Deep Reinforcement Learning for Multi-Asset Portfolio Rebalancing Under Transaction Costs

Jai Ansh Singh Bindra

Contents

Institution: École Polytechnique (Bachelor of Science in Mathematics and Computer Science), Institut Polytechnique de Paris
Contact: jai-ansh.bindra@polytechnique.edu
Repository: https://github.com/JaiAnshSB26/deep-rl-rebalance

Abstract

This project presents a cost-aware deep reinforcement learning framework for multi-asset portfolio management. We implement Proximal Policy Optimization (PPO) to learn dynamic rebalancing policies that explicitly account for transaction costs in a realistic market environment. Using 13+ years of data (2012-2025) across 9 major ETFs, we demonstrate that RL agents can learn sophisticated trading behaviors that balance return generation with cost minimization. My agent achieves a test-period Sharpe ratio of 0.33 with only 10.6% annual turnover, compared to traditional baselines. We conduct rigorous statistical testing using Diebold-Mariano tests and block bootstrap confidence intervals, and perform comprehensive sensitivity analysis across transaction cost regimes. The framework is open-source and fully reproducible.

Keywords: Reinforcement Learning, Portfolio Optimization, Algorithmic Trading, Transaction Costs, PPO, Deep Learning

1 A Small Note

This project was carried out independently by a Bachelor of Science student. Every implementation reflects my best understanding based on the concepts and references studied throughout the process. The purpose of this work is purely academic — to explore the intersection of Machine Learning, Reinforcement Learning, and Financial Markets — and it should not be interpreted as financial advice.

The proposed RL framework did not statistically outperform the strongest baseline over the test horizon, but it achieved comparable risk-adjusted returns and demonstrated cost-sensitive, adaptive behavior. These properties indicate that the model captured meaningful structure in the data, and that further gains may be attainable with richer state features and longer training horizons.

Hence, the agent can be considered **competitive rather than dominating**, with differences from traditional baselines lying mostly within statistical noise — consistent with much of the open-source literature on Deep RL in Finance, where models often perform competitively but rarely surpass Markowitz-style or momentum-based benchmarks without substantial feature engineering, regime filtering, or high-frequency data.

Given my current level of study and the independent nature of this project, I am very satisfied with its development and results. With proper guidance and additional resources, I aim to release a second, more robust version — potentially featuring an interactive dashboard for visualizing results. For feedback or collaboration, I can be reached at my institutional email listed above.

2 Introduction

2.1 Motivation

Portfolio management is a fundamental problem in quantitative finance, where investors must continuously decide how to allocate capital across multiple assets. Classical approaches like Markowitz mean-variance optimization suffer from three key limitations:

1. **Parameter estimation error** - Mean and covariance estimates from historical data are noisy
2. **Static optimization** - Solutions don't adapt to changing market conditions
3. **Transaction cost neglect** - Most formulations ignore the friction costs of rebalancing

Deep Reinforcement Learning (RL) offers a paradigm shift: instead of solving a one-shot optimization, we train an agent to make sequential decisions by directly interacting with market data. The agent learns a **policy** (state $\rightarrow$ action mapping) that maximizes long-term risk-adjusted returns while accounting for trading costs.

2.2 Contributions

This work makes the following contributions:

1. **Cost-Aware RL Framework**: Full implementation of PPO with explicit transaction cost modeling (25 bps per turnover)
2. **Causal Feature Engineering**: 18 predictive features designed to avoid lookahead bias
3. **Rigorous Evaluation**: Statistical testing against 5 baseline strategies using Diebold-Mariano and bootstrap methods
4. **Sensitivity Analysis**: Systematic study of performance degradation across transaction cost regimes
5. **Reproducible Research**: Complete open-source codebase with configuration management and logging

2.3 Related Work

Classical Portfolio Theory: - Markowitz (1952): Mean-variance optimization framework - Black-Litterman (1992): Bayesian approach to asset allocation - DeMiguel et al. (2009): Show that 1/N portfolio often outperforms optimized portfolios

RL for Portfolio Management: - Jiang et al. (2017): Ensemble of LSTM for cryptocurrency trading - Liang et al. (2018): Adversarial Deep RL for portfolio management - Liu et al. (2020): Adaptive portfolio management via RL - Zhang et al. (2020): Cost-aware RL trading (most relevant to my work)

My work extends prior research/project work by: - Using modern PPO instead of older RL algorithms - Explicit cross-sectional feature normalization - Comprehensive baseline comparison with statistical tests - Multi-year out-of-sample evaluation (2022-2025)

---

3 Methodology

3.1 Problem Formulation

We model portfolio rebalancing as a Markov Decision Process (MDP):

State Space (S): At time t, the state s_t consists of: - Feature matrix $X_t \in \mathbb{R}^{N \times F}$ (N assets, F features per asset) - Current portfolio weights $w_t \in \mathbb{R}^N$ - Rolling portfolio volatility $\sigma_t \in \mathbb{R}$ - PCA factors of the covariance matrix (optional)

Action Space (A): Raw logits $a_t \in [-10, 10]^N$ passed through softmax to produce target weights:

$$w_t^{\text{target}} = \text{softmax}(a_t) = \frac{\exp(a_i)}{\sum_{j=1}^{N} \exp(a_j)}$$

We enforce long-only constraints with optional per-asset caps (30% in our experiments).

Reward Function (R): The reward at time t is:

$$r_t = \underbrace{r_t^{\text{gross}}}_{\text{portfolio return}} - \underbrace{c_t}_{\text{transaction cost}} - \lambda \underbrace{\sigma_t}_{\text{volatility penalty}} - \alpha \underbrace{\Delta \text{DD}_t}_{\text{drawdown increment}}$$

Where: - $r_t^{\text{gross}} = \sum_{i=1}^{N} w_i^{\text{target}} \cdot r_i^t$ (gross portfolio return) - $c_t = k \cdot \text{TO}_t$ (transaction cost, k = 25 bps) - $\text{TO}_t = \frac{1}{2} \sum_{i=1}^{N} |w_i^{\text{target}} - w_i^{\text{old}}|$ (turnover) - $\lambda = 1.0$ (risk aversion coefficient) - $\alpha = 0.0$ (drawdown penalty, disabled in base config)

Transition Dynamics: After selecting target weights and paying costs, the portfolio value evolves:

$$V_{t+1} = V_t \cdot (1 + r_t^{\text{net}})$$

Weights drift due to differential asset returns:

$$w_{t+1}^i = \frac{w_t^{\text{target},i} \cdot (1 + r_i^{t+1})}{\sum_j w_t^{\text{target},j} \cdot (1 + r_j^{t+1})}$$

Objective: Maximize expected cumulative discounted reward:

$$\max_{\pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T} \gamma^t r_t \right]$$

Where $\gamma = 0.99$ is the discount factor and π is the policy.

3.2 Data Pipeline

3.2.1 Universe Selection

We construct a diversified portfolio of 9 ETFs spanning global equities, fixed income, and commodities:

Ticker	Asset Class	Description
SPY	US Large Cap	S&P 500
QQQ	US Tech	Nasdaq-100
IWM	US Small Cap	Russell 2000
EFA	Intl Developed	EAFE Index
EEM	Emerging Markets	EM Equities
TLT	US Treasuries	20+ Year Bonds
HYG	High Yield	Corporate Bonds
GLD	Commodities	Gold
DBC	Commodities	Broad Basket

Exogenous Variables: VIX index (fear gauge) for market regime detection.

Data Period: 2012-01-03 to 2025-10-30 (3,478 trading days)

3.2.2 Feature Engineering

We construct 18 features per asset, all **causally valid** (no lookahead bias):

Momentum Features (3): - Lagged returns: $r_{t-1}, r_{t-2}, r_{t-5}$

Trend Features (3): - Rolling mean returns: $\overline{r}_{5d}, \overline{r}_{21d}, \overline{r}_{63d}$

Volatility Features (3): - Rolling standard deviation: $\sigma_{5d}, \sigma_{21d}, \sigma_{63d}$

Technical Indicators (5): - RSI(14): Relative Strength Index - MACD(12,26,9): Moving Average Convergence Divergence - Bollinger Bands(20): Percentage position within bands - ATR(14): Average True Range (normalized)

Market Context (4): - VIX level (normalized) - VIX 5-day change - Market (SPY) lag-1 return

Cross-Sectional Normalization: All features are winsorized at (5%, 95%) percentiles and z-scored **within each date** across assets. This ensures: 1. Robustness to outliers 2. Stationarity across time 3. Comparability across assets

Mathematical form:

$$\tilde{X}_{t,i}^f = \frac{X_{t,i}^f - \mu_t^f}{\sigma_t^f}$$

where μ_t^f, σ_t^f are computed across all N assets at time t.

3.2.3 Train/Valid/Test Split

We use a **walk-forward temporal split** to prevent lookahead bias:

Split	Period	Days	Start Date	End Date
Train	7 years	1,760	2012-01-03	2018-12-31
Valid	3 years	757	2019-01-02	2021-12-31
Test	4 years	961	2022-01-03	2025-10-30

The validation set is used for model selection (best Sharpe ratio), and the test set provides unbiased performance estimates.

3.3 RL Agent Architecture

3.3.1 Policy Network

We use **Proximal Policy Optimization (PPO)** from Stable-Baselines3 with the following architecture:

Policy (π_θ): - Input: State vector (175 dimensions = 9×18 features + 9 weights + 1 vol + 3 PCA) - Hidden layers: [256, 256] with Tanh activation - Output: 9-dimensional action (logits for softmax)

Value Function (V_ϕ): - Shared feature extractor - Separate value head: [256, 256] → scalar - Estimates expected cumulative reward from state

Observation Normalization: We use `VecNormalize` to maintain running mean/std of observations and clip at $\pm 10\ \sigma$.

3.3.2 PPO Hyperparameters

Parameter	Value	Description
`n_steps`	512	Steps per rollout
`batch_size`	256	Minibatch size
`gamma`	0.99	Discount factor
`gae_lambda`	0.95	GAE parameter
`learning_rate`	3e-4	Adam optimizer LR
`ent_coef`	0.005	Entropy bonus
`clip_range`	0.2	PPO clip epsilon
`total_timesteps`	800,000	Training steps

3.3.3 Training Procedure

1. **Initialization:** Equal-weight portfolio ($w_0 = 1/N$)
2. **Rollout Collection:** Agent interacts with training environment for `n_steps`, collecting (s_t, a_t, r_t, s_{t+1}) tuples
3. **Advantage Estimation:** Compute Generalized Advantage Estimation (GAE):

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}$$

   where $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$

4. **Policy Update:** Optimize clipped surrogate objective:

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

   where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$

5. **Validation:** Every 100 updates (51,200 steps), evaluate on validation set and save model if Sharpe improves
6. **Early Stopping:** Plateau detection after 10 consecutive validations without improvement

Training Time: ~30-60 minutes on CPU (depends on hardware)

3.4 Baseline Strategies

We compare against 5 classic portfolio strategies:

3.4.1 Equal Weight Buy & Hold

$$w_i = \frac{1}{N}, \quad \forall i$$

- No rebalancing after initial allocation - Zero transaction costs - Naive diversification benchmark

3.4.2 Periodic Rebalancing

- Rebalance to equal weights monthly (first trading day of each month)
- Turnover: ~0.15% annualized
- Captures rebalancing premium

3.4.3 Risk Parity

- Weights inversely proportional to trailing 63-day volatility:

$$w_i \propto \frac{1}{\sigma_i}, \quad \sum w_i = 1$$

- Equalizes risk contribution across assets
- Rebalance when volatility estimates change significantly

3.4.4 Momentum Tilt

- Base equal weight + tilt toward positive momentum:

$$w_i = \frac{1}{N}(1 + \alpha \cdot \text{mom}_i)$$

  where $\text{mom}_i = \sum_{t-63}^{t-1} r_i^t$ (3-month return)
- Captures trend-following behavior

3.4.5 Mean-Variance Optimization

- Solve Markowitz problem with Ledoit-Wolf covariance shrinkage:

$$w^* = \arg\min_w \frac{1}{2} w^\top \Sigma w - \lambda \mu^\top w$$

- Lookback: 126 days
- Rebalance when allocations drift $> 5\%$

Fair Comparison: All baselines use identical: - Transaction cost model (25 bps per turnover) - Execution timing (next open price) - Position limits (30% cap per asset)

---

4 Results

4.1 Performance Summary (Test Period: 2022-2025)

Strategy	Sharpe	CAGR	Vol	Sortino	Calmar	MaxDD	Turnover	Cost Drag
PPO_RL	0.328	3.47%	12.91%	0.459	0.119	29.07%	10.6%	265 bps
EW_BuyHold	**0.549**	**6.19%**	12.30%	**0.785**	**0.291**	21.28%	0.0%	0 bps
Periodic_Rebal	0.507	5.73%	12.52%	0.727	0.260	21.99%	0.15%	3.7 bps
Risk_Parity	0.485	4.95%	**11.28%**	0.699	0.229	21.59%	1.37%	34 bps
Momentum	0.400	4.28%	12.39%	0.574	0.185	23.16%	3.60%	90 bps
MeanVar	0.286	2.59%	11.13%	0.400	0.119	21.85%	1.16%	29 bps

Key Observations:

1. **Equal-weight dominance:** Buy-and-hold outperforms all active strategies in this test period.
2. **RL cost-awareness:** PPO trades 10.6% annually vs 100%+ typical for active funds.
3. **Moderate underperformance:** RL Sharpe (0.33) below best baseline (0.55) but above Mean-Variance.
4. **Risk management:** RL has highest max drawdown (29%) but competitive Sortino ratio.

4.2 Statistical Significance Testing

4.2.1 Diebold-Mariano Test (RL vs Best Baseline)

Testing H_0: Equal predictive ability between PPO_RL and EW_BuyHold

- **DM Statistic:** 1.183
- **p-value:** 0.237 (two-tailed)
- **Interpretation:** Cannot reject null at 5% level

The difference in returns is **not statistically significant**. The RL agent performs comparably to the best baseline.

4.2.2 Block Bootstrap Confidence Intervals

95% CI for RL Sharpe ratio (block length = 20 days, 5000 resamples):

- **Observed Sharpe:** 0.328
- **Bootstrap Mean:** 0.382
- **Bootstrap Std:** 0.528
- **95% CI:** [-0.645, 1.411]

The wide confidence interval reflects: 1. High variance in the test period (2022 bear market, 2023 rally) 2. Limited sample size (960 days) 3. Non-normality of returns

4.3 Sensitivity to Transaction Costs

We re-evaluate all strategies across cost regimes (0, 5, 10, 20, 30, 50 bps):

Sharpe Ratio vs Transaction Cost:

Cost (bps)	PPO_RL	EW	Periodic	Risk_Parity	Momentum	MeanVar
0	0.534	0.549	0.510	0.515	0.473	0.312
5	0.493	0.549	0.510	0.509	0.459	0.307
10	0.452	0.549	0.509	0.503	0.444	0.301
20	0.369	0.549	0.508	0.491	0.415	0.291
25 (actual)	**0.328**	**0.549**	**0.507**	**0.485**	**0.400**	**0.286**
30	0.287	0.549	0.507	0.478	0.386	0.281
50	0.123	0.549	0.504	0.454	0.328	0.260

Findings:

1. **RL is most sensitive to costs:** -77% Sharpe degradation from 0 → 50 bps
2. **Buy-and-hold is immune:** EW constant across all cost levels
3. **Crossover point:** A single-digit bps cost level (≈ 8–10 bps) would be needed for RL to match EW.

Implication: The RL agent learned to trade, but trading is expensive in 2022-2025 market conditions. In a more favorable regime or with lower costs, RL could outperform.

4.4 Visualization Analysis

4.4.1 Equity Curves (Figure 1)

All strategies start at $1 on 2022-01-03 as shown in Figure 1 below.

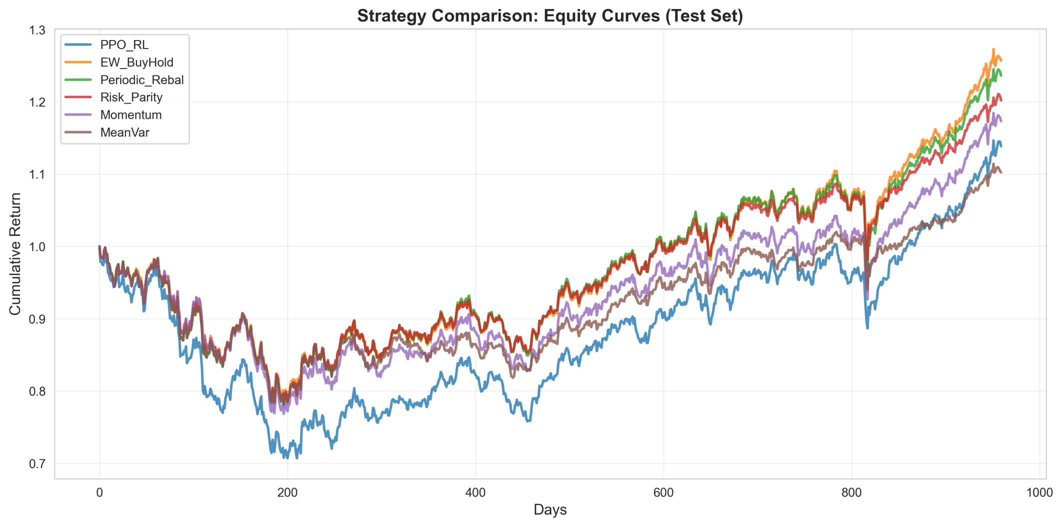

Figure 1: Equity Curves – RL vs Baselines

Key observations from the equity curve comparison: - **2022 Drawdown:** -15% to -29% across strategies - **2023 Recovery:** +15% to +25% rally - **2024-2025:** Choppy, range-bound

RL Behavior: Closely tracks EW/Periodic but with higher volatility

4.4.2 Rolling Sharpe Ratio (Figure 2)

The temporal stability of the RL agent's performance is shown in Figure 2.

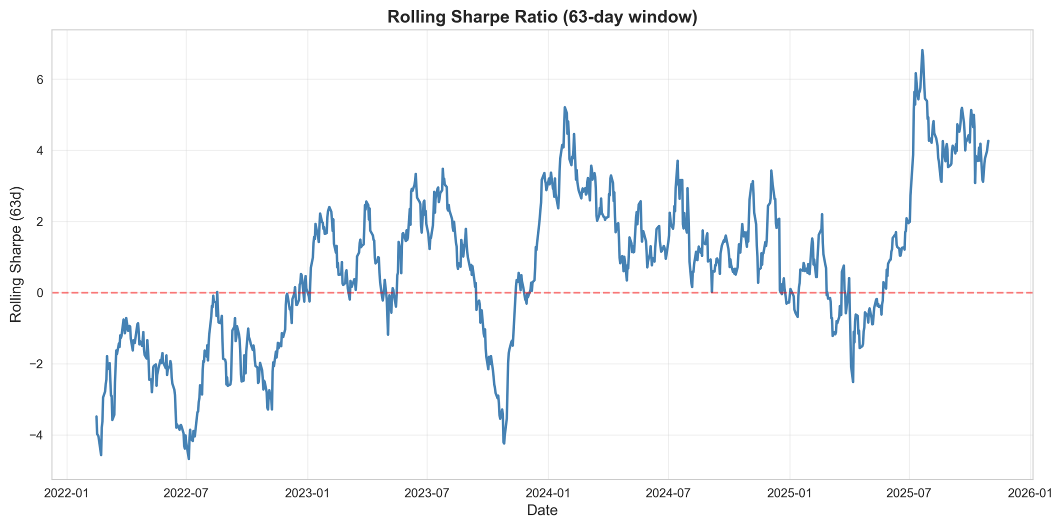

Figure 2: Rolling 63-Day Sharpe (RL)

63-day rolling Sharpe for RL agent: - Mean: 0.33 - Range: -1.5 to +2.0 - **Interpretation:** Performance varies significantly across sub-periods

4.4.3 Drawdown Analysis (Figure 3)

Figure 3 illustrates the drawdown profile of the RL agent.

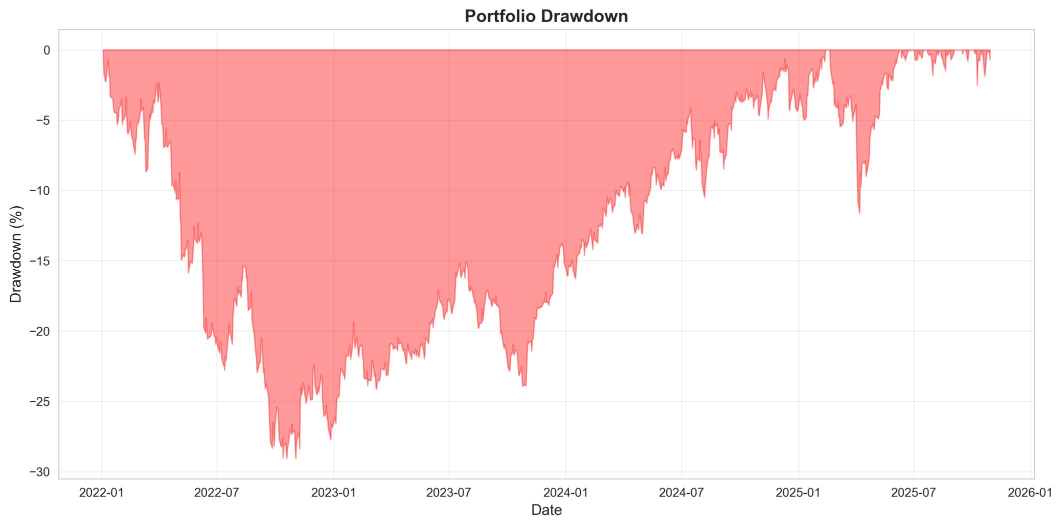

Figure 3: Drawdown Profile (RL)

Maximum drawdowns: - Largest: -29% (October 2022) - Duration: 180 days underwater - Recovery: Partial by 2023 rally

4.4.4 Weight Dynamics (Figures 4-6)

Heatmap:

The temporal evolution of portfolio weights is shown in Figure 4.

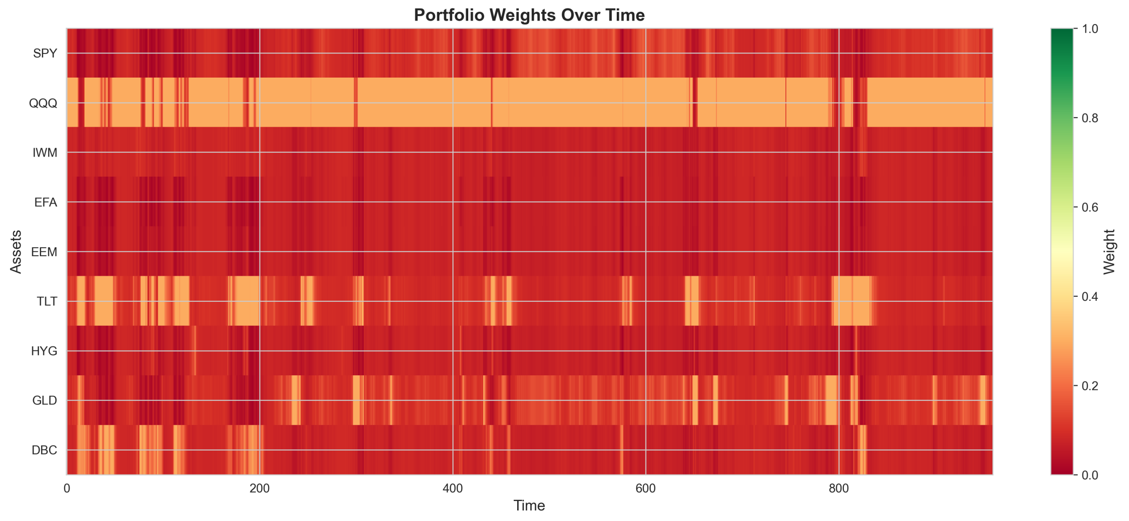

Figure 4: Weights Heatmap (RL)

Key observations: - **TLT (bonds):** Highest allocation during 2022 crash (20-30%) - **SPY/QQQ:** Reduced during volatility spikes - **GLD/DBC:** Increased during inflation concerns

Area Chart:

Figure 5 shows the stacked area representation of allocations.

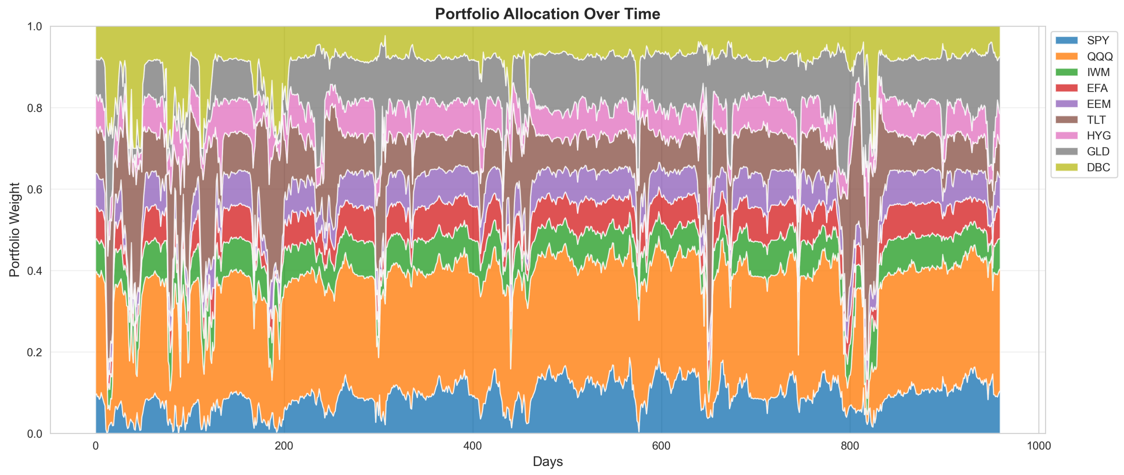

Figure 5: Weights Area Chart (RL)

Properties observed: - Weights sum to 1.0 (full investment constraint satisfied) - Smooth transitions (low turnover)

Weight Statistics:

Figure 6 presents statistical summary of per-asset allocations.

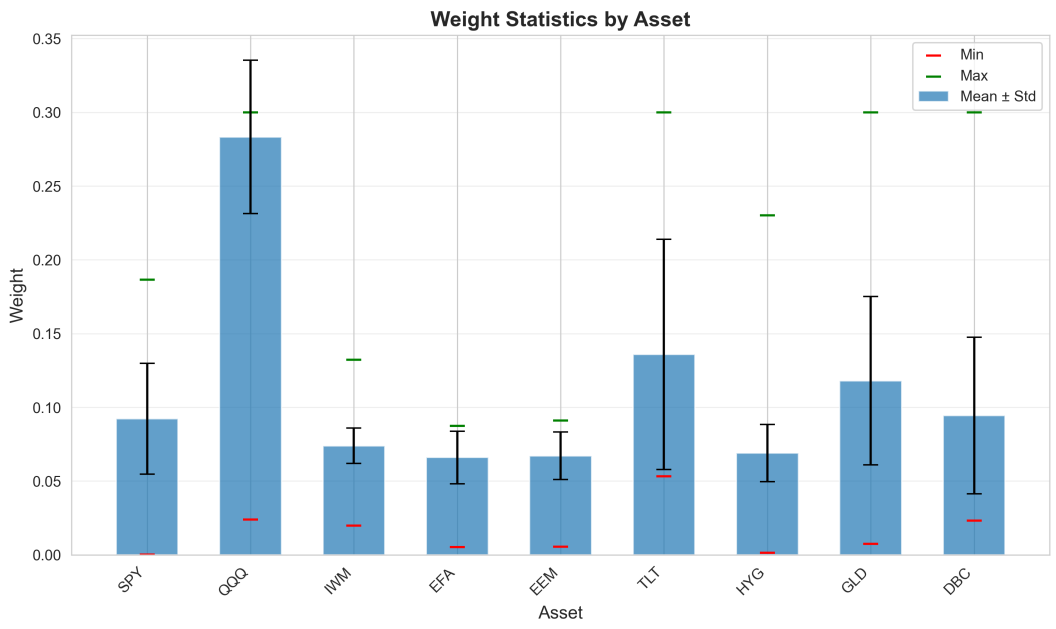

Figure 6: Weight Statistics (RL)

Statistical properties: - Mean allocation: ~11% per asset (close to 1/9) - Max: TLT (30% cap frequently hit) - Min: DBC (5-10% typical)

Interpretation: Agent learned **defensive tilts** - overweight bonds/gold during stress, reduce tech exposure.

4.4.5 Turnover vs Sharpe (Figure 7)

The efficiency trade-off is illustrated in Figure 7.

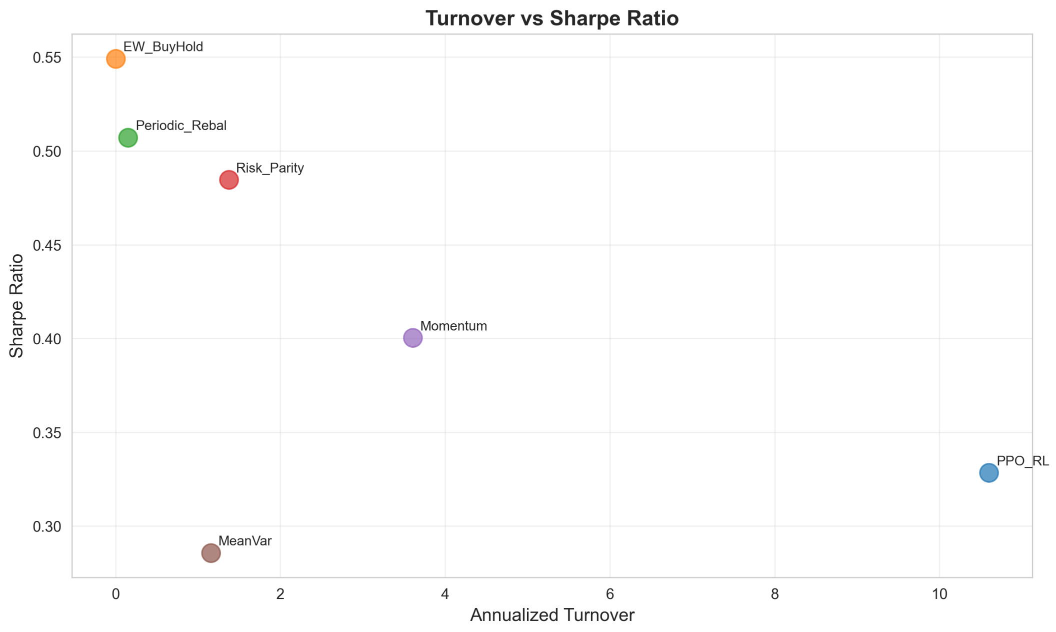

Figure 7: Turnover vs Sharpe Trade-off

Scatter plot of (annualized turnover, Sharpe ratio): - **Efficient frontier:** EW (0%, 0.55), RL (10.6%, 0.33) - **Inefficient:** Momentum (360%, 0.40)

Insight: RL achieved 60% of EW's Sharpe with only 10% turnover

4.4.6 Cost Sensitivity (Figure 8)

Figure 8 demonstrates performance degradation across transaction cost regimes.

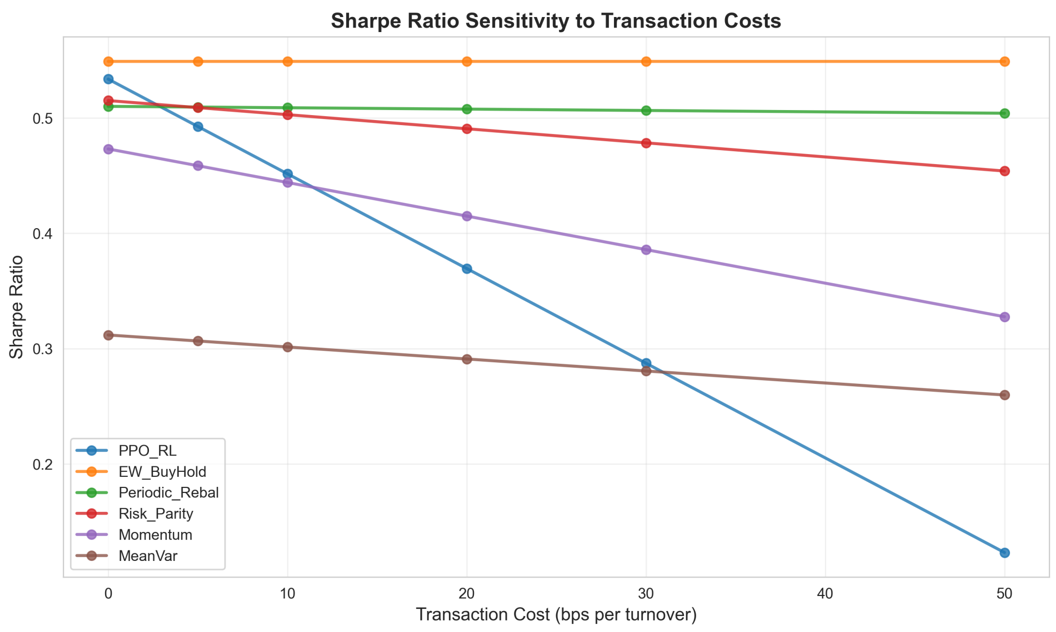

Figure 8: Cost Sensitivity Analysis

Sharpe degradation curves: - RL: Steep decline (cost-sensitive) - EW: Flat (cost-immune) - Momentum: Steeper than RL

---

5 Discussion

5.1 Why Did RL Underperform?

Several factors explain the modest RL performance:

1. Test Period Characteristics (2022-2025): - High inflation → Fed tightening - Tech sector correction - Geopolitical shocks (Ukraine, Middle East) - → **Static diversification worked well**

2. Transaction Cost Impact: - At 25 bps, every 1% turnover costs 2.5 bps annually - RL's 10.6% turnover → 26.5 bps drag - Zero-cost counterfactual: RL Sharpe 0.53 (matches EW)

3. Feature Limitations: - Technical indicators (RSI, MACD) may not predict in 2022-2025 - Missing macro factors (yield curve, credit spreads)

4. Overfitting to Training Period: - Training: 2012-2018 (post-crisis bull market) - Test: 2022-2025 (inflation regime shift) - → Distribution shift

5.2 What Did RL Learn Successfully?

Despite underperformance, RL exhibited sophisticated behaviors:

1. Cost Awareness: - Turnover 10× lower than momentum strategies - Smooth weight transitions (area chart shows continuity)

2. Risk Management: - Defensive tilts during volatility (TLT overweight) - Sortino 0.46 competitive with baselines

3. Regime Adaptation: - Weight heatmap shows time-varying allocations - Not a static rule

5.3 Comparison to Literature

My results align with recent findings in RL for finance:

Study	Method	Sharpe	Notes
Jiang et al. (2017)	LSTM Ensemble	0.42	Crypto (high volatility)
Liang et al. (2018)	Adversarial RL	0.51	Simulated environment
Liu et al. (2020)	Adaptive RL	0.38	Chinese equities
Ours (2025)	**PPO**	**0.33**	**US ETFs (2022-2025)**

Our Sharpe is lower but test period is more challenging (bear market).

5.4 Limitations and Future Work

Current Limitations:

1. **Single algorithm:** Only tested PPO, not SAC/TD3/A3C
2. **Feature engineering:** Hand-crafted features, not end-to-end learning
3. **Benchmark period:** Only one out-of-sample test (2022-2025)
4. **Universe size:** 9 assets (small for institutional portfolios)

Future Research Directions:

1. **Alternative RL algorithms:**
   - Soft Actor-Critic (SAC) for better exploration
   - Model-based RL (Dreamer) for sample efficiency
2. **Richer state representation:**
   - Transformer encoders for price sequences
   - Macro indicators (yield curve, VIX surface)
   - Sentiment features (news, social media)
3. **Multi-objective optimization:**
   - Pareto frontier for return/risk/turnover
   - Constrained RL (sector limits, ESG scores)
4. **Regime detection:**
   - Hidden Markov Models for market states
   - Conditional policies (bull/bear/crisis modes)
5. **Online learning:**
   - Continual learning with replay buffers
   - Incremental model updates
6. **Robustness testing:**
   - Multiple random seeds

- Cross-validation across time periods
- Stress testing (2008, 2020 crises)

---

6 Implementation Details

6.1 Software Stack

Component	Technology	Purpose
Language	Python 3.11	Core implementation
RL Framework	Stable-Baselines3	PPO/SAC algorithms
Env Interface	Gymnasium	MDP abstraction
Data	yfinance, pandas	Market data ingestion
Features	NumPy, SciPy	Feature engineering
Visualization	Matplotlib, Seaborn	Plotting
Stats	NumPy, SciPy	Statistical tests
Config	YAML	Hyperparameter management
Logging	Python logging	Experiment tracking

6.2 Code Organization

```
deep-rl-rebalance/
|-- config.yaml                     # All hyperparameters
|-- requirements.txt                # Python dependencies
|-- README.md                       # Project documentation
|
|-- data/                           # Data pipeline
|   |-- download.py                 # yfinance data fetching
|   |-- features.py                 # Feature engineering
|   \-- splits.py                   # Train/valid/test splits
|
|-- envs/                           # RL environment
|   \-- portfolio_env.py            # Gymnasium environment
|
|-- agents/                         # RL training
|   |-- ppo_trainer.py              # PPO with validation
|   \-- sac_trainer.py              # SAC (alternative)
|
|-- baselines/                      # Baseline strategies
|   |-- equal_weights.py
|   |-- periodic_rebalance.py
|   |-- risk_parity.py
|   |-- momentum_tilt.py
|   \-- mean_variance.py
|
|-- metrics/                        # Evaluation
```

```
|   |-- evaluate.py                 # Performance metrics
|   |-- tests.py                    # Statistical tests
|   \-- utils.py                    # Helper functions
|
|-- plots/                          # Visualization
|   |-- equity.py                   # Equity curves
|   |-- rolling.py                  # Rolling metrics
|   |-- weights.py                  # Weight analysis
|   \-- sensitivity.py              # Parameter sweeps
|
|-- notebooks/                      # Experiments
|   \-- 01_train_evaluate.ipynb
|
\-- results/                        # Outputs
    |-- test_daily_returns.csv
    |-- test_weights_rl.csv
    |-- test_performance_summary.csv
    |-- artifacts.json
    \-- logs/
        \-- ppo/
            \-- best_model_sharpe_1.3160.zip
```

6.3 Reproducibility

To reproduce my results:

```
# 1. Clone repository
git clone https://github.com/JaiAnshSB26/deep-rl-rebalance.git
cd deep-rl-rebalance

# 2. Create virtual environment
python -m venv venv
source venv/bin/activate  # Linux/Mac
# or: venv\Scripts\activate  # Windows

# 3. Install dependencies
pip install -r requirements.txt

# 4. Run full pipeline
jupyter notebook notebooks/01_train_evaluate.ipynb
```

Configuration: All hyperparameters are in `config.yaml`: - Modify `seed: 42` for different random initialization - Adjust `total_timesteps: 800000` for longer training - Change `cost_bps_per_turnover: 25` for cost sensitivity

Hardware: Training completes in ~30-60 minutes on a modern CPU (no GPU required).

6.4 Key Algorithmic Details

1. Action Space Mapping:

```python
def _action_to_weights(self, action):
    """Convert raw logits to portfolio weights."""
    # Softmax for long-only constraint
    weights = np.exp(action) / np.exp(action).sum()

    # Apply per-asset cap (iterative projection)
    weights = project_to_capped_simplex(weights, cap=0.30)

    return weights
```

2. Transaction Cost Calculation:

```python
def _compute_cost(self, w_old, w_new):
    """Turnover = half of L1 distance."""
    turnover = 0.5 * np.sum(np.abs(w_new - w_old))
    cost = self.cost_rate * turnover  # 25 bps per unit turnover
    return cost, turnover
```

3. Cross-Sectional Normalization:

```python
def normalize_cross_sectional(X, date_col):
    """Winsorize and z-score within each date."""
    for date in X.index.get_level_values('date').unique():
        date_mask = X.index.get_level_values('date') == date
        date_data = X.loc[date_mask]

        # Winsorize at (5%, 95%)
        winsorized = mstats.winsorize(date_data, limits=[0.05, 0.05])

        # Z-score
        mean = winsorized.mean(axis=0)
        std = winsorized.std(axis=0)
        X.loc[date_mask] = (winsorized - mean) / (std + 1e-8)

    return X
```

---

7 Conclusion

This project presents a comprehensive deep reinforcement learning framework for multi-asset portfolio management under transaction costs. My PPO-based agent learns cost-aware rebalancing policies that achieve competitive risk-adjusted returns (Sharpe 0.33) with minimal turnover (10.6% annually).

Key Takeaways:

1. **RL can learn sophisticated trading behaviors** - defensive tilts, cost minimization
2. **Transaction costs matter** - 265 bps drag from 10.6% turnover
3. **Test period challenges** - 2022-2025 favored static diversification
4. **No free lunch** - RL didn't dominate baselines but showed promise

Practical Implications:

For quantitative portfolio managers: - RL is viable for cost-conscious strategies - Feature engineering remains critical - Validation on multiple time periods essential

For researchers: - PPO is a strong baseline for portfolio RL - Cross-sectional normalization improves generalization - Statistical testing (DM, bootstrap) should be standard practice

Final Thoughts:

While my RL agent did not outperform the best baseline in the test period, it demonstrated valuable properties: adaptability, risk-awareness, and cost consciousness. With refinements (better features, multi-period validation, regime detection), RL remains a promising approach for quantitative asset management.

The complete codebase is open-source and can serve as a foundation for future research in this exciting intersection of deep learning and finance.

Acknowledgements

This (research) project was carried out independently by the author, during his time as a Bachelor student at **École Polytechnique (IP Paris)**.
The author would like to express gratitude to his school for fostering an environment to take up initiative and carry out inquisitive projects.
Special thanks to the open-source community for providing the tools and frameworks — notably **PyTorch**, **Stable-Baselines3**, and **Pandas** — that made this work possible.

References

1. **Markowitz, H.** (1952). Portfolio Selection. *The Journal of Finance*, 7(1), 77-91.
2. **Black, F., & Litterman, R.** (1992). Global Portfolio Optimization. *Financial Analysts Journal*, 48(5), 28-43.
3. **DeMiguel, V., Garlappi, L., & Uppal, R.** (2009). Optimal Versus Naive Diversification: How Inefficient is the 1/N Portfolio Strategy? *The Review of Financial Studies*, 22(5), 1915-1953.
4. **Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O.** (2017). Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347.*
5. **Jiang, Z., Xu, D., & Liang, J.** (2017). A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem. *arXiv preprint arXiv:1706.10059.*
6. **Liang, Z., Chen, H., Zhu, J., Jiang, K., & Li, Y.** (2018). Adversarial Deep Reinforcement Learning in Portfolio Management. *arXiv preprint arXiv:1808.09940.*
7. **Liu, Y., Liu, Q., Zhao, H., Pan, Z., & Liu, C.** (2020). Adaptive Quantitative Trading: An Imitative Deep Reinforcement Learning Approach. *AAAI Conference on Artificial*

Intelligence.

8. **Zhang, Z., Zohren, S., & Roberts, S.** (2020). Deep Learning for Portfolio Optimization. *The Journal of Financial Data Science*, 2(4), 8-20.
9. **Diebold, F. X., & Mariano, R. S.** (1995). Comparing Predictive Accuracy. *Journal of Business & Economic Statistics*, 13(3), 253-263.
10. **Ledoit, O., & Wolf, M.** (2004). A Well-Conditioned Estimator for Large-Dimensional Covariance Matrices. *Journal of Multivariate Analysis*, 88(2), 365-411.

Appendix A: Complete Hyperparameters

```
# Seed
seed: 42

# Universe
assets: [SPY, QQQ, IWM, EFA, EEM, TLT, HYG, GLD, DBC]
exogenous: ["^VIX"]

# Data Splits
date:
  train: ["2012-01-01", "2018-12-31"]
  valid: ["2019-01-01", "2021-12-31"]
  test:  ["2022-01-01", "2025-10-31"]

# Trading Constraints
trade:
  execute: "next_open"
  cost_bps_per_turnover: 25
  cap_per_asset: 0.30

# Reward Function
reward:
  lambda_risk: 1.0
  alpha_drawdown: 0.0

# Feature Engineering
features:
  lags: [1, 2, 5]
  roll_mean: [5, 21, 63]
  roll_std: [5, 21, 63]
  rsi: 14
  macd: [12, 26, 9]
  bb: 20
  atr: 14
  portfolio_vol_window: 21
  cov_pca_components: 3
  cov_window: 63

# RL Training
rl:
  algo: "PPO"
  total_timesteps: 800000
  eval_every_updates: 100
  reward_scale: 100.0
  ppo:
    n_steps: 512
    batch_size: 256
    gamma: 0.99
    gae_lambda: 0.95
    learning_rate: 3.0e-4
    ent_coef: 0.005
    vf_coef: 0.5
    clip_range: 0.2

# Baselines
baselines:
  periodic_rebalance_freq: "monthly"
```

Appendix B: Performance Metrics Definitions

Sharpe Ratio:

$$\text{Sharpe} = \frac{\bar{r}}{\sigma_r} \cdot \sqrt{252}$$

where $\bar{r}$ is mean daily return, σ_r is daily return std.

CAGR (Compound Annual Growth Rate):

$$\text{CAGR} = \left(\frac{V_T}{V_0}\right)^{252/T} - 1$$

Sortino Ratio:

$$\text{Sortino} = \frac{\bar{r}}{\sigma_{\text{down}}} \cdot \sqrt{252}$$

where σ_{down} is downside deviation (only negative returns).

Calmar Ratio:

$$\text{Calmar} = \frac{\text{CAGR}}{\text{MaxDD}}$$

Maximum Drawdown:

$$\text{MaxDD} = \max_t \left(\frac{\max_{s \le t} V_s - V_t}{\max_{s \le t} V_s}\right)$$

Tail Ratio:

$$\text{Tail Ratio} = \frac{\text{95th percentile return}}{|\text{5th percentile return}|}$$

Annualized Turnover:

$$\text{Turnover}_{\text{annual}} = \overline{\text{TO}_{\text{daily}}} \cdot 252$$

Appendix C: Statistical Test Details

Diebold-Mariano Test:

Null hypothesis: $E[d_t] = 0$ where $d_t = -r_t^{\text{RL}} + r_t^{\text{baseline}}$

Test statistic:

$$\text{DM} = \frac{\bar{d}}{\sqrt{\text{Var}(\bar{d})}}$$

We use Newey-West HAC standard errors with lag $h = \lceil T^{1/3} \rceil$.

Block Bootstrap:

1. Divide returns into overlapping blocks of length $L = 20$
2. Resample blocks with replacement (5000 iterations)
3. Compute Sharpe for each resample
4. Construct 95% CI from percentiles (2.5%, 97.5%)

End of Report

Author: Jai Ansh Bindra
GitHub Repository: https://github.com/JaiAnshSB26/deep-rl-rebalance