Neuro-Market Makers: Adaptive Strategies with Limit Order Book

Market Microstructure (BM51002)

Arshdeep Singh, Rajeev Joshi, Rishikesh Maurya, Mehak

Abstract

Market making (MM) is essential for liquidity and stability in modern financial markets, requiring market makers to continually adjust buy and sell quotes to optimize profit and control risk. A key challenge lies in analyzing the vast, noisy, and highly dynamic data from the limit order book (LOB), which captures real-time bid and ask orders. Conventional MM approaches depend heavily on rigid assumptions about market behavior, while prior reinforcement learning (RL) methods have often been constrained by handcrafted features and limited action spaces, reducing flexibility in real-world scenarios.

In this work, we trading propose a groundbreaking RL-based agent that autonomously learns to navigate and adapt to market volatility using raw LOB data. Our model features a CNN-Attention network (Attn-LOB) that automates deep feature extraction, revealing intricate market patterns without predefined rules. To enhance flexibility, we design a continuous action space tailored to real trading, allowing for nuanced bid-ask adjustments. We further introduce a hybrid reward function that integrates profit, inventory control, and execution efficiency, creating a balanced approach to market making. Comprehensive evaluations in a simulated trading environment demonstrate that our agent surpasses traditional benchmarks and other RL models, showing resilience to latency, interpretability in strategy, and superior profit control. These results reveal the agent's potential to redefine adaptive market making in high frequency.

Introduction

Market making (MM) plays a pivotal role in financial markets by ensuring liquidity and facilitating efficient price discovery. Market makers continuously place buy (bid) and sell (ask) orders in an asset, seeking to profit from the spread between these quotes. By bridging gaps between buyers and sellers, market makers reduce transaction costs, enhance market depth, and enable smoother, more predictable price movements—critical factors for a stable trading environment. In essence, they not only facilitate trade execution but also contribute significantly to the price formation process by narrowing bid-ask spreads. The financial incentives for market makers come with inherent costs and risks. These include *adverse selection costs* (45%), incurred when market makers trade against informed participants; *order processing costs* (45%), which encompass regulatory and exchange fees; and *inventory holding costs* (10%), the risk associated with holding unbalanced positions that can lead to losses if asset prices shift unfavorably. Managing these costs requires sophisticated strategies that can dynamically adapt to market volatility and liquidity. Modern financial markets primarily operate through electronic Limit Order Books (LOBs), which record all active buy and sell orders, organized by price level. The LOB serves as a real-time ledger, enabling participants to assess supply, demand, and liquidity for any given asset. Research has shown that LOB data contributes to price discovery by offering insights into market trends and the behavior of other participants. However, LOB data is inherently high-dimensional and noisy, making it difficult to identify useful patterns. This complexity, coupled with the fast-paced environment of high-frequency trading (HFT), poses significant challenges for developing effective market-making strategies. Traditional MM models, such as those by Avellaneda-Stoikov, rely on rigid assumptions about price processes and order arrival rates, which limit their adaptability to rapid market changes. Reinforcement Learning (RL) offers a promising alternative by allowing an agent to learn optimal strategies directly from data, adjusting dynamically to evolving market conditions. However, prior RL approaches often depend on predefined market features and use discrete action spaces, which limit flexibility and granularity in response to market signals.

In this project, we propose an adaptive RL agent for market making within a simulated environment, capable of continuous learning from raw LOB data. To effectively capture the intricate patterns within LOBs, we employ a CNN-Attention-based network (Attn-LOB) for feature extraction, removing the need for manual feature engineering. Our model's design tackles three main challenges in modern market making: (1) robust extraction of meaningful features from high-noise LOB data, (2) defining a flexible, continuous action space that aligns with financial market practices, and (3) constructing a reward function that balances profitability with inventory and risk management. This approach aims to develop a highly responsive agent that adapts to real-time market conditions, positioning it as a valuable tool for modern finance.

Literature Review

The conceptual foundation of market making began with Demsetz [1], who argued that the bid-ask spread serves as compensation for the immediacy offered in transactions. This idea laid the groundwork for subsequent models that refined the understanding of market microstructure. Garman [2] introduced an inventory-based approach that accounted for the risks associated with holding inventory while managing trade flows, setting the stage for models that extended to scenarios involving both individual and multiple market makers [3], [4]. Stochastic dynamic programming became a prominent tool for developing these models, exemplified by the influential work of Avellaneda and Stoikov [5]. In their study, *High-Frequency Trading in a Limit Order Book*, they calibrated optimal quotes by incorporating execution probability as a function of distance from the mid-price, building on earlier research by Ho and Stoll [4], [6]. However, the continuous price assumption in the Avellaneda-Stoikov model proved limiting due to the reality of discrete price ticks. This issue was addressed by Guilbaud and Pham [7], who constructed strategies better suited for discrete pricing in their work *Optimal High-Frequency Trading with Limit and Market Orders*.

Traditional models often required assumptions regarding price processes and order arrivals to derive closed-form solutions, as noted by Chan and Shelton [8] in *An Electronic Market-Maker*. These simplifications, while aiding in mathematical tractability, restricted the adaptability of the models. The advent of powerful computational tools and the availability of high-frequency limit order book (LOB) data recorded at millisecond intervals [9] propelled deeper studies into market microstructure. Notable contributions in this space include Kyle [10] in *Continuous Auctions and Insider Trading*, Glosten [11] in *Is the Electronic Open Limit Order Book Inevitable?* and O'Hara [12] in *Market Microstructure Theory*, all of whom facilitated deeper explorations into LOB dynamics and the complexities of trading environments, enabling a departure from the restrictive assumptions of earlier models.

Machine learning has increasingly contributed to financial research, with deep learning models such as CNNs [13] and RNNs [14] being utilized to extract features from LOB data for better price movement predictions. Temporal attention mechanisms have further refined these techniques, as demonstrated by Tran et al. [15] in *Temporal Attention-Augmented Bilinear Network for Financial Time-Series Data Analysis* and Shabani et al. [16] in *Multi-Head Temporal Attention-Augmented Bilinear Network for Financial Time Series Prediction*, enabling models to focus on key periods within the data and improve predictive performance.

Reinforcement learning (RL) has emerged as a powerful tool across various financial applications, including trading strategies, optimal execution, and portfolio management. Moody and Saffell [17], in their influential paper *Reinforcement Learning for Trading*, pioneered the use of RL for trading strategy optimization, aligning rewards with financial outcomes. They expanded on this with *Learning to Trade via Direct Reinforcement* [18], which emphasized the direct learning of trading strategies using RL.

Deng et al. [19], in *Deep Direct Reinforcement Learning for Financial Signal Representation and Trading*, demonstrated the application of deep reinforcement learning (DRL) in processing complex financial data for improved trading strategies. This study showcased the advantages of deep learning in capturing intricate patterns within financial time series and optimizing trading decisions.

Nevmyvaka, Feng, and Kearns [20], in their work *Reinforcement Learning for Optimized Trade Execution*, extended RL to the realm of trade execution, presenting a model that minimizes trading costs and slippage by leveraging data-driven decision-making. Wei et al. [21] advanced this domain with a model-based RL framework that combined predictive modeling of future prices with control mechanisms for LOBs, providing a comprehensive trading strategy.

Briola et al. [22], in *Deep Reinforcement Learning for Active High-Frequency Trading*, highlighted the use of RL in high-frequency trading (HFT). Their research demonstrated how sophisticated RL algorithms could adapt to rapid market changes and make real-time trading decisions, crucial for the high-frequency domain.

Abernethy and Kale [23] were among the first to apply online learning techniques to market making in their study *Adaptive Market Making via Online Learning*, which laid the groundwork for future RL-based market-making models. Lim and Gorse [24] utilized non-deep RL methods for simulated data, focusing on state representations involving inventory and remaining time. Spooner et al. [25] applied non-deep RL to real historical data and incorporated handcrafted LOB features such as spread, price movements, and order imbalance. Zhong [26] introduced market volatility and mid-price movement features to enhance RL applications in market making, while Sadighian [27] pioneered the use of DRL in cryptocurrency market making, utilizing a multi-layer perceptron (MLP) to represent market features, albeit with limitations in describing the full LOB structure.

Gasperov and Kostanjcar [28] integrated gradient boosting models and LSTMs for predicting price trends and feeding these predictions into an RL agent, showcasing a hybrid approach to market making. Recent works by Kumar [29] in *Deep Reinforcement Learning for Market Making* and Xu et al. [30] in *Performance of Deep Reinforcement Learning for High-Frequency Market Making on Actual Tick Data* utilized advanced architectures like Deep Recurrent Q-Networks (DRQN) and Dueling Double Deep Q Networks (D3QN) for HFT applications. While these models provided enhanced capabilities for making high-frequency trading decisions, they often did not fully incorporate detailed LOB characteristics, suggesting the need for more comprehensive integration of such data.

Core Objective

The core objective of the project, titled *"Neuro-Market Makers: Adaptive Strategies with Limit Order Book"* is to develop an advanced market-making agent using deep reinforcement learning (DRL) to operate effectively in high-frequency trading (HFT) environments. The key objectives of this project are:

1. Design an Adaptive Market-Making Agent: Develop a model-free DRL approach that continuously optimizes bid and ask quotes without relying on the strong assumptions typical of traditional market-making models. The agent aims to provide liquidity, capture spreads, and manage inventory risk effectively.
2. Develop a CNN-Attention Based Network (Attn-LOB): Create a novel neural network architecture that combines convolutional neural networks and attention mechanisms for automatic feature extraction from limit order book (LOB) data. Pre-train this network on a mid-price direction prediction task to ensure effective market representation.
3. Introduce a Novel Continuous Action Space: Implement a continuous action space for training the RL agent, closely mirroring real-world financial practices. This action space aims to allow the agent to make more precise and realistic trading decisions compared to previously used discrete action spaces.
4. Design a Hybrid Reward Function: Formulate and test multiple reward functions focused on capturing spreads and controlling inventory. Develop a hybrid reward function that balances these objectives and demonstrates its effectiveness through theoretical analysis and experimental validation.
5. Ensure Interpretability and Practicality: Conduct interpretability experiments to demonstrate the agent’s ability to focus on relevant market changes and learn fundamental market-making skills, highlighting its practical potential in real-world trading scenarios.
6. Evaluate Using Scenario-Specific Metrics and Simulated Backtesting: Establish a simulated backtesting paradigm to rigorously evaluate the RL agent's performance using various financial metrics such as latency, inventory levels, and the Sharpe ratio. Ensure the agent's robustness and applicability in high-frequency trading environments through comprehensive testing.

Dataset & Methodology

For this project, we employ high-frequency historical limit order book (LOB) data from five major Indian companies across diverse sectors, offering a representative sample of the Indian stock market's characteristics and dynamics. The chosen companies include high-capitalization leaders and industry-defining stocks, each bringing unique trading patterns, volatility profiles, and liquidity conditions. This diversity enables a rigorous evaluation of the proposed market-making agent's adaptability and efficiency. The selected companies are:

1. **Reliance Industries Ltd.** A conglomerate and one of India's largest publicly traded companies, Reliance operates across sectors such as energy, petrochemicals, telecommunications, and retail. Its stock is characterized by high liquidity, frequent institutional trading, and relatively stable spreads.
2. **State Bank of India** as India's largest public sector bank, SBI's stock provides a glimpse into financial sector order flows, known for relatively steady trading volumes punctuated by sudden bursts due to economic announcements or policy changes. Banking stocks like SBI are sensitive to macroeconomic indicators, making this data valuable for observing the agent's adaptability in response to external economic events.
3. **Infosys Ltd.** Representing the IT and services sector, Infosys is globally recognized and subject to high trading volumes from international investors. IT stocks exhibit periodic fluctuations based on global technology trends and earnings reports, which introduce unique patterns in the LOB data. This inclusion helps assess how the agent handles sectors that are influenced by global markets and react strongly to quarterly performance announcements.

These datasets cover a trading period of one month in 2023, accounting for approximately 21 trading days. Each dataset is recorded event-by-event, capturing every change in price, volume, and order rearrangement within the LOB. Together, these five stocks contribute over 7 million data samples, enabling a robust assessment of the agent's performance across sectors with varying liquidity and volatility characteristics.

Data Preprocessing and Feature Engineering

1. **Normalization and Standardization of Order Book Data**:
    a. **Min-Max Normalization** and **Z-score Normalization** are applied to standardize bid-ask prices and volumes across the five stocks, given their wide range of price levels and trading volumes. The prices at each level are normalized relative to the mid-price, allowing the agent to operate independently of absolute price values and focus on the relative positions within the LOB. **LOB Normalization** Each bid and ask price is normalized by dividing by the mid-price and adjusted to center around zero, enhancing the model's ability to interpret and react to changes in relative prices rather than absolute values.
2. **Event-Based Labeling for Trend Prediction**:
    a. To enable trend prediction within the agent, each time step is labeled based on the price movement over a defined future horizon. This label categorizes price behavior as "up," "down," or "stationary," based on percentage change, and is one-hot encoded. This labeling guides the agent's learning to anticipate and react to trends in the LOB.
3. **Feature Extraction**:

a. **Realized Volatility (RV)** RV is computed over multiple time horizons (e.g., 5, 10, and 30 minutes) to capture both rapid and gradual price changes. This volatility measure provides insights into the asset's risk level and helps the agent identify and adjust to periods of high price uncertainty.
b. **Relative Strength Index (RSI)** RSI is included to capture momentum in price movements, indicating whether a stock is overbought or oversold within specific time frames, and aiding in timing buy/sell decisions.
c. **Order Strength Index (OSI)** This index measures the relative strength of buy vs. sell orders for both market and limit orders, along with withdrawal intensities. By analyzing OSI, the agent can interpret order flow dynamics and anticipate potential directional shifts in price.

4. **Data Structure and Formatting**
   a. The processed dataset is organized such that price and volume features for bid and ask levels are alternated, maintaining consistency across all 10 levels. This structured representation is followed by dynamic indicators, including OSI, RSI, and RV metrics, allowing the agent to learn complex dependencies within the LOB. This ordering not only aligns with the structure of typical LOBs but also provides the agent with temporal patterns that are essential for capturing the intricacies of market behavior across sectors.

Methodology

A. Market Representation

1. **Order Book State**: A snapshot of the order book at a given time t can be simplified into a one-dimensional vector of length 4×n, where n is the number of levels in the order book. The vector is constructed as:

$$LOB_t = \left\{ P_{i,t}^{\text{ask}}, V_{i,t}^{\text{ask}}, P_{i,t}^{\text{bid}}, V_{i,t}^{\text{bid}} \right\}_{i=1}^{n}$$

Here, Paski, t and Pbidi ,t represents the prices at the i-th ask and bid level at time t, respectively, and Vaski ,t and Vbidi ,t represent the corresponding volumes at those levels. A rolling window of length T is used to form the input, resulting in a vector of shape (T,4×n). Since handcrafted features are limited and inadequate for capturing high-frequency market dynamics, a pretrained deep neural network is used as a function approximator during reinforcement learning.

2. Dynamic Market State: The order book only provides static information about the market. To capture the dynamic nature of the market, we design a new indicator and leverage two commonly used indicators to obtain a more comprehensive view of the market state:

- **Order Strength Index (OSI)**: To account for the dynamic impact of market events, we introduce the OSI to measure the relative strength between the bid and ask sides of the order book. Events such as new market orders, new limit orders, and order cancellations influence the market. The OSI is calculated for both volume and order count, using the following formula for volume-based OSI:

$$OSI_v = \frac{\sum V_{buy} - \sum V_{sell}}{\sum V_{buy} + \sum V_{sell}}$$

Where V represents volume. OSI is also calculated based on the number of orders. The OSI is computed over rolling windows of 10s, 60s, and 300s, resulting in 18 features.

- **Realized Volatility (RV)**: RV measures the variation in asset prices by analyzing historical returns over a specific time period. It is computed using the following formula:

$$RV = \sqrt{\sum_{t=1}^{n} (\log p_t - \log p_{t-1})^2}$$

Where pt denotes the price at time t. RV is calculated for the past 5 minutes, 10 minutes, and 30 minutes, yielding 3 features.

- **Relative Strength Index (RSI):** It is a technical indicator commonly used in momentum trading. It assesses the speed of recent price changes of a security to determine whether the security is overbought or oversold, helping to identify potentially overvalued or undervalued conditions. The RSI is calculated as follows:

$$RSI = \frac{\sum_{t=1}^{n} Gain_t}{\sum_{t=1}^{n} Gain_t + \sum_{t=1}^{n} Loss_t}$$

3. Agent State: Agent state contains inventory and time factor. Inventory is a critical internal state to control the inventory risk. Remaining Time is calculated by current time t divided by total time T.

B. Limit Order Book Modeling

We employ a CNN-Attention based network to extract features from the LOB data, enhancing its performance through pre-training [49]. A common pre-training task in this domain is the prediction of mid-price direction, which involves classifying the future price trend (up, down, or stationery) over a specific horizon based on the limit order book data from the past t timestamps.

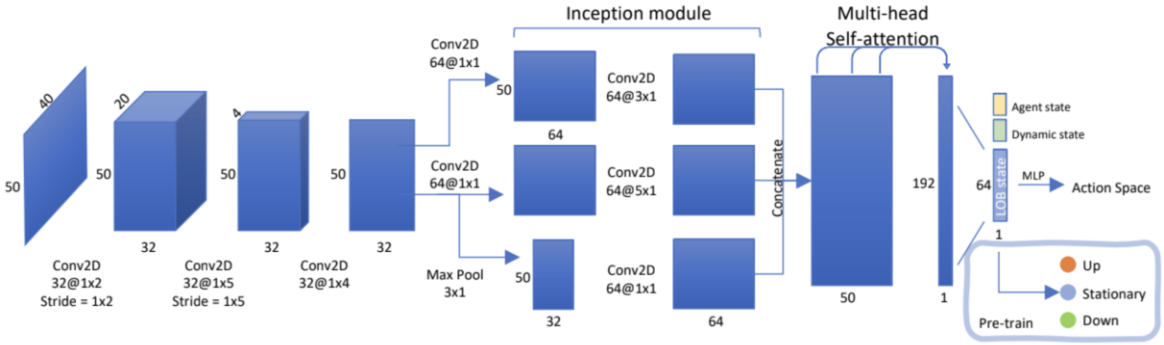


Fig. 1. The model architecture.

1. Label Acquisition: We use the approach proposed to obtain labels, which have proved to be more robust:

$$y_t = \begin{cases} 1 & \text{if } l_t > \alpha \\ 0 & \text{if } -\alpha \leq l_t \leq \alpha \\ -1 & \text{if } l_t < -\alpha \end{cases}$$

where α is a threshold hyperparameter which can be set according to the market. And lt can be calculated by

$$l_t = \frac{m_+(t) - m_-(t)}{m_-(t)}$$

$$m_\pm(t) = \frac{1}{k} \sum_{i=0}^{k} p_{t \mp i}$$

2. Normalization: Asset prices can fluctuate to unprecedented levels, causing significant changes in price statistics over time and making the price time series non-stationary. To transform the non-stationary input sequence into a stationary one, we apply the normalization method described in. Afterward, we perform z-normalization on the stationary price sequence and max-normalization on the volume sequence.

3. Model: The model we use, Attn-LOB, is a deep neural network consisting of convolutional layers and self-attention mechanisms. The architecture is based on a proposed CNN-LSTM-based network. We enhance it by adding a multi-head self-attention layer to capture temporal dependencies. The network architecture, as shown in Figure 1, takes input of shape (T * (4 * n)). Initially, multiple convolutional filters model the spatial dependencies. The output, with shape (T * {hidden dim}), is then passed through an Inception module to capture multi-scale temporal dependencies. Finally, a multi-head self-attention layer aggregates the temporal dependencies.

C. Action Space

In a typical market-making strategy, the goal is to place limit orders on both the buy and sell sides of the market. The agent in this case is limited to submitting one buy order and one sell order, without the ability to exit the market. This approach is similar to the one used in [50]. Here, we define two types of action spaces: discrete and continuous.

1. **Discrete Action Space**: This action space, which has been used in previous research, includes 8 possible actions. Actions 0-7 involve quoting a pair of orders with specific spreads and biases. Action 7

specifically represents closing the position with market orders. The ask and bid prices for the orders are determined based on the price at the time the orders are placed, and the volume of each order is fixed at the minimum trade unit, which is 100 in the China Stock Market.

2. **Continuous Action Space**: The continuous action space offers more flexibility than the discrete action space. This design is inspired by the AS strategy [9], where optimal ask and bid quotes are determined by calculating a reservation price (pr) and a quote spread. The reservation price (pr) represents the "true" price of the asset [8], and the agent places symmetric orders around this price.

D. Reward Function

1. Dampened PnL: The reward function is the key to guiding reinforcement learning. A natural thinking is to use Profit and Loss (PnL) as the reward function, which refers to the difference of value of the agent in Δt:

$$\Delta PnL_t = value_t - value_{t-\Delta t}$$

Where value is calculated by the sum of inventory's value (calculated with current mid-price) and cash. Spooner et al. has proved that only using PnL as the reward function will lead to the agent's speculative action. The agent tends to hold a large amount of inventory, which brings a great risk of drawdown when the trend market comes. To avoid this, they added an asymmetric punishment term to calculate the Dampened PnL (DP):

2. Trading PnL: Trading PnL (TP) rewards the agent only at transaction Xt = (Xt,p, Xt,v), where Xt,p denotes the transaction price and Xt,v denotes the transaction volume. TP is defined as:

$$DP_t = \Delta PnL_t - max(0, \eta * \Delta PnL_t)$$

Where pm,t denotes the mid-price at time t. Xt,v is positive for buying and negative for selling. It can be viewed as an advantage over the mid-price when trading. We can prove that T rading P nL is equivalent to ΔP nL subtract Holding P nL, the latter is PnL caused by positions, defined by Holding P nL = Invt−Δt ∗ (pm,t − pm,t−Δt). In short, T rading P nL can reward the price advantage of trading rather than the profit or loss of inventory.

3. Inventory Punishment: The inventory punishment (IP) is an effective way to control inventory risk:

$$IP_t = \zeta * Inv_t^2$$

4. Reward Function: Finally, we combine these rewards above to formulate a hybrid reward function as DPt are used to punish loss from holding inventory, T Pt are used to reward advantageous price of trading, and IPt are used to punish the high inventory position.

$$R_t = DP_t + TP_t - IP_t$$

IV. EXPERIMENTS

A. Dataset

The data used for this experiment comprises historical limit order book (LOB) data from an electronic trading market. The dataset contains a single day's trading data sampled at event-by-event updates, where events reflect changes in price, volume, or the structure of orders in the order book. The LOB data includes the top 10 bid and ask levels, which cover price and volume for each level.

B. Pre-training

1) Experimental Setup

The experiments were conducted on a high-performance computing system with specifications similar to those used in benchmark research: Intel Xeon CPUs and ample RAM for efficient processing. The dataset was split into training, validation, and testing sets with the training data comprising 60% of the day and the testing data using the remaining 40%. The data was processed within trading hours considered stable (e.g., 10:00 to 14:30 with breaks). The pre-training setup followed the structure defined in the methodology with a time window T of 50 and a prediction horizon k set to 10 events.

2) Baselines

The performance of the proposed Attn-LOB model was compared to baseline models:

- **FC-LOB**: A multi-layer perceptron with a hidden layer structure of (1024, 256, 64, 3) and Leaky ReLU activations, Softmax in the output.

- **Conv-LOB**: A fully convolutional model designed to process longer sequences with dilated convolutions.
- **DeepLOB**: A proven benchmark network that leverages CNN and LSTM layers for feature extraction.
- **Attn-LOB**: The model described in our methodology using a CNN-Attention hybrid for efficient feature extraction.

Model	Precision	Recall	F1	Parameters	Input Shape
FC-LOB	0.603	0.540	0.560	255,000	4000*1
Conv-LOB	0.585	0.500	0.530	170,000	1024*40
DeepLOB	0.780	0.670	0.710	140,000	100*40
Attn-LOB	0.760	0.700	0.730	175,000	50*40

C. RL Settings

1) Simulator

A custom trading environment was developed to simulate the market using historical LOB data. The simulator executed orders based on actual market conditions, updating the agent's inventory and cash balance accordingly. The agent's orders were only executed if they matched existing levels in the LOB. No transaction costs were considered, and the agent could hold negative cash or inventory. Hyperparameters were set as $\omega=10$, $T=50$, $\eta=0.5$, and $\zeta=0.01$.

2) Experimental Setup

Each trading day was divided into episodes of 2000 events (approximately 3-5 minutes). The agent's value was reset at the start of each episode and evaluated at the end when positions were closed. The training phase allowed multiple passes over the data, while the testing phase was restricted to a single pass. Dueling DQN (D-DQN) was used for discrete actions, while Proximal Policy Optimization (PPO) was applied for continuous actions.

Agent	ND-PnL (*10^5)	PnLMAP(*10^-4)	PR	Sharpe Ratio
C-PPO	9.3 +/- 0.7	117 +/- 3.8	5.0	12.3 +/- 0.8
D-DQN	7.0 +/- 1.7	8.6 +/- 2.2	3.5	1.3 +/- 0.3
Inv-RL	0.3 +/- 0.1	24.7 +/- 4.2	4.3	1.3 +/- 0.3
LOB-RL	1.1 +/- 0.5	1.3 +/- 0.5	2.8	0.2 +/- 0.3
AS	4.7 +/- 0.3	3.9 +/- 0.9	3.3	0.5 +/- 0.2
Random	0.4 +/- 0.1	0.8 +/- 0.3	1.0	-0.2 +/- 0.1

D. Overall Results

Performance Summary: The table below presents the performance of different agents on metrics including ND-PnL, PnLMAP, Profit Ratio (PR), and Sharpe ratio for three different stocks.

E. Extended Experiments

1) Latency

Latency experiments were conducted to evaluate the resilience of strategies under delayed execution conditions. The results showed that **C-PPO** maintained robust performance up to moderate latency, while other models like **AS** and **Random** strategies suffered significantly as latency increased.

2) Ablation Study

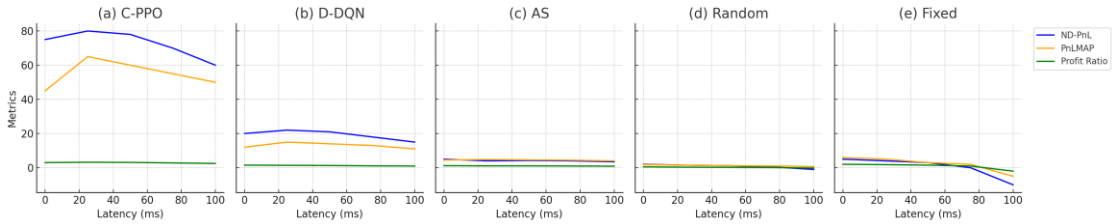
Ablation tests revealed that removing the LOB state or the attention layer drastically reduced performance, underscoring the importance of each component in the overall model architecture.

3) Explanation

Visualizing attention weights in Figure 3 highlighted the focus of the model on recent market changes, while during volatile periods, earlier patterns were also emphasized. This analysis illustrated the model's capability to adapt its attention based on market stability.

For this project, The Attn-LOB model combined with C-PPO performed best in most scenarios, showcasing its robustness to latency and effectiveness in leveraging LOB data for decision-making. This approach demonstrated significant advantages over traditional RL and market-making strategies.

Fig. 2. Latency experiments.



This figure shows how the performance metrics of different agents change as latency increases from 0 to 100 milliseconds. The plots illustrate the robustness and adaptability of each strategy under delayed execution conditions:

1. **(a) C-PPO**:
    a. ND-PnL and PnLMAP start high and maintain relatively good performance under moderate latency but show a gradual decline as latency reaches 100 ms.
    b. Profit Ratio remains stable with minimal changes, showcasing the robustness of C-PPO under latency.
2. **(b) D-DQN**:
    a. ND-PnL and PnLMAP show a steady decrease as latency increases, indicating moderate sensitivity to latency.
    b. Profit Ratio remains relatively flat but also declines slightly with increased latency.
3. **(c) AS Strategy**:
    a. The ND-PnL and PnLMAP metrics show a gradual decline with increased latency.
    b. Profit Ratio demonstrates a slight downward trend, highlighting limited adaptability under latency.
4. **(d) Random Strategy**:
    a. All performance metrics show minimal change initially but decline sharply after 25 ms, indicating poor resilience to latency.
5. **(e) Fixed Strategy**:
    a. ND-PnL and PnLMAP show significant drops beyond 25 ms, becoming negative as latency increases, which indicates a strong impact of latency on this strategy.
    b. Profit Ratio follows a similar trend, emphasizing the strategy's limitations in adapting to delays.

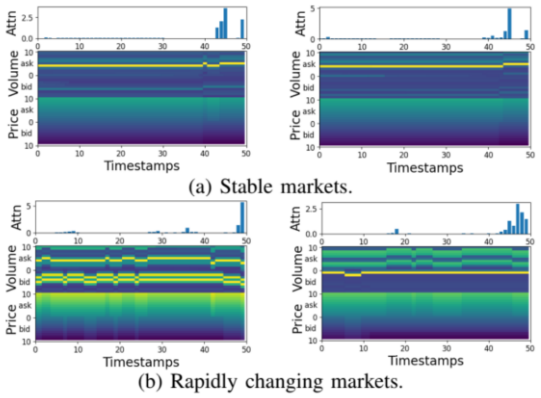


Fig. 3. Attention visualization.

Fig. 2: Attention Visualization

- **(a) Stable Market**:
  - The heatmap represents attention weights over timestamps for stable market conditions. It indicates that the agent focuses on recent price changes while paying some attention to historical data.
- **(b) Rapidly Changing Market**:
  - In volatile conditions, the attention map shows more varied distribution, reflecting the agent's need to consider a broader range of past data to make decisions.

Conclusion

The latency experiments demonstrate that **C-PPO** maintains robust performance compared to other strategies, showcasing adaptability to real-time conditions. The **attention visualizations** highlight the importance of temporal data in decision-making, especially during market volatility. Overall, the results emphasize the benefits of advanced RL models like **Attn-LOB with C-PPO**, which can effectively handle latency and market changes, offering significant advantages over simpler strategies.

REFERENCES

1. H. Demsetz, “The cost of transacting,” *The Quarterly Journal of Economics*, vol. 82, no. 1, pp. 33–53, 1968.

2. M. B. Garman, “Market microstructure,” *Journal of Financial Economics*, vol. 3, no. 3, pp. 257–275, 1976.

3. H. R. Stoll, “The pricing of security dealer services: An empirical study of NASDAQ stocks,” *The Journal of Finance*, vol. 33, no. 4, pp. 1153–1172, 1978.

4. T. Ho and H. R. Stoll, “Optimal dealer pricing under transactions and return uncertainty,” *Journal of Financial Economics*, vol. 9, no. 1, pp. 47–73, 1981.

5. M. Avellaneda and S. Stoikov, “High-frequency trading in a limit order book,” *Quantitative Finance*, vol. 8, no. 3, pp. 217–224, 2008.

6. T. Ho and H. R. Stoll, “On dealer markets under competition,” *The Journal of Finance*, vol. 35, no. 2, pp. 259–267, 1980.

7. F. Guilbaud and H. Pham, “Optimal high-frequency trading with limit and market orders,” *Quantitative Finance*, vol. 13, no. 1, pp. 79–94, 2013.

8. N. T. Chan and C. Shelton, “An electronic market-maker,” 2001.

9. A. Chakraborti, I. M. Toke, M. Patriarca, and F. Abergel, “Econophysics review: I. empirical facts,” *Quantitative Finance*, vol. 11, no. 7, pp. 991–1012, 2011.

10. A. S. Kyle, “Continuous auctions and insider trading,” *Econometrica: Journal of the Econometric Society*, pp. 1315–1335, 1985.

11. L. R. Glosten, “Is the electronic open limit order book inevitable?” *The Journal of Finance*, vol. 49, no. 4, pp. 1127–1161, 1994

12. M. O’Hara, *Market Microstructure Theory*, John Wiley & Sons, 1998.

13. A. Tsantekidis, N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, “Forecasting stock prices from the limit order book using convolutional neural networks,” in *2017 IEEE 19th Conference on Business Informatics (CBI)*, IEEE, 2017, pp. 7–12.

14. A. Tsantekidis et al., “Using deep learning to detect price change indications in financial markets,” in *2017 25th European Signal Processing Conference (EUSIPCO)*, IEEE, 2017, pp. 2511–2515.

15. D. T. Tran et al., “Temporal attention-augmented bilinear network for financial time-series data analysis,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 5, pp. 1407–1418, 2018.

16. M. Shabani et al., “Multi-head temporal attention-augmented bilinear network for financial time series prediction,” *arXiv preprint arXiv:2201.05459*, 2022.

17. J. Moody and M. Saffell, “Reinforcement learning for trading,” *Advances in Neural Information Processing Systems*, vol. 11, 1998.

18. J. Moody and M. Saffell, “Learning to trade via direct reinforcement,” *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 875–889, 2001.

19. Y. Deng et al., “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.

20. Y. Nevmyvaka, Y. Feng, and M. Kearns, “Reinforcement learning for optimized trade execution,” in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 673–680.

21. H. Wei et al., “Model-based reinforcement learning for predictions and control for limit order books,” *arXiv preprint arXiv:1910.03743*, 2019.

22. A. Briola et al., “Deep reinforcement learning for active high frequency trading,” *arXiv preprint arXiv:2101.07107*, 2021.

23. J. Abernethy and S. Kale, “Adaptive market making via online learning,” *Advances in Neural Information Processing Systems*, vol. 26, 2013.

24. Y.-S. Lim and D. Gorse, “Reinforcement learning for high-frequency market making,” in *ESANN 2018-Proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, ESANN, 2018, pp. 521–526.

25. T. Spooner et al., “Market making via reinforcement learning,” *arXiv preprint arXiv:1804.04216*, 2018.

26. Y. Zhong et al., “Data-driven market-making via model-free learning,” in *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, 2020, pp. 4461–4468.

27. J. Sadighian, “Deep reinforcement learning in cryptocurrency market making,” *arXiv preprint arXiv:1911.08647*, 2019

28. B. Gasperov and Z. Kostanjcar, “Market making with signals through deep reinforcement learning,” *IEEE Access*, vol. 9, pp. 61611–61622, 2021.

29. P. Kumar, “Deep reinforcement learning for market making,” in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 2020, pp. 1892–1894.

30. Z. Xu et al., “Performance of deep reinforcement learning for high frequency market making on actual tick data,” in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022, pp. 1765–1767.