tier-1 reward functions averaged across algorithms.

Reward Function	Avg Portfolio Value ($)	Avg Drawdown (%)	Avg Outperformance (%)
Turnover-Weighted Return	136,623.57	-16.06	13.41
Information Ratio Reward	134,888.14	-9.33	11.67
Return minus Drawdown Penalty	132,289.84	-9.78	9.07
Online Sharpe Ratio	103,344.77	-25.98	-19.87

Tier 2: Solid Performers

Momentum Exploitation Reward averages \$126k with manageable drawdown (−12.6%) and low variance across algorithms, making it a dependable baseline. *Return over Rolling Volatility* lands between the true information-ratio objectives and return-only metrics, highlighting that penalising downside but ignoring turnover leaves some performance untapped.

Tier 3: Underperformers

Naively maximising returns introduces structural weaknesses (Table 7). *Raw Portfolio Return* and *Log Return* demand more risk for less reward, *Market Timing Reward* underperforms by 15–27%, and *Excess Return over Benchmark* suffers the worst drawdowns. These rewards lack explicit drawdown or turnover controls and therefore allow volatile allocations to dominate learning.

Table 7: Tier-3 reward weaknesses aggregated across algorithms.

Reward Function	Avg Drawdown (%)	Avg Outperformance (%)	Key Weakness
Raw Portfolio Return	-16.20	-0.85	Fails to penalise tail risk; high variance outcomes.
Log Return	-17.11	-14.73	Large downside tails with minimal alpha.
Market Timing Reward	-18.76	-15.24	Severe underperformance; timing signals unstable.
Excess Return over Benchmark	-20.76	-9.05	Largest average drawdowns; high dispersion.

Drawdown-Penalised Rewards

Both *Return minus Drawdown Penalty* and the Calmar-like objective explicitly incorporate drawdown terms, yet their realised drawdowns in the testing period remain in the same −10% to −15% band as the top turnover- and information-ratio rewards. *Return minus Drawdown Penalty* averages −9.78% drawdown (PPO −10.28%, SAC −9.56%, TD3 −9.50%) with \$132.3k terminal wealth, while the Calmar-like reward averages −14.89% drawdown, \$126.8k wealth, and +3.6% excess return across the same algorithms. SAC captures most of the upside in both cases (Calmar drawdown −13.4%, +15.5% excess, 200,150 steps), whereas PPO requires 1.7 million timesteps for only 5.4% excess and TD3 still posts −10.2% outperformance despite matching SAC's 200,150–200,313-step budget. These results indicate that adding drawdown penalties to the reward does not materially reduce drawdown during out-of-sample testing.

5.3 Risk-Return Characteristics

Drawdown Frontier

The updated drawdown frontier (final value vs. max drawdown) shows how tightly SAC clusters along the low-risk, high-wealth edge: eight of the fifteen runs above \$130k with drawdowns milder than −12% are SAC, five are TD3, and only two belong to PPO. SAC's *Information Ratio Reward* remains the cleanest Sharpe-style compromise at \$148,233 with a −8.0% trough, and even the entropy-heavy *Turnover-Weighted Return* configuration sustains \$150,926 after just an −11.4% slide. TD3 supplies the next-best risk control—seven of its twenty-two runs post drawdowns better than −9%—and its *Turnover-Weighted Return* variant finishes at \$149,703 with −8.44% drawdown. PPO seldom occupies the frontier: only *Information Ratio Reward* and *Return minus Drawdown Penalty* surpass \$120k while staying inside a −15% drawdown bound, leaving most PPO points higher (riskier) on the chart. Tail-sensitive shaping behaves predictably: *CVaR Reward* for

SAC ($126,038, −9.26%) and TD3 ($116,701, −11.09%) reins in losses but trims upside relative to return-centric rewards. *Return minus Drawdown Penalty* remains a pragmatic middle ground, pairing $144,117 and −10.3% in PPO with $123,158 and −9.6% in SAC. Finally, the revised *Risk-Adjusted by Concentration* runs now exhibit normal drawdowns (e.g., TD3 at $117,433 with −9.2%), confirming that the concentration clamp eliminated the previous all-cash pathology.

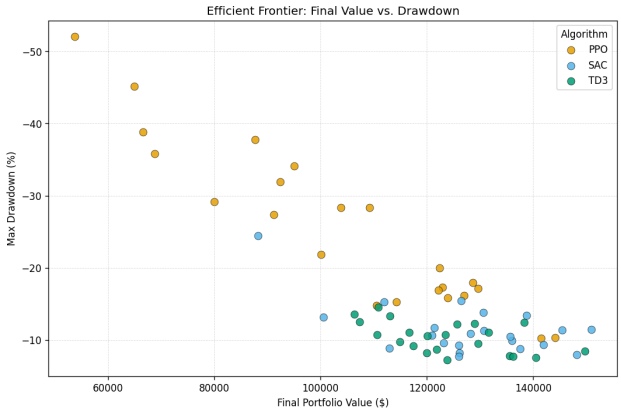


Figure 5: Final Value vs Drawdown

Cash Allocation Findings

PPO still exhibits the highest propensity to retreat into cash: *Risk-On Risk-Off Reward* (41.1%), *Sortino-like Reward* (29.1%), *Market Timing Reward* (26.9%), *Calmar-like Reward* (24.6%), and *Return minus Drawdown Penalty* (19.8%) all lead to cash allocations exceeding one-fifth of the portfolio. SAC's strongest cash signal is substantially lower at 21.1% (*Market Timing Reward*), and most of its reward formulations keep cash usage below 15%. TD3 remains the most fully invested: even its cash-heaviest configurations (*Market Timing*, *Online Sharpe Ratio*) remain close to 10%, while *Risk-Adjusted by Concentration* holds 8.5%.

Turnover-heavy rewards almost eliminate idle capital for every agent. The *Trading Activity Bonus* pushes cash usage below 0.1% for PPO, 0.5% for SAC, and 7.8% for TD3, while *Turnover-Weighted Return* leaves PPO with only 1.0% cash (SAC: 7.6%, TD3: 9.2%). Regime-switching objectives also minimise unused capital: all *Regime-Adaptive Reward* runs remain under 3.7% cash, with TD3 effectively eliminating it (0.058%). Collectively, these diagnostics show that PPO's weakest financial outcomes arise under rewards that implicitly encourage policies to sit in cash, whereas SAC and TD3 maintain consistently higher market exposure and thus greater potential for compounding returns.

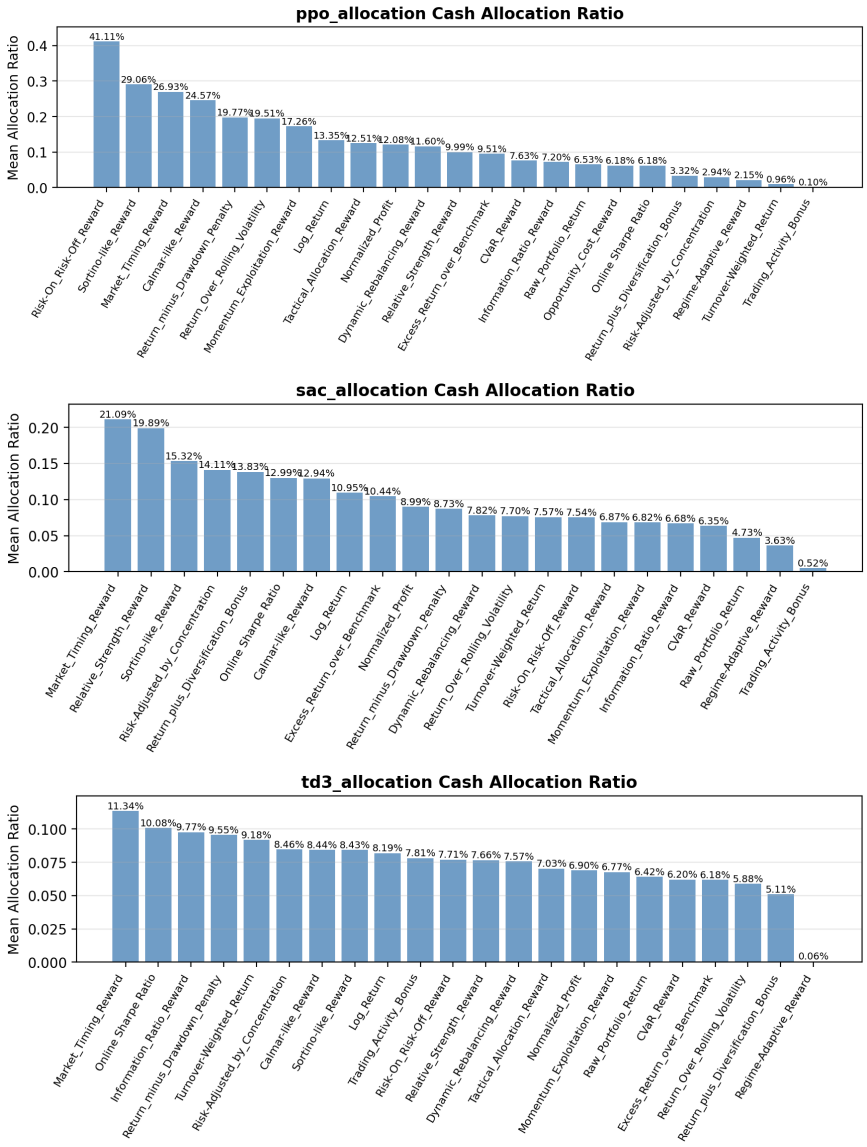


Figure 6: Mean Cash Allocation Ratio

5.4 Sample Efficiency and Convergence

Off-policy agents converge quickly: both SAC and TD3 show median training timesteps of 200,150 across all effective rewards, with only SAC's *Return over Rolling Volatility* (780,000 steps) and TD3's *Online Sharpe Ratio*/*Return over Rolling Volatility* (420,000 steps) requiring additional rollouts. PPO, by contrast, has a 1.775 million-step median and spans 1.55–2.5 million steps, i.e., roughly 7–12× more data for comparable results. Some PPO rewards (e.g., *Turnover-Weighted Return* at 2.5 million steps) still finish with double-digit underperformance, highlighting poor sample efficiency. Reward-specific costs further emphasise robustness: SAC keeps all but one reward at 200,150 steps, and TD3 only doubles its budget for the most variance-sensitive objectives. This stability reduces hyperparameter retuning overhead and simplifies experimentation planning.

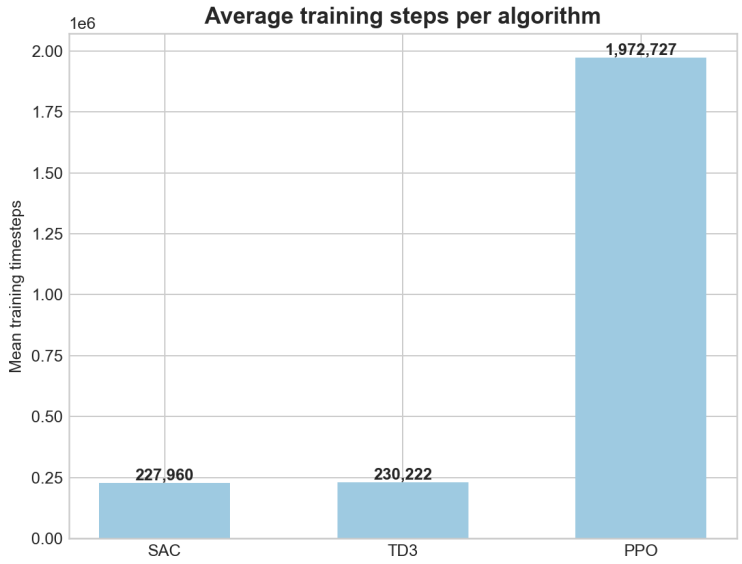


Figure 7: Comparison of sample efficiency across different reinforcement learning algorithms (SAC, TD3, PPO) and various reward functions.

Sharpe-Linked Convergence Patterns

Sharpe-oriented rewards (*Return over Rolling Volatility*, *Online Sharpe Ratio*) are the only objectives that consistently slow down the off-policy agents. SAC averages 227,960 training steps across all rewards in `results.csv`, yet its Sharpe variants require 780,000 and 231,175 steps (mean 505,588), more than 2.2 *times* the SAC-wide average. TD3 exhibits the same pattern: its overall mean is 230,222 steps, but both Sharpe rewards plateau only after 420,000 steps (1.8 *times* the TD3 mean). The data therefore confirm that ratio-style rewards are the longest-to-converge cases for both SAC and TD3, despite every other reward finishing near 200,150 steps. PPO breaks this trend; while its global mean is 1,972,727 steps, the Sharpe rewards converge earlier at 1,550,000 steps and are eclipsed by several other PPO objectives (e.g., *Turnover-Weighted Return* at 2,500,000 steps or *Regime-Adaptive Reward* at 2,450,000 steps). These figures, and the visual comparison in Figure 8, highlight that the convergence penalty of Sharpe-centric training is specific to the off-policy setups, not an inherent property of the reward when used with PPO.

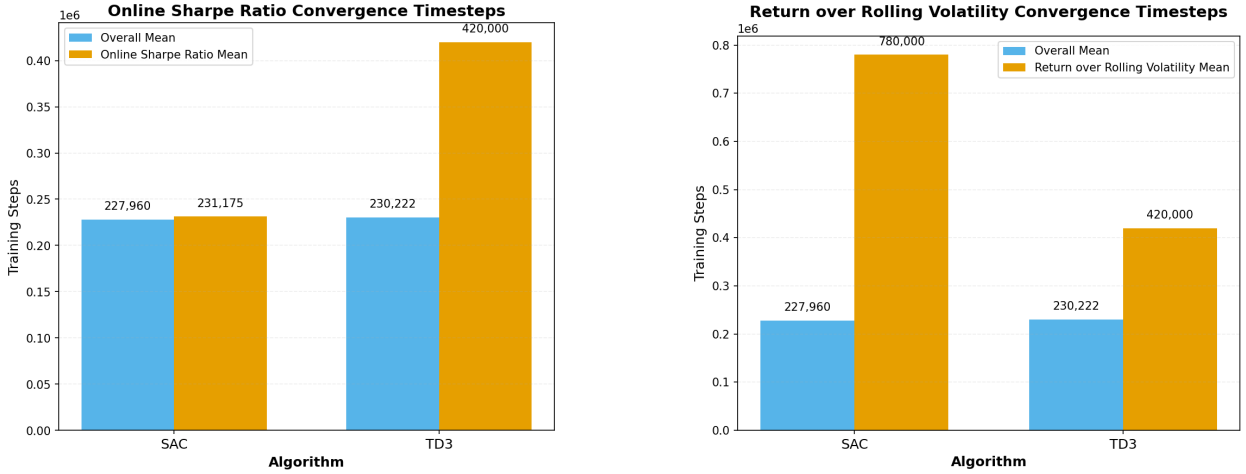


Figure 8: Training-step comparison for Sharpe rewards versus the overall mean (SAC and TD3 only).

Equation (1) captures the rolling Sharpe-style target: maximise the excess return $\bar{r}_t - r_f$ while simultaneously damping the trailing volatility $\widehat{\sigma}_t$. Because both terms are estimated on finite windows, small perturbations in $\widehat{\sigma}_t$ can flip the reward sign, inflating gradient variance just as the critic tries to fit a smooth value surface. Off-policy replay aggravates the problem—transitions collected under older policies are reassessed with a volatility window computed from the latest policy, so the reward associated with a stored sample drifts over time. This non-stationarity violates the assumptions SAC and TD3 make when learning from replay and forces them to accumulate substantially more timesteps before the volatility estimator stabilises. PPO's on-policy sampling keeps the reward window and trajectory distribution aligned, which is why the same Sharpe-linked objectives do not slow PPO despite penalising SAC and TD3.

$$\widehat{S}_t = \frac{\bar{r}_t - r_f}{\sqrt{\widehat{\sigma}_t^2 + \varepsilon}} \tag{1}$$

5.5 Limitations and Future Work

- **Universe bias.** The study trades only the Dow Jones 30, a blue-chip index that already exhibits strong buy-and-hold performance. This raises the hurdle for active strategies and tilts the environment against risk-averse rewards that favour holding cash, potentially understating their value in broader or more volatile universes.

  Further improvements: Extend experiments to include diverse asset classes such as small-cap equities, emerging markets, sector ETFs, fixed income, commodities, and cryptocurrencies. Test across multiple market regimes (bull, bear, sideways, high-volatility) to identify regime-dependent reward efficacy and validate whether cash-holding strategies demonstrate superior risk-adjusted returns in more turbulent environments.

- **Trading realism.** The backtests assume zero transaction costs, perfect fills at the close, unlimited liquidity, and no regulatory constraints (short-sale bans, margin, position limits). Real markets impose commissions, bid-ask spreads, slippage, finite depth, and compliance rules that would erode returns and alter optimal policies.

  Further improvements: Implement realistic cost structures including proportional transaction costs ($c \cdot |\Delta w_i|$), quadratic market impact models ($\kappa \cdot \Delta w_i^2$) calibrated to actual order book data, and bid-ask spread penalties. Incorporate volume-dependent liquidity constraints and regulatory position limits. Develop a production-ready pipeline with real-time data feeds, broker API integration, latency modelling, and paper trading validation before live deployment.

- **Reproducibility.** Although `torch.use_deterministic_algorithms(True)` and fixed seeds are set, PyTorch GPU operations can remain non-deterministic, environment and library versions (PyTorch, Stable Baselines3, Gym) affect results, and some vectorised environment steps may bypass the global seed. Exact replication therefore requires replicating the full software stack and may still show small numerical differences.

  Further improvements: Provide containerised environments (Docker, Singularity) with pinned dependency versions and document the complete software stack including CUDA/cuDNN versions. Publish pre-trained model checkpoints and training logs alongside code. Conduct sensitivity analysis across multiple random seeds to report mean performance and confidence intervals rather than point estimates, enhancing statistical robustness and transparency of results.

6 Conclusion

Across twenty-two reward formulations and three policy-gradient families, the experiments show that reward shaping dictates performance more than algorithm choice. Off-policy SAC and TD3 consistently compound capital while staying invested, whereas PPO's poorer sample efficiency and tendency to hide in cash translate into both lower terminal wealth and higher opportunity cost. A practical implication for portfolio teams is that the reward function should be calibrated to the desk's mandate (return seeking, drawdown control, or turnover governance) before expending iterations on architectural tweaks.

Table 8 highlights the most actionable combinations for common objectives. TD3 paired with the Turnover-Weighted Return reward delivers the largest terminal value without sacrificing double-digit drawdown control, while SAC combined with the Information Ratio reward dominates both risk-adjusted and robustness lenses. Investors prioritising downside preservation can still look to TD3 with the Return plus Diversification Bonus, which keeps peak-to-trough losses near -7% yet remains marginally ahead of the benchmark.

Table 8: Best algorithm–reward pairings for common objectives.

Objective	Best Choice	Portfolio Value ($)	Drawdown (%)	Outperformance (%)
Max return	TD3 + Turnover-Weighted Return	149,703.28	-8.44	26.49
Best risk-adjusted	SAC + Information Ratio Reward	148,233.36	-8.00	25.02
Lowest drawdown	TD3 + Return plus Diversification Bonus	123,831.12	-7.23	0.61
Most robust	SAC + Information Ratio Reward	148,233.36	-8.00	25.02

Beyond headline metrics, three operational findings stand out. First, SAC and TD3 converge within roughly 200,000 timesteps for nearly every reward, so rolling out broad reward sweeps is computationally feasible; PPO, by contrast, needs 1.6–2.5 million steps, making it ill-suited for rapid research loops. Second, the cash-allocation heatmaps show that underperforming PPO rewards are those that retreat into cash, whereas SAC and TD3 keep average idle balances below 15%, reinforcing that capital deployment rather than reward variance drives the edge. Third, Sharpe-style objectives uniquely slow down off-policy agents and demand longer training horizons, so desks adopting them should budget extra simulation time or warm-start from neighbouring rewards.

Taken together, the study suggests a pragmatic playbook: begin with SAC or TD3, choose a reward that encodes the desk's tolerance for drawdown and turnover, monitor cash usage as a diagnostic, and reserve high-ratio objectives for scenarios where additional compute is justified. Coupled with the limitations noted above—especially universe coverage, market microstructure realism, and reproducibility—these conclusions clarify both where the current stack excels and which extensions (broader assets, cost-aware execution, multi-seed reporting) are most likely to unlock incremental alpha.

A Full Results Table

Tables 9–11 reproduce every algorithm–reward combination captured in `results.csv`, providing the exact terminal portfolio values, drawdowns, relative outperformance, and the number of training timesteps required for each experiment.

Table 9: PPO training outcomes sourced from `results.csv`.

Algorithm	Reward	Final value ($)	Max drawdown (%)	Training steps
PPO	CVaR Reward	122,957.21	-17.34	1,550,000
PPO	Calmar-like Reward	128,619.42	-17.97	1,700,000
PPO	Online Sharpe Ratio	53,707.65	-52.03	1,550,000
PPO	Dynamic Rebalancing Reward	114,225.52	-15.26	1,550,000
PPO	Excess Return over Benchmark	64,980.07	-45.16	1,750,000
PPO	Information Ratio Reward	141,475.30	-10.24	1,550,000
PPO	Log Return	92,412.72	-31.88	2,150,000
PPO	Market Timing Reward	68,826.86	-35.83	2,500,000
PPO	Momentum Exploitation Reward	126,999.20	-16.16	1,950,000
PPO	Normalized Profit	110,489.77	-14.81	1,850,000
PPO	Raw Portfolio Return	91,241.05	-27.34	2,100,000
PPO	Regime-Adaptive Reward	122,363.30	-19.95	2,450,000
PPO	Relative Strength Reward	87,726.90	-37.77	1,750,000
PPO	Return minus Drawdown Penalty	144,117.30	-10.28	1,550,000
PPO	Return plus Diversification Bonus	122,209.75	-16.90	1,550,000
PPO	Risk-Adjusted by Concentration	123,914.57	-15.85	1,550,000
PPO	Risk-On Risk-Off Reward	66,558.00	-38.77	2,050,000
PPO	Return over Rolling Volatility	129,627.43	-17.17	1,550,000
PPO	Sortino-like Reward	79,947.69	-29.11	1,800,000
PPO	Tactical Allocation Reward	95,026.08	-34.09	2,200,000
PPO	Trading Activity Bonus	103,800.93	-28.33	2,150,000
PPO	Turnover-Weighted Return	109,241.65	-28.31	2,500,000

Table 10: SAC training outcomes sourced from `results.csv`.

Algorithm	Reward	Final value ($)	Max drawdown (%)	Training steps
SAC	CVaR Reward	126,038.12	-9.26	200,150
SAC	Calmar-like Reward	138,742.34	-13.40	200,150
SAC	Online Sharpe Ratio	145,465.77	-11.38	231,175
SAC	Dynamic Rebalancing Reward	120,994.24	-10.68	200,150
SAC	Excess Return over Benchmark	141,933.28	-9.32	200,150
SAC	Information Ratio Reward	148,233.36	-8.00	201,086
SAC	Log Return	112,903.54	-8.89	200,150
SAC	Market Timing Reward	126,155.98	-8.20	200,150
SAC	Momentum Exploitation Reward	128,249.19	-10.88	200,150
SAC	Normalized Profit	111,950.49	-15.26	200,150
SAC	Raw Portfolio Return	137,502.40	-8.78	200,150
SAC	Regime-Adaptive Reward	130,656.96	-13.82	200,150
SAC	Relative Strength Reward	121,412.34	-11.70	200,150
SAC	Return minus Drawdown Penalty	123,158.24	-9.56	200,150

(continued on next page)

(continued from previous page)

Algorithm	Reward	Final value ($)	Max drawdown (%)	Training steps
SAC	Return plus Diversification Bonus	126,039.79	-7.75	200,150
SAC	Risk-Adjusted by Concentration	100,561.41	-13.12	200,150
SAC	Risk-On Risk-Off Reward	135,974.56	-9.91	200,150
SAC	Return over Rolling Volatility	126,488.64	-15.40	780,000
SAC	Sortino-like Reward	130,751.18	-11.30	200,150
SAC	Tactical Allocation Reward	135,677.01	-10.47	200,150
SAC	Trading Activity Bonus	88,225.18	-24.43	200,150
SAC	Turnover-Weighted Return	150,925.79	-11.44	200,150

Table 11: TD3 training outcomes sourced from `results.csv`.

Algorithm	Reward	Final value ($)	Max drawdown (%)	Training steps
TD3	CVaR Reward	116,700.77	-11.09	200,150
TD3	Calmar-like Reward	113,055.03	-13.30	200,313
TD3	Online Sharpe Ratio	110,860.90	-14.52	420,000
TD3	Dynamic Rebalancing Reward	121,791.25	-8.73	200,150
TD3	Excess Return over Benchmark	135,601.51	-7.81	200,150
TD3	Information Ratio Reward	114,955.75	-9.74	201,666
TD3	Log Return	120,129.53	-10.57	200,150
TD3	Market Timing Reward	128,943.83	-12.25	200,150
TD3	Momentum Exploitation Reward	123,534.43	-10.76	200,150
TD3	Normalized Profit	140,470.36	-7.59	200,499
TD3	Raw Portfolio Return	138,344.38	-12.47	200,150
TD3	Regime-Adaptive Reward	106,372.14	-13.54	200,150
TD3	Relative Strength Reward	110,641.97	-10.76	200,150
TD3	Return minus Drawdown Penalty	129,593.98	-9.50	200,150
TD3	Return plus Diversification Bonus	123,831.12	-7.23	200,150
TD3	Risk-Adjusted by Concentration	117,432.76	-9.20	201,143
TD3	Risk-On Risk-Off Reward	107,322.91	-12.52	200,150
TD3	Return over Rolling Volatility	125,648.77	-12.15	420,000
TD3	Sortino-like Reward	136,254.62	-7.70	200,150
TD3	Tactical Allocation Reward	131,593.26	-11.04	200,150
TD3	Trading Activity Bonus	119,956.03	-8.20	200,150
TD3	Turnover-Weighted Return	149,703.28	-8.44	200,150

B Statistical Significance Tests

To rigorously evaluate the performance differences between the algorithms (SAC, TD3, and PPO), we conducted paired statistical tests across all reward functions. Given that financial return distributions and algorithmic performance metrics often deviate from normality, we employed the **Wilcoxon Signed-Rank Test** as a robust non-parametric alternative to the paired t-test.

Table 12 summarizes the results for Final Portfolio Value, Outperformance (%), and Maximum Drawdown (%). Statistical significance is determined at the $\alpha = 0.05$ level.

Table 12: Paired Statistical Tests (Wilcoxon Signed-Rank) comparing algorithm performance across all reward functions. Significance is denoted by bold p-values ($p < 0.05$).

Comparison	**Metric**	**Mean Diff.**	p**-value**	**Result**
SAC vs PPO	Final Portfolio Value	$+23,071$	**0.0032**	Significant
	Outperformance (%)	+23.07%	**0.0032**	Significant
	Max Drawdown (%)	+13.98	$<$ **0.001**	Significant
TD3 vs PPO	Final Portfolio Value	$+19,194$	**0.0115**	Significant
	Outperformance (%)	+19.20%	**0.0115**	Significant
	Max Drawdown (%)	+15.07	$<$ **0.001**	Significant
SAC vs TD3	Final Portfolio Value	$+3,877$	0.3209	Not Sig.
	Outperformance (%)	+3.88%	0.3209	Not Sig.
	Max Drawdown (%)	-1.08	0.4434	Not Sig.

The results demonstrate that both off-policy algorithms (SAC and TD3) significantly outperform the on-policy baseline (PPO) across all key metrics ($p < 0.05$). Specifically, SAC and TD3 achieved significantly higher final portfolio values and reduced maximum drawdowns compared to PPO. However, the performance difference between SAC and TD3 was not statistically significant, suggesting comparable effectiveness for this portfolio allocation task.

References

[1] T. Hendershott, C. M. Jones, and A. J. Menkveld, "Does algorithmic trading improve liquidity?" *The Journal of Finance*, vol. 66, no. 1, pp. 1–33, 2011.

[2] M. López de Prado, *Advances in Financial Machine Learning*. Hoboken, NJ: John Wiley & Sons, 2018.

[3] D. Krueger, E. Caballero, J.-H. Jacobsen, A. Zhang, J. Binas, A. Durand, H. Larochelle, and S. Lacoste-Julien, "Out-of-distribution generalization via risk extrapolation (rex)," in *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021. [Online]. Available: https://www.researchgate.net/publication/339642303_Out-of-Distribution_Generalization_via_Risk_Extrapolation_REx

[4] Statistics4U, "Extrapolation in artificial neural networks," http://www.statistics4u.com/fundstat_eng/cc_ann_extrapolation.html, accessed: 2025-11-15.

[5] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA: MIT Press, 2018.

[6] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[7] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[8] Z. Jiang, D. Xu, and J. Liang, "A deep reinforcement learning framework for the financial portfolio management problem," *arXiv preprint arXiv:1706.10059*, 2017, available at arXiv:1706.10059v2.

[9] H. Buehler, L. Gonon, J. Teichmann, and B. Wood, "Deep hedging," *Quantitative Finance*, vol. 19, no. 8, pp. 1271–1291, 2019.

[10] N. Pippas, E. A. Ludvig, and C. Turkay, "The evolution of reinforcement learning in quantitative finance: A survey," *arXiv preprint arXiv:2408.10932v3*, 2025, version v3. [Online]. Available: https://arxiv.org/abs/2408.10932v3

[11] H. Chen and C. Xiong, "Adaptive asset allocation with hierarchical reinforcement learning: A multi-class investment strategy," in *Technical Innovation in Financial Economics: Advanced Theory, Practice, and Policy*. IGI Global, 2026, chapter 6, 26 pages. [Online]. Available: https://www.igi-global.com/chapter/adaptive-asset-allocation-with-hierarchical-reinforcement-learning/389239

[12] Y. Bai, Y. Gao, R. Wan, S. Zhang, and R. Song, "A review of reinforcement learning in financial applications," *arXiv preprint arXiv:2411.12746v1*, 2024, available at arXiv:2411.12746v1.

[13] J. E. Moody and M. Saffell, "Reinforcement learning for trading," in *Advances in Neural Information Processing Systems (NeurIPS)*, 1999, pp. 917–923. [Online]. Available: https://papers.nips.cc/paper/1551-reinforcement-learning-for-trading

[14] X.-Y. Liu, Z. Xiong, S. Zhong, H. B. Yang, and A. Walid, "Practical deep reinforcement learning approach for stock trading," *arXiv preprint arXiv:1811.07522*, 2018, available at arXiv:1811.07522.

[15] Z. Gao, Y. Chen, and G. Sun, "Applying deep q-learning with dueling networks and prioritized replay in portfolio optimization," *arXiv preprint arXiv:2006.01813*, 2020.

[16] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1587–1596.

[17] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[18] X.-Y. Liu, H. Yang, Q. Chen, R. Zhang, L. Yang, B. Xiao, and C. D. Wang, "Finrl: A deep reinforcement learning library for automated stock trading in quantitative finance," *arXiv preprint*, 2020, arXiv:2011.09607 [q-fin.TR].

[19] H. Yang, J. Zhou, and X. Huang, "Deep reinforcement learning for automated stock trading: An ensemble strategy," *arXiv preprint arXiv:2004.06627*, 2020.

[20] J. Wang, L. Liu, and J. Hao, "Hierarchical reinforced portfolio management with latent market conditions," *Neurocomputing*, vol. 489, pp. 1–15, 2022.

[21] C. Ma, Y. Ding, and J. Li, "A multi-agent deep reinforcement learning approach for stock trading," *Expert Systems with Applications*, vol. 165, p. 113816, 2021.

[22] Y. Wang, X. Zhang, and J. Li, "Alphastock: A buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks," *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

[23] A. Malm *et al.*, "Deep reinforcement learning agent for S&P 500 stock selection," *Entropy*, vol. 22, no. 9, p. 130, 2020. [Online]. Available: https://www.mdpi.com/2075-1680/9/4/130

[24] V. Santos *et al.*, "Management of investment portfolios employing reinforcement learning," *Applied Sciences*, 2024. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC10773882/

[25] A. Mirzaeinia *et al.*, "Deep reinforcement learning strategies in finance: Insights into asset holding, trading behavior, and purchase diversity," *arXiv preprint arXiv:2407.09557*, 2024. [Online]. Available: https://arxiv.org/abs/2407.09557

[26] Y. Zhang *et al.*, "Deep reinforcement learning approach to portfolio optimization in the Australian stock market," *IntechOpen*, 2024. [Online]. Available: https://www.intechopen.com/journals/1/articles/410

[27] A. Gorgulho, R. Neves, and N. Horta, "Deep reinforcement learning for portfolio selection," *Research in International Business and Finance*, vol. 70, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1044028324000887

[28] J. J. Murphy, *Technical Analysis of the Financial Markets*, 2nd ed. New York Institute of Finance, 1999.

[29] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representation and trading," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 3, pp. 653–664, 2016.

[30] J. W. Wilder, *New Concepts in Technical Trading Systems*. Trend Research, 1978.

[31] G. Appel, *The Moving Average Convergence Divergence Trading Method*. Signalert Corporation, 1979.

[32] J. Bollinger, *Bollinger on Bollinger Bands*. McGraw-Hill, 2001.

[33] X. Zhai, X. Chen, L. Song, and Z. Xiao, "A survey of deep reinforcement learning for algorithmic trading," *arXiv preprint arXiv:2107.14039*, 2021.

[34] S. DeCosta, L. Tran, and H. Sudarsan, "Hybrid risk-aware reward design in rl-based equity portfolio optimization," *Journal of Computational Finance*, 2025, forthcoming.

[35] Y. Wu, L. Chen, and W. Li, "Risk-aware deep reinforcement learning for portfolio optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 12, pp. 5595–5608, 2021.

[36] Z. Huang, M. Wang, and L. Tran-Thanh, "Sharpe ratio optimization using reinforcement learning with constrained policy," *Quantitative Finance*, vol. 22, no. 9, pp. 1453–1468, 2022.

[37] J. Wang, T. Wang, and J. Zhang, "Information ratio and turnover-aware reinforcement learning for portfolio allocation," *arXiv preprint arXiv:2212.04174*, 2022.

[38] R. Jin and M. Wu, "Cvar optimization in portfolio management using reinforcement learning," *Computational Economics*, vol. 60, pp. 89–112, 2022.

[39] R. Srivastava, A. Khurana, and R. Kumar, "Multi-objective reward engineering for reinforcement learning in finance," *Finance and AI Review*, 2025, in press.

[40] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[41] G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, J. Hunt, T. Mann, T. Weber, T. Degris, and B. Coppin, "Deep reinforcement learning in large discrete action spaces," *arXiv preprint arXiv:1512.07679*, 2015.

[42] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[43] E. Greensmith, P. L. Bartlett, and J. Baxter, "Variance reduction techniques for gradient estimates in reinforcement learning," in *Journal of Machine Learning Research*, vol. 5, 2004, pp. 1471–1530.

[44] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[45] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[46] R. J. Williams and J. Peng, "Function optimization using connectionist reinforcement learning algorithms," *Connection Science*, vol. 3, no. 3, pp. 241–268, 1991.

[47] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.

[48] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[49] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[50] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International Conference on Machine Learning*. PMLR, 2014, pp. 387–395.

[51] P. Fortune, "A primer on u.s. stock price indices," *New England Economic Review*, Nov/Dec 1998.

[52] J. Lin, G. C. Selden, J. B. Shoven, and C. Sialm, "Replicating the dow jones industrial average," National Bureau of Economic Research, Tech. Rep. 28528, 2021, nBER Working Paper.

[53] D. Yang, Y. Yang, W. Zheng, and H. Sha, "Research on the impact of algorithmic trading on market volatility," *Scientific Reports*, vol. 15, p. Article 30073, 2025.

[54] H. Yang, X. Liu, S. Zhong, and A. Walid, "Deep reinforcement learning for automated stock trading: An ensemble strategy," 2025.

[55] WongJiaJie and LiuLiLi, "Portfolio optimization through a multi-modal deep reinforcement learning framework," 2025.