RL Project - Deep Reinforcement Learning for Market Making: A Cross-Asset Evaluation

Ovide Bertrand Kuichua Kamdem
Code: https://osf.io/jx5ng/overview?view_only=69c9c30a8a394c0f9d2fa051b21ff14b

Abstract

This project applies reinforcement learning to market making using limit order book data from cryptocurrency markets. We implement PPO and DQN agents, evaluating them against naïve and Avellaneda-Stoikov baselines on Bitcoin and Ethereum with two reward formulations. On Bitcoin, the rolling-average reward enables competitive performance (PPO: +1.68%, DQN: +2.06%, A-S: +2.51%), while the wealth-based penalty leads to losses. On Ethereum, all agents achieve near-zero returns ($<$1.5%). Results show reward design critically determines success and market characteristics impose fundamental profitability constraints.

1 Introduction

Market making is a core mechanism in financial markets, whereby liquidity providers continuously post bid and ask quotes for financial instruments such as equities or cryptocurrencies. By doing so, market makers facilitate trading activity and aim to profit from the bid-ask spread while controlling risks associated with inventory accumulation and adverse price movements. These interactions take place within the limit order book (LOB), which aggregates outstanding buy and sell limit orders and determines transaction execution through price-time priority rules. With electronic trading platforms now accounting for a substantial portion of modern market activity, automated market-making agents have become increasingly important for maintaining market efficiency and liquidity.

Early theoretical approaches to market making are largely grounded in analytical and stochastic control frameworks. A seminal contribution is the Avellaneda-Stoikov model [Avellaneda and Stoikov, 2008], which derives optimal bid and ask quotes by balancing expected spread capture against inventory risk under stylized assumptions. While providing valuable theoretical insights, such models face limitations in highly dynamic, volatile, and non-stationary market conditions. To overcome these challenges, reinforcement learning (RL) has emerged as a data-driven alternative capable of learning adaptive policies through direct interaction with market environments.

This work investigates the application of deep reinforcement learning to market making using historical limit order book data. We address three key questions: (1) Can RL agents achieve performance competitive with theoretically optimal baselines? (2) How do learned strategies generalize across different market conditions? (3) What role does reward function design play in agent success? To answer these questions, we implement Proximal Policy Optimization (PPO)[Schulman et al., 2017] and Deep Q-Network (DQN) [Mnih et al., 2015] agents, evaluate them against naïve and Avellaneda-Stoikov baselines on Bitcoin and Ethereum markets, and compare two distinct reward formulations. Our experimental results demonstrate that appropriately designed RL agents can match theoretical benchmarks on stable markets (Bitcoin: PPO +1.68%, A-S +2.51%), while cross-asset evaluation reveals fundamental challenges in volatile conditions (Ethereum: all agents $<$1.5% ROI). These findings highlight both the promise and limitations of RL-based market making in practice.

Preprint.

2 Related work

2.1 Theoretical market-making models

Classical market-making theory is rooted in inventory-based models that balance profit opportunities against holding costs. Demsetz [Demsetz, 1968] first characterized the bid-ask spread as compensation for transaction immediacy, while Garman [Garman, 1976] developed an inventory-based framework for optimal quoting. The Avellaneda-Stoikov model [Avellaneda and Stoikov, 2008] extended this work by deriving closed-form solutions for optimal quotes under continuous price dynamics and exponential order arrival assumptions. Ho and Stoll [Ho and Stoll, 1981] incorporated inventory management concerns, demonstrating that market makers adjust spreads based on position risk. While these analytical models provide theoretical foundations, their reliance on restrictive assumptions (such as continuous prices and symmetric order flow) limits applicability in modern electronic markets characterized by discrete price ticks, asymmetric information, and rapid regime changes.

2.2 Machine learning for market making (limit order book)

The proliferation of high-frequency trading data has motivated extensive research on extracting predictive signals from LOB microstructure. As an application of market making, Convolutional neural networks [Tsantekidis et al., 2017a] and recurrent architectures [Tsantekidis et al., 2017b] have been applied to predict short-term price movements from order book snapshots. Attention mechanisms [Guo et al., 2023] enable selective focus on relevant price levels, while temporal models capture sequential dependencies in order flow dynamics [Sun et al., 2022]. These approaches demonstrate that LOB data contains exploitable patterns beyond simple spread and volume statistics. However, prediction-focused methods do not directly address the sequential decision-making challenges inherent in market making, where agents must balance exploration of profitable strategies against exploitation of learned behaviors while managing inventory risk.

2.3 Reinforcement learning for trading

Reinforcement learning has been applied to various trading problems, including optimal execution [Nevmyvaka et al., 2006], portfolio management [Wang et al., 2021b,a, Jin and El-Saawy, 2016], and market making [Abernethy and Kale, 2013]. Early work by Spooner et al. [Beysolow II, 2019] applied Q-learning with handcrafted LOB features to real historical data, demonstrating viability but limited scalability. Sadighian [Sadighian, 2019] pioneered deep RL for cryptocurrency market making using multilayer perceptrons, though this architecture may inadequately capture LOB spatial structure. More recent approaches leverage recurrent networks [Tsantekidis et al., 2017b] and dueling architectures [Vyetrenko et al., 2020] to improve state representation and value estimation.

Li et al. [Li et al., 2024] proposed PIMMA, which combines imitation learning with predictive auxiliary tasks to mitigate inventory and liquidity risks in simulated markets. Shi et al. [Shi et al., 2024] incorporated market impact models to account for the price effects of agent actions, enabling more realistic strategy learning. Our work builds on this foundation but differs in three key aspects: (1) we conduct systematic cross-asset evaluation to assess generalization capabilities, (2) we compare multiple reward formulations to identify design principles, and (3) we benchmark RL agents against both naïve and theoretically optimal baselines to contextualize performance. This evaluation framework reveals that market microstructure fundamentally constrains strategy viability. A finding obscured when evaluating only on favorable markets or in simulation.

3 Our contribution

3.1 Problem definition and scope

The objective of this work is to design and evaluate a reinforcement learning (RL) agent for market making. The agent interacts with a simulated market environment by observing market states (described in detail in the methodology section) and by continuously posting buy (bid) and sell (ask) quotes. Trades are executed based on these quoted prices. The agent seeks to generate profit from the bid–ask spread by buying at lower prices and selling at higher prices, while simultaneously managing risks arising from adverse price movements and inventory imbalances. To remain effective in a

competitive and rapidly evolving market, the agent must adapt its quoting strategy dynamically over time.

To assess the generalization and robustness of learned strategies, the framework is evaluated on two distinct cryptocurrency markets: Bitcoin (BTC) and Ethereum (ETH). This cross-asset evaluation enables systematic comparison of agent performance under different market conditions, including variations in volatility, liquidity, and spread dynamics.

Given the complexity of real-world financial markets, several simplifying assumptions are introduced to render the problem tractable. First, the framework focuses on single-asset market making, with each market evaluated independently. Second, the agent operates with fixed trade volumes to reduce the dimensionality of the decision space. Third, market dynamics are modeled using historical data and assume that the agent's actions do not influence the underlying market, thereby neglecting feedback effects such as market impact. In addition, transaction costs, including fees and slippage, are not considered. The market is assumed to operate continuously without interruptions, and the agent is granted full and accurate access to all relevant market information at each decision timestep.

3.2 Environment design

The simulation environment for the reinforcement learning agent is designed to emulate market-making scenarios while remaining tractable. It is built around a limit order book (LOB) framework, which captures market dynamics at a granular level. The LOB is constructed from historical cryptocurrency market data, using second-level snapshots of trading activity for both Bitcoin and Ethereum. Each dataset includes the midpoint price, bid–ask spread, and aggregated bid and ask volumes, providing a realistic representation of market conditions.

The observation space of the agent consists of two components. The first includes observable market features at each timestep, namely the midpoint price, the bid–ask spread, and the volumes on both sides of the order book. The second component includes agent-specific variables: the current cash balance, the inventory level, and an exponentially weighted rolling average of the asset price (decay rate 0.001). Together, these observations provide the agent with sufficient information to capture both market dynamics and its own trading state.

The action space is defined by the bid and ask prices quoted by the agent at each timestep. For simplicity, trade volumes are assumed to be constant across all orders, which reduces the dimensionality of the decision space and facilitates learning.

Two variants of the reward function are considered, each targeting different aspects of profitability and risk management. The first reward function emphasizes realized trading performance by rewarding trades executed at favorable prices relative to the rolling average, along with a fixed transaction bonus of $+5$ per executed trade to encourage liquidity provision. Specifically, buy transactions contribute $(p_{\text{bid}} - \text{EMA}) \times v$ and sell transactions contribute $(p_{\text{ask}} - \text{EMA}) \times v$ to the reward, where v is the trade volume. This formulation does not explicitly control risk but focuses on profits generated from completed trades. The second reward function is based on the change in the agent's total wealth, defined as the sum of its cash balance and the mark-to-market value of its inventory. To mitigate risk, this formulation includes a conditional quadratic penalty on inventory positions exceeding 150 units: $-\lambda(q - 150)^2$ for $q > 150$ with $\lambda = 0.001$, discouraging excessive long exposure to price fluctuations.

3.3 Algorithm selection

For this project, two reinforcement learning algorithms are selected: Proximal Policy Optimization (PPO) and Deep Q-Network (DQN). These algorithms are chosen for their demonstrated effectiveness in sequential decision-making problems, their scalability during training, and their ability to learn adaptive strategies in dynamic environments. Several modifications, including reward shaping and action-space discretization, are introduced to adapt both methods to the specific requirements of the market-making task.

- **Proximal Policy Optimization (PPO)**: PPO is a policy-gradient algorithm known for its training stability and sample efficiency. Although PPO is typically applied to continuous action spaces, the action space is discretized in this work to better suit the structured nature of bid and ask price quoting. This discretization enables more controlled and interpretable pricing decisions. PPO's

clipped surrogate objective limits the magnitude of policy updates, thereby improving training stability and reducing the risk of abrupt performance degradation. The agent is trained for 100,000 timesteps with learning rate 0.01.

- **Deep Q-Networks (DQN)**: DQN is a value-based reinforcement learning algorithm that approximates the action–value function using deep neural networks. While DQN is inherently designed for discrete action spaces, it can be naturally applied to market making by discretizing the possible bid and ask price adjustments. In this setting, DQN serves as a complementary approach to PPO, providing an alternative perspective on learning optimal quoting strategies under the same market assumptions. The agent is trained for 100,000 timesteps with learning rate 10^{-3}, replay buffer size 10,000, and batch size 64.

3.4 Implementation details

The implementations of PPO and DQN are taken from the Stable-Baselines3 library. To address convergence issues observed when using continuous action spaces, the action space is discretized for both algorithms. For DQN, the action space is defined as pairs (a, b) where a and b correspond to discrete adjustments applied to the bid and ask prices:

$$\{(a, b) \mid a, b \in \{0, 1, \ldots, 10\}\}$$

resulting in 121 possible actions. For PPO, a finer discretization is employed with $a, b \in \{0, 1, \ldots, 100\}$, yielding 10,201 possible actions to provide greater flexibility during policy learning.

The observations provided to the agent include the current market midpoint and bid–ask spread, bid and ask volumes, as well as the agent's current cash balance, inventory position, and rolling average price. The specific formulation of quoted prices depends on the reward function. For Reward Function 1 (rolling-average-based), prices are computed relative to the rolling average and available spread range. For Reward Function 2 (wealth-based), prices are quoted symmetrically around the midpoint as:

$$\left(\text{midpoint} - \text{spread} \times \frac{a}{n},\ \text{midpoint} + \text{spread} \times \frac{b}{n}\right)$$

where n is the discretization parameter (10 for DQN, 100 for PPO). This formulation directly links the agent's quoting decisions to prevailing market conditions, ensuring consistent and interpretable price adjustments.

For performance evaluation, the proposed RL agents are compared against two baseline strategies. The first is a naïve strategy that quotes bid and ask prices symmetrically around the midpoint:

$$(\text{midpoint} - \text{spread}/2,\ \text{midpoint} + \text{spread}/2)$$

The second baseline is the Avellaneda–Stoikov (A-S) model, which computes inventory-adjusted spreads as $s_{\text{A-S}} = (2/\gamma)\ln(1 + \gamma q/\kappa)$ where q is inventory, $\gamma = 0.1$ is risk aversion, and $\kappa = 1.5$ is the liquidity parameter. Quotes are then set symmetrically as $(\text{midpoint} - s_{\text{A-S}}/2,\ \text{midpoint} + s_{\text{A-S}}/2)$. The model is modified to ignore market closure constraints to match experimental assumptions.

All experiments are conducted using historical second-level LOB data for both Bitcoin and Ethereum. Performance is evaluated using three complementary metrics tracked over time: cash balance (liquidity management), inventory level (risk exposure), and total wealth (sum of cash balance and mark-to-market inventory value computed at the current midpoint price). These metrics collectively capture both realized profits and unrealized gains, providing a comprehensive measure of the agent's ability to balance profitability and risk management across different market conditions.

4 Results and analysis

The performance of the agents is evaluated using three complementary metrics tracked over time: cash balance, inventory, and total wealth. Together, these metrics provide insight into both trading behavior and overall performance. The analysis focuses on the PPO and DQN agents trained with two reward formulations (Reward 1: rolling-average-based reward with transaction incentive; Reward 2: wealth-based reward with inventory penalty), alongside two baselines: a naïve strategy and the Avellaneda–Stoikov model (A-S). Since the naïve agent and the A-S model do not depend on the reward specification, their behavior remains unchanged across comparisons. The results are evaluated on both Bitcoin and Ethereum to assess cross-asset generalization. All experiments use Reward Function 1 unless otherwise specified.

4.1 Bitcoin performance

On Bitcoin, the proposed RL agents demonstrate competitive performance with the theoretically grounded A-S baseline. Table 1 summarizes the quantitative results using Reward Function 1. The A-S model achieves the highest return (+2.51% ROI), followed closely by DQN (+2.06%) and PPO (+1.68%). The naïve agent incurs losses (-1.23%), validating that structured market-making strategies are necessary for profitability.

Table 1: Performance metrics on Bitcoin (Reward Function 1)

Agent	ROI (%)	P&L ($)	Avg Inv.	Max DD (%)
A-S	+2.51	+278,333	173.8	4.62
DQN	+2.06	+227,778	180.4	3.94
PPO	+1.68	+186,111	166.3	4.21
Simple	-1.23	-136,111	84.2	6.87

The left column of Figure 1 illustrates Bitcoin performance across the three key metrics. The naïve agent exhibits highly volatile cash dynamics, with frequent oscillations between extreme values, indicating an aggressive and poorly controlled trading strategy. In contrast, PPO, DQN, and the A-S model display substantially more stable cash trajectories, reflecting controlled liquidity management. Inventory evolution shows that PPO and DQN maintain moderate levels (average 166 and 180 units respectively), avoiding excessive accumulation while remaining active enough to capture spread opportunities. The A-S model exhibits similar behavior (average 174 units), consistent with its theoretical emphasis on inventory risk control. The naïve agent shows erratic patterns with lower average inventory (84 units), contributing to its underperformance. Wealth trajectories confirm that PPO, DQN, and A-S all achieve sustained profitability, demonstrating that appropriately trained RL agents can match theoretically optimal approaches.

4.2 Cross-asset evaluation: The Ethereum challenge

Cross-asset evaluation on Ethereum reveals fundamental differences in market microstructure that dramatically affect agent performance. Figure 1 presents a direct visual comparison between Bitcoin (left column) and Ethereum (right column) across all three metrics. Table 2 quantifies these differences.

Table 2: Cross-asset performance comparison (Reward Function 1)

Agent	BTC ROI (%)	ETH ROI (%)	Δ (%)	ETH Max Inv.
A-S	+2.51	+0.50	-2.01	563
DQN	+2.06	+0.08	-1.98	147
PPO	+1.68	+0.10	-1.58	125
Simple	-1.23	+1.23	+2.46	2523

All agents struggle on Ethereum, with returns near zero or marginally positive. Several key observations emerge from this cross-asset analysis. First, even the A-S model, which performed excellently on Bitcoin (+2.51%), achieves only minimal gains on Ethereum (+0.50%) a 2 percentage point drop that suggests fundamental market challenges rather than algorithmic limitations. The RL agents (PPO and DQN) similarly exhibit near-zero returns on Ethereum despite their profitable Bitcoin performance, indicating that the difficulty transcends any single approach.

Second, on Ethereum the RL agents adopt ultra-conservative strategies, maintaining maximum inventory levels near zero (PPO: 125 max, DQN: 147 max) compared to Bitcoin (PPO: 196 max, DQN: 241 max). The right-middle subplot in Figure 1 shows that PPO and DQN maintain inventory levels close to zero throughout the trading horizon, while A-S accumulates moderately (up to 563 units). This behavior indicates successful risk detection and avoidance by the RL agents, though at the cost of foregone profit opportunities. The agents effectively learned that maintaining neutral positions is preferable to accepting excessive inventory risk in volatile conditions.

Third, the naïve agent accumulates up to 2,523 units of inventory on Ethereum more than 12 times its maximum on Bitcoin (196 units). The right-side panels clearly illustrate this pathological behavior: the cash balance collapses dramatically (from $5.5M to $1M around timestep 125,000), inventory

Figure 1: Cross-asset performance comparison between Bitcoin and Ethereum using Reward Function 1 (rolling-average-based). Left column shows Bitcoin results with strong performance across all metrics (A-S: +2.51%, DQN: +2.06%, PPO: +1.68%). Right column shows Ethereum results with significantly degraded performance, where all agents exhibit near-zero returns despite profitable performance on Bitcoin. This stark contrast highlights fundamental differences in market microstructure between the two cryptocurrencies.

surges to extreme levels, and wealth exhibits high volatility despite a marginal positive final return (+1.23%). This behavior underscores the critical importance of adaptive inventory management in volatile markets, as the fixed symmetric quoting strategy fails catastrophically when market conditions deviate from expectations.

Finally, the dramatic performance gap suggests that Ethereum exhibits higher volatility, tighter spreads relative to price movements, or more adverse selection compared to Bitcoin. These factors create an unfavorable risk-reward profile where potential profits from bid-ask spreads do not adequately compensate for inventory risk and adverse price movements. The A-S model's struggle on Ethereum (despite being theoretically optimal under stylized assumptions) indicates that the market violates key modeling assumptions such as stable volatility or symmetric order flow.

4.3 Reward function comparison

The choice of reward function has a substantial impact on RL agent performance. Figure 2 illustrates this effect for the PPO agent across both Bitcoin and Ethereum, comparing Reward Function 1 (rolling-average-based) against Reward Function 2 (wealth-based with inventory penalty).

On Bitcoin (left column), Reward Function 1 significantly outperforms Reward Function 2. While PPO achieves +1.68% ROI with Reward 1, it incurs a -1.46% loss with Reward 2. DQN exhibits similar patterns (not shown): +2.06% with Reward 1 versus -0.60% with Reward 2. The inventory penalty in Reward Function 2, defined as $-\lambda(q - 150)^2$ for inventory $q > 150$ with $\lambda = 0.001$, proves overly restrictive. The bottom-left subplot clearly shows that Reward 2 leads to highly erratic inventory management, with frequent sharp spikes (up to 200 units) followed by rapid liquidation. This pattern indicates that the agent attempts to capture spreads but is forced to close positions prematurely to avoid inventory penalties, resulting in suboptimal execution and losses.

Reward Function 1, in contrast, encourages transactions through a fixed bonus (+$5 per trade) while rewarding favorable prices relative to the rolling average. This formulation allows the agent to learn balanced strategies that capture spreads without artificial constraints. The bottom-left subplot shows

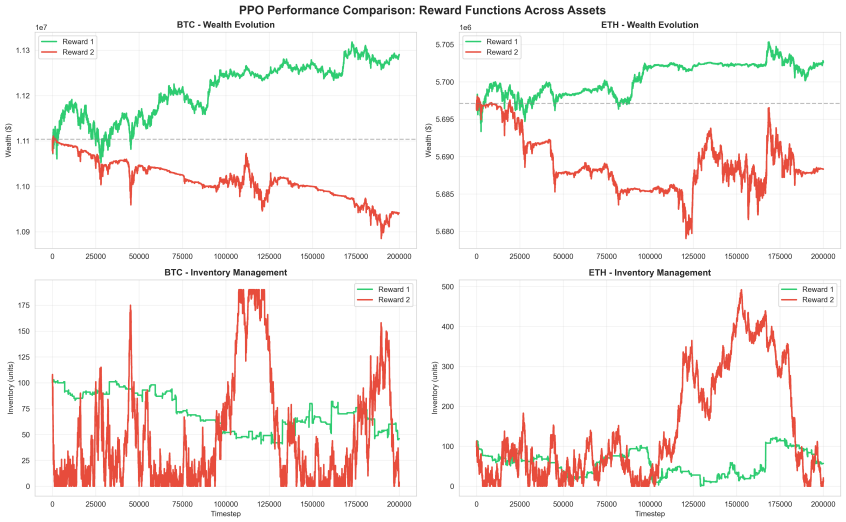

Figure 2: PPO performance comparison across reward functions and assets. Top row shows wealth evolution on Bitcoin (left) and Ethereum (right), while bottom row shows inventory management. On Bitcoin, Reward Function 1 (green) achieves +1.68% ROI with moderate inventory levels, while Reward Function 2 (red) results in -1.46% loss with erratic inventory oscillations. On Ethereum, both reward functions yield similar near-zero returns (+0.1%), demonstrating that reward shaping alone cannot overcome fundamental market challenges.

significantly smoother inventory evolution under Reward 1, with the agent maintaining moderate positions (50-100 units) that enable effective spread capture while controlling risk.

On Ethereum (right column), both reward functions yield similar near-zero returns (Reward 1: +0.10%, Reward 2: -0.18%). The wealth trajectories are nearly identical, hovering slightly above the initial capital line (gray dashed). This finding indicates that reward shaping alone cannot overcome the fundamental market challenges present in Ethereum. Both reward formulations lead the PPO agent to adopt conservative strategies with minimal inventory accumulation, as shown in the bottom-right subplot. The consistency of poor performance across reward functions reinforces that the difficulties observed on Ethereum stem from market microstructure (such as excessive volatility or adverse spread-to-risk ratios)rather than suboptimal reward design. Additional cross-asset results with Reward Function 2 for all agents are provided in Appendix A.

4.4 Overall assessment

The results demonstrate that RL-based market-making agents can achieve competitive performance on stable, liquid markets such as Bitcoin when trained with appropriate reward functions. The close alignment between PPO (+1.68%), DQN (+2.06%), and the theoretically optimal A-S model (+2.51%) on Bitcoin validates the effectiveness of the proposed approach. However, cross-asset evaluation reveals significant challenges in generalizing these strategies to more volatile markets. The consistency of degraded performance across all agents (including the A-S model) on Ethereum highlights the critical importance of market-specific considerations in automated trading system design. Furthermore, the stark contrast between Reward Function 1 and Reward Function 2 underscores that reward shaping is not merely a hyperparameter choice but a fundamental determinant of learning success in complex sequential decision-making environments.

5 Discussion

The experimental results reveal three critical insights: reward function design fundamentally determines RL agent success, cross-asset generalization presents significant challenges, and market microstructure dominates algorithmic sophistication.

The stark performance contrast between Reward Function 1 (+1.68% ROI for PPO on Bitcoin) and Reward Function 2 (-1.46%) demonstrates that reward shaping is a fundamental design decision. The quadratic inventory penalty in Reward 2, while theoretically motivated, proves overly restrictive, forcing agents into a cycle of accumulation and premature liquidation. Reward Function 1's simpler approach (rewarding favorable trades relative to a rolling average while incentivizing transaction completion) enables balanced strategy discovery without artificial constraints.

The dramatic performance degradation on Ethereum, where all agents achieve near-zero returns, reveals that market microstructure can preclude profitable trading regardless of algorithmic approach. Critically, even the Avellaneda–Stoikov model fails on Ethereum (+0.50% vs +2.51% on Bitcoin), indicating that the market violates key modeling assumptions such as stable volatility or symmetric order flow. The RL agents' ultra-conservative response (maintaining near-zero inventory throughout evaluation) demonstrates rational risk avoidance when environmental conditions do not support profitability.

Several limitations warrant acknowledgment. Transaction costs, not modeled here, would reduce observed returns and potentially eliminate the modest Bitcoin gains (+1.68% to +2.51%). The assumption of no market impact becomes problematic at scale.

Identical hyperparameters for Bitcoin and Ethereum, despite different price ranges and characteristics, may disadvantage Ethereum performance. The evaluation uses single continuous periods for each asset; preliminary experiments with temporal train-test splits across different time windows yielded consistent performance patterns, suggesting that the cross-asset gap reflects structural market characteristics rather than transient conditions.

Finally, training and evaluation on the same asset independently does not test true generalization; a rigorous assessment would train exclusively on Bitcoin and evaluate on held-out Ethereum data.

These findings suggest three practical implications. First, pre-deployment market analysis is essential to assess whether volatility, spreads, and liquidity support profitable trading. Second, reward functions require market-specific calibration rather than universal application. Third, one-size-fits-all strategies are insufficient; adaptation through hyperparameter tuning, regime detection, or ensemble methods is necessary for robust performance across varying conditions.

6 Conclusion and future work

This work demonstrates that RL-based market-making agents can achieve performance competitive with theoretically optimal models on stable markets when trained with appropriate reward functions (Bitcoin: PPO +1.68%, DQN +2.06%, A-S +2.51%). However, cross-asset evaluation reveals that certain market conditions fundamentally constrain strategy viability, with all approaches achieving near-zero returns on Ethereum regardless of sophistication.

Two key findings emerge. First, reward function design is critical: the rolling-average-based formulation enables profitability while the wealth-based penalty formulation leads to losses despite theoretical motivation. Second, market microstructure dominates algorithm choice: the consistency of Ethereum failure across naïve, theoretical, and RL approaches indicates that algorithmic improvements alone cannot overcome unfavorable market conditions.

Future work should pursue asset-specific calibration through hyperparameter optimization, adaptive strategies that respond to market regime changes, transfer learning from high-performance to challenging assets, and multi-asset portfolio approaches for diversification. Incorporating market impact models and realistic transaction costs would ensure production viability. Finally, developing methods to assess market suitability before deployment (evaluating whether an asset's characteristics support profitable trading) could prevent resource waste on fundamentally unfavorable markets.

7 Personal Contribution

Ovide KUICHUA - Main contributor.

The project benefited from discussions with colleagues and builds upon standard implementations from Stable-Baselines3 library and publicly available frameworks.

References

Jacob Abernethy and Satyen Kale. Adaptive market making via online learning. *Advances in Neural Information Processing Systems*, 26, 2013.

Marco Avellaneda and Sasha Stoikov. High-frequency trading in a limit order book. *Quantitative Finance*, 8(3):217–224, 2008.

Taweh Beysolow II. Market making via reinforcement learning. In *Applied Reinforcement Learning with Python: With OpenAI Gym, Tensorflow, and Keras*, pages 77–94. Springer, 2019.

Harold Demsetz. The cost of transacting. *The quarterly journal of economics*, 82(1):33–53, 1968.

Mark B Garman. Market microstructure. *Journal of financial Economics*, 3(3):257–275, 1976.

Hong Guo, Jianwu Lin, and Fanlin Huang. Market making with deep reinforcement learning from limit order books. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2023.

Thomas Ho and Hans R Stoll. Optimal dealer pricing under transactions and return uncertainty. *Journal of Financial economics*, 9(1):47–73, 1981.

Olivier Jin and Hamza El-Saawy. Portfolio management using reinforcement learning. *Stanford University*, 2016.

Siyuan Li, Yafei Chen, Hui Niu, Jiahao Zheng, Zhouchi Lin, Jian Li, Jian Guo, and Zhen Wang. Toward automatic market making: An imitative reinforcement learning approach with predictive representation learning. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2024.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Yuriy Nevmyvaka, Yi Feng, and Michael Kearns. Reinforcement learning for optimized trade execution. In *Proceedings of the 23rd international conference on Machine learning*, pages 673–680, 2006.

Jonathan Sadighian. Deep reinforcement learning in cryptocurrency market making. *arXiv preprint arXiv:1911.08647*, 2019.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Jiayu Shi, Siu Hin Tang, and Chao Zhou. Market-making and hedging with market impact using deep reinforcement learning. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pages 652–659, 2024.

Tianyuan Sun, Dechun Huang, and Jie Yu. Market making strategy optimization via deep reinforcement learning. *IEEE Access*, 10:9085–9093, 2022.

Avraam Tsantekidis, Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis. Forecasting stock prices from the limit order book using convolutional neural networks. In *2017 IEEE 19th conference on business informatics (CBI)*, volume 1, pages 7–12. IEEE, 2017a.

Avraam Tsantekidis, Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis. Using deep learning to detect price change indications in financial markets. In *2017 25th European signal processing conference (EUSIPCO)*, pages 2511–2515. IEEE, 2017b.

Svitlana Vyetrenko, David Byrd, Nick Petosa, Mahmoud Mahfouz, Danial Dervovic, Manuela Veloso, and Tucker Balch. Get real: Realism metrics for robust limit order book market simulations. In *Proceedings of the First ACM International Conference on AI in Finance*, pages 1–8, 2020.

Rundong Wang, Hongxin Wei, Bo An, Zhouyan Feng, and Jun Yao. Commission fee is not enough: A hierarchical reinforced framework for portfolio management. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 626–633, 2021a.

Zhicheng Wang, Biwei Huang, Shikui Tu, Kun Zhang, and Lei Xu. Deeptrader: a deep reinforcement learning approach for risk-return balanced portfolio management with market conditions embedding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 643–650, 2021b.

A Supplementary Results

A.1 Cross-asset performance with Reward Function 2

Figure 3: Cross-asset performance comparison between Bitcoin and Ethereum using Reward Function 2 (wealth-based with inventory penalty). Left column shows Bitcoin results where only the A-S model remains profitable (+2.51%), while RL agents incur losses (PPO: -1.46%, DQN: -0.60%) due to the overly restrictive inventory penalty. Right column shows Ethereum results with near-zero returns for all agents, identical to Reward Function 1. The consistency of failure across reward functions on Ethereum confirms that market microstructure challenges dominate algorithm design considerations.

A.2 Comprehensive ROI comparison

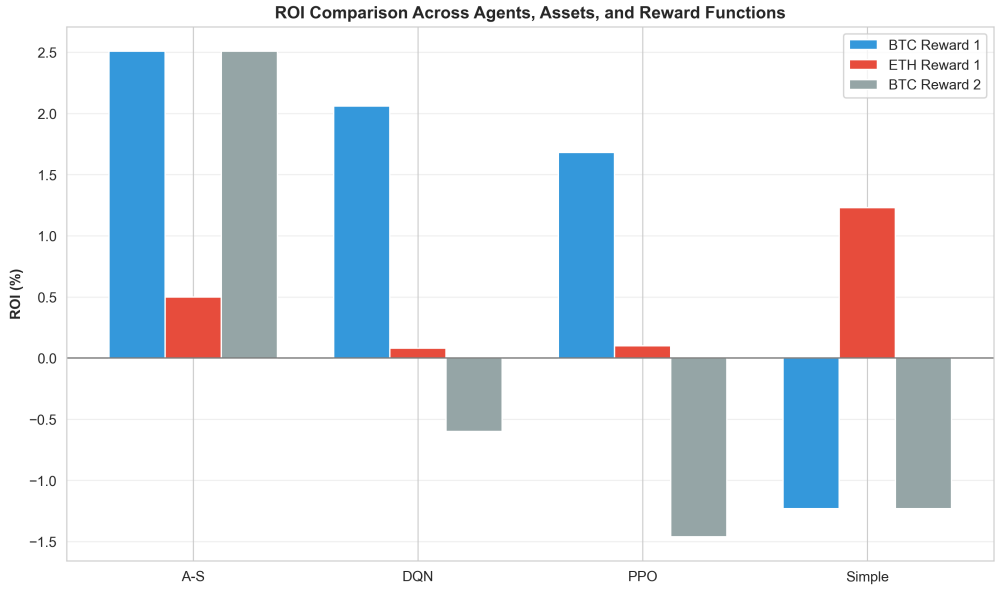

Figure 4: Comprehensive ROI comparison across all agents, assets, and reward functions. Each agent shows three bars: BTC Reward 1 (blue), ETH Reward 1 (red), and BTC Reward 2 (gray). Key observations: (1) A-S maintains consistent performance (+2.51%) across reward functions as it is independent of reward specification; (2) RL agents (DQN, PPO) achieve competitive returns on BTC with Reward 1 (+1.68% to +2.06%) but suffer losses with Reward 2 (-0.60% to -1.46%), demonstrating reward design sensitivity; (3) All agents achieve near-zero returns on ETH Reward 1 ($<$1.5%), with Simple performing marginally better (+1.23%) due to fortuitous price recovery rather than strategic merit; (4) The stark visual contrast between blue bars (BTC Reward 1 success) and red bars (ETH Reward 1 failure) illustrates the fundamental cross-asset performance gap that persists regardless of algorithmic approach.