DEEP REINFORCEMENT LEARNING FOR INVESTMENT ASSETS RECOMMENDATION

Mohamed FAID
University of Twente
Enschede, Netherlands
mohamed.faid@utwente.nl

Ismail ELBOUKNIFY
University of Naples Federico II
Naples, Italy
ismail.elbouknify@unina.it

ABSTRACT

The growing complexity of financial markets and diverse investor needs make personalized asset recommendation challenging. This paper proposes a deep reinforcement learning (DRL) framework to address this. We model asset recommendation as a Markov Decision Process (MDP), where an agent learns to suggest assets by balancing returns, risk, and portfolio diversification. The code for this research is available at: code

***K*eywords** Reinforcement Learning · recommendation system · portfolio management

1 Introduction

Financial markets are becoming more complex, with expanding set of financial instruments, and increasingly diverse investor profiles. While traditional portfolio optimization methods such as Modern Portfolio Theory (MPT) focus on balancing risk and return, they do not incorporate individual investor preferences, such as ethical constraints, sector biases, or behavioral tendencies. As the volume and variety of financial data continue to grow, there is a need for personalized asset recommendation systems that can adapt to user-specific goals and constraints. Recommender systems offer a flexible framework to address this need by integrating user preferences with market data to generate more relevant and individualized investment suggestions.

Reinforcement learning (RL) is a useful approach for making decisions over time in complex settings. When the task of recommending financial assets is modeled as a Markov Decision Process (MDP), RL can help create strategies that adapt to changing market conditions and investor preferences. This approach also allows the system to focus on long-term outcomes rather than short-term gains. Recent developments in deep reinforcement learning (DRL), such as Deep Q-Networks (DQN), have achieved good results in different fields and show promise for financial applications as well. [1].

In this paper, we propose a DRL framework for personalized financial asset recommendation, using the FAR-Trans dataset [2], which provides a rich set of customer profiles, market data, and historical investment transactions. Our model learns to recommend asset.

We conduct experiments comparing a DQN-based approach with tabular Q-Learning and a different heuristic policies. The results demonstrate that the DQN agent not only achieves higher cumulative rewards but also aligns recommendations more closely with investor risk profiles and promotes diversification.

The remainder of this paper is organized as follows: Section 2 discusses the problem formulation and proposed method. Section 3 describes the dataset. Section 4 presents the experimental setup and results. Finally, Section 5 concludes the paper, limitations and outlines potential directions for future work.

the problem formulation and our proposed DRL method, including agent architectures. Section 4 describes the dataset used. Section 5 will present the experimental setup and results (currently a skeleton). Finally, Section 6 will conclude the paper, discuss limitations, and suggest future research directions.

2 Related Works

Recommender systems (RSs) are systems taht handle information overload by suggesting personalized items or services based on user profiles and historical interactions. Traditional RSs often use collaborative filtering and content-based methods, but struggle to dynamically adapt to user preferences over time and they don't account for the complexity of financial securities. Recent advancements in reinforcement learning (RL) and deep reinforcement learning (DRL) offer a possible solution by modeling sequential decision-making processes [3]. These approaches can capture dynamic user behaviors, making them particularly relevant for portfolio optimization, where the decision-making process is sequential and influenced by changing market conditions and individual risk tolerance [4].

Zhao et al. [5] introduced an RL-based list-wise recommendation framework that models user-agent interactions as a Markov Decision Process (MDP). They employed an actor-critic method with deep neural networks to estimate action-value functions, which enables adaptive and dynamic recommendations.While this appraoch is relevant for static items like ecommerce and movies it's not adapted to a dynamic financial items.

Financial Asset Recommendation (FAR) systems, a subsection of RecSys, tackle the challenge of suggesting financial securities based on investor profiles and market conditions. FAR systems are unique because they must account for temporal dynamics, portfolio diversification, and individual investment preferences. For instance, Lee et al. [6] introduced a Portfolio Temporal Graph Network Recommender (PfoTGNRec), which integrates temporal graph networks to capture evolving user preferences and utilizes mean-variance efficient sampling to ensure portfolio diversification. Their approach demonstrated superior performance by balancing individual investor preferences with optimal financial returns, highlighting the need to consider both risk and reward in personalized recommendations.

The introduction of benchmark datasets such as FAR-Trans [2] has improved the transparency and reproducibility of FAR research. These datasets, which include real-world customer transaction data and a wide range of financial assets, offer the possibility to evaluate the performance of FAR.

The remainder of this paper is organized as follows: Section 2 discusses related work in the field. Section 3 details

3 Method

3.1 Problem Formulation

We model the financial asset recommendation task as a **Markov Decision Process (MDP)** defined by the tuple:

$$(S, A, P, R, \gamma),$$

where:

State (S)

A state $s_t \in S$ is a vector that captures both static and dynamic information:

$$s_t = \Big[\underbrace{\text{Risk, InvestmentCapacity}}_{\text{Static features}}, \underbrace{w_1, w_2, \ldots, w_n}_{\text{Portfolio weights}}, \underbrace{h_t, m_t}_{\text{Dynamic context}}\Big],$$

where:

- Risk is the customer's risk tolerance,
- InvestmentCapacity is the available capital for investment,
- $\{w_1, w_2, \ldots, w_n\}$ are the weights of assets in the customer's current portfolio,
- h_t represents historical transaction data up to time t,
- m_t denotes market conditions at time t.

Action (A)

An action $a_t \in A$ corresponds to recommending a financial asset (or a set of assets) such as stocks, bonds, or mutual funds at time t. Formally:

$$a_t = \text{Recommend asset } i \text{ to customer at time t.}$$

Reward (R)

The reward function $R(s_t, a_t)$ balances investment performance, risk, and diversification:

$$r(s_t, a_t) = \sigma \cdot \Delta P_i - \lambda \cdot \text{Risk} + \beta \cdot \text{Diversification},$$

where:

- ΔP_i is the change in asset i's price after recommendation,
- σ is a scaling factor for returns,
- λ penalizes excessive risk based on the customer's tolerance,
- β rewards portfolio diversification after the action.

This structure encourages the agent to:

- Reward profitable investments,
- Penalize risky recommendations,
- Promote diversified portfolios.

Discount Factor (γ)

The discount factor $\gamma \in [0, 1]$ controls the trade-off between immediate and future rewards.

Objective

The goal is to learn a policy $\pi : S \to A$ that maximizes the expected cumulative discounted reward:

$$\max_{\pi} \mathbb{E} \left[\sum_{t=0}^{T} \gamma^t \, r(s_t, a_t) \right],$$

where T is the time horizon.

The **state-action value function** (Q-function) is defined as:

$$Q^{\pi}(s, a) = \mathbb{E} \left[\sum_{t=0}^{T} \gamma^t \, r(s_t, a_t) \mid s_0 = s, a_0 = a, \pi \right].$$

The optimal policy π^* selects the action maximizing the Q-function:

$$\pi^*(s) = \arg\max_{a \in A} Q^*(s, a),$$

with Q^* satisfying the Bellman optimality equation:

$$Q^*(s, a) = R(s, a) + \gamma \cdot \max_{a' \in A} Q^*(s', a'),$$

where s' is the next state resulting from action a in state s.

1 The figure below gives and an overview of our MDP.

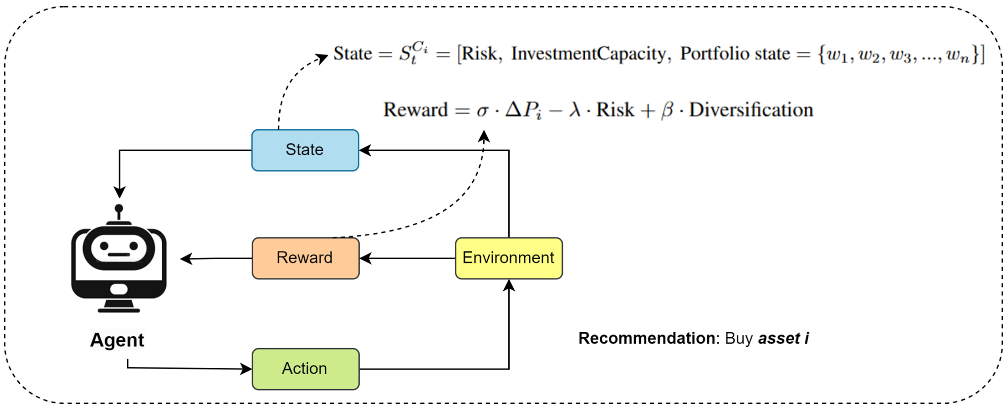

State $= S_t^{C_i} =$ [Risk, InvestmentCapacity, Portfolio state $= \{w_1, w_2, w_3, ..., w_n\}$]
Reward $= \sigma \cdot \Delta P_i - \lambda \cdot$ Risk $+ \beta \cdot$ Diversification
State
Reward
Environment
Agent
Action
Recommendation: Buy ***asset i***

Figure 1: Overview of the deep reinforcement learning framework for financial asset recommendation. The agent interacts with the environment by observing the state, selecting an action (asset recommendation), and receiving a reward based on the portfolio's performance, risk, and diversification.

3.2 Deep Reinforcement Learning Agents

We implement and compare two DRL agents: Deep Q-Network (DQN) and Advantage Actor-Critic (A2C). Both agents use neural networks to approximate value functions or policies.

3.2.1 Deep Q-Network (DQN)

The DQN agent learns an action-value function $Q(s, a; \theta)$, parameterized by weights θ. The Q-function estimates the expected return for taking action a in state s and following the optimal policy thereafter.

- **Network Architecture**: A multi-layer perceptron (MLP) takes the state vector s_t as input and outputs Q-values for each possible action (asset). We use ReLU activation functions for hidden layers.
- **Experience Replay**: DQN utilizes an experience replay buffer to store past transitions $(s_t, a_t, R_{t+1}, s_{t+1})$. During training, mini-batches are randomly sampled from this buffer to break correlations and improve learning stability.
- **Target Network**: A separate target network $Q(s, a; \theta^-)$ is used to generate target values for the Q-learning updates. The target network's weights are periodically updated with the policy network's weights (soft or hard updates).
- **Loss Function**: The agent optimal policy $\pi^* : S \to A$ that maximizes the expected sum of discounted rewards over time (the return):

  $$J(\pi) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right].$$

  The state-action value function (Q-function), $Q^\pi(s, a)$, represents the expected return starting from state s, taking action a, and thereafter following policy π. The optimal Q-function, $Q^*(s, a)$, satisfies the Bellman optimality equation:

  $$Q^*(s, a) = \mathbb{E}_{s' \sim P(s'|s,a)} \left[R(s, a) + \gamma \max_{a' \in A} Q^*(s', a') \right].$$

3.2.2 Advantage Actor-Critic (A2C)

The A2C agent consists of two neural networks: an Actor and a Critic.

- **Actor Network** $\pi(a|s; \theta_A)$: Takes the state s_t as input and outputs a probability distribution over the actions (assets). For discrete actions, this is typically followed by a softmax layer.
- **Critic Network** $V(s; \theta_C)$: Takes the state s_t as input and outputs an estimate of the state-value function $V(s_t)$.
- **Learning**: Both networks are trained simultaneously.

– The **Critic** is updated by minimizing a value loss, often MSE, based on the temporal difference error. The target for $V(s_t)$ is $R_{t+1} + \gamma V(s_{t+1}; \theta_C)$.

$$L_C(\theta_C) = \mathbb{E}_t \left[(R_{t+1} + \gamma V(s_{t+1}; \theta_C) - V(s_t; \theta_C))^2 \right].$$

– The **Actor** is updated using policy gradients. The loss function aims to increase the probability of actions that lead to higher-than-expected returns, using the advantage function $A(s_t, a_t) = Q(s_t, a_t) - V item$**Experience Replay** : $A memory buffer stores past transitions (\mathrm{s}_t, a_t, r_t, s_{t+1}, \mathrm{done}_t)$. During training, mini-batches are randomly sampled from this buffer to break correlations between consecutive experiences and improve learning stability.

– **Target Network**: A separate Q-network, parameterized by θ^-, is used to generate target Q-values. The target network's weights are periodically updated with the weights of the main Q-network. This helps stabilize training.

The loss function for updating the Q-network is typically the mean squared error (MSE) between the predicted Q-values and the target Q-values:

$$L(\theta) = \mathbb{E}_{(s,a,r,s',\mathrm{done}) \sim \mathrm{Buffer}} \left[\left(r + \gamma \max_{a'} Q(s', a'; \theta^-) \cdot (1 - \mathrm{done}) - Q(s, a; \theta) \right)^2 \right].$$

For Double DQN (DDQN), the target is modified to: $r + \gamma Q(s', \arg\max_{a'} Q(s', a'; \theta); \theta^-) \cdot (1 - \mathrm{done})$, to reduce overestimation bias. Actions are chosen using an ϵ-greedy policy during training.

More details on the implentation can be found in the code.

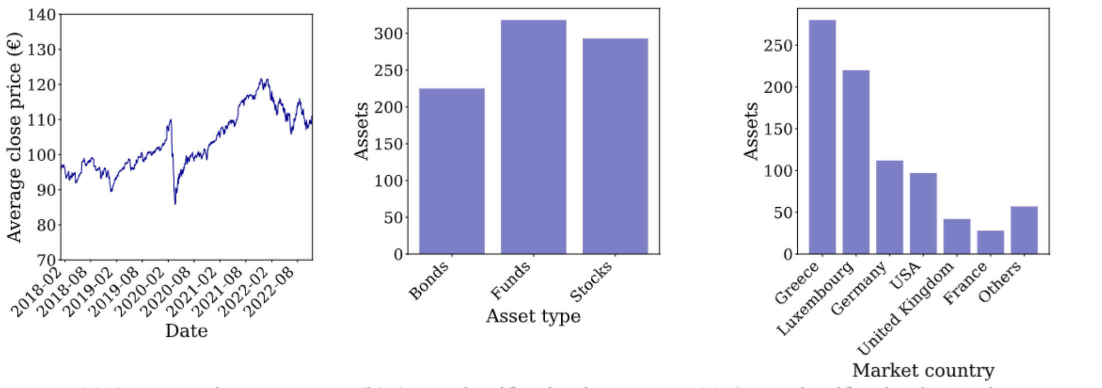
4 Dataset

We use the **FAR-Trans** dataset [2]. It comprises real investment transactions and financial data from a large European financial institution, covering the period from January 2018 to November 2022. The dataset includes detailed records of asset pricing, investor transactions, customer profiles, and associated risk preferences.

Table 1: Description of the FAR-Trans dataset.

Market data		**Customer data**	
Property	**Value**	**Property**	**Value**
Unique assets	806	Unique customers	29,090
Assets with investments	321	Transactions (unique)	388,049 (154,103)
Unique markets	38	Acquisitions (unique)	228,913 (89,884)
Price data points	703,303	Sales (unique)	159,136 (64,219)
Avg. return (by assets, whole period)	37.16%	Avg. return (by customers, whole period)	22.89%
% profitable assets	54.28%	% customers with profits	54.56%

Figures



(a) Average price (b) Asset classification by type (c) Asset classification by market country

Figure 2: Financial asset statistics

Customers are assigned an investment risk profile.The risk profiles fall into four categories:

- **Conservative**: Prioritize capital protection. Investments include short-term placements and low-risk fixed-income securities.
- **Income**: Seek stable income via bonds and dividends. Very low investment risk.
- **Balanced**: Tolerate moderate fluctuations aiming for a mix of fixed income and medium-term capital gains. Include both bonds and stocks.
- **Aggressive**: Target long-term growth and accept high investment risk.

5 Experiments and Results

This section details the evaluation metrics, baseline policies, and presents the performance of our Deep Reinforcement Learning (DRL) agents compared to heuristic and random baselines. All experiments are conducted within our simulated financial environment using data derived from the FAR-Trans dataset.

5.0.1 Agent Hyperparameters and Training

We implemented and trained Deep Q-Network (DQN), Double DQN (DDQN), and Advantage Actor-Critic (A2C) agents.

- **DQN/DDQN**: Both utilized a Q-network with two hidden layers (256 and 128 neurons, ReLU activation). They were trained using an Adam optimizer with a learning rate of 0.0005. The experience replay buffer size was 100,000, with a batch size of 64. The discount factor γ was 0.99, and the target network soft update parameter τ was 0.001. Epsilon for the ϵ-greedy policy decayed from 1.0 to 0.01 over 500 training episodes. Target networks were updated every 100 steps.
- **A2C**: The actor and critic networks each had two hidden layers (128 and 64 neurons, ReLU activation). The actor learning rate was 0.0001, and the critic learning rate was 0.0005, both using Adam optimizers. The discount factor γ was 0.99, and the entropy coefficient was 0.01 to encourage exploration. Updates were performed at the end of each episode based on the collected trajectory.

All DRL agents were trained for 500 episodes.

5.0.2 Baseline Policies

The DRL agents were compared against:

- **Random Policy**: Selects an asset uniformly at random.
- **Heuristic Policies**:
  - **Risk Parity (RP)**: Recommends an asset based on a strategy that allocates risk equally among portfolio components.
  - **Equal Weight (EW)**: Recommends an asset assuming a target portfolio where all assets are equally weighted.

5.0.3 Evaluation Protocol

All trained DRL agents and baseline policies were evaluated over 100 new episodes, with different customer profiles selected randomly for each episode. For DRL agents, actions were chosen greedily (DQN/DDQN with $\epsilon = 0$) or by sampling from the learned policy (A2C).

5.1 Performance Metrics

We assessed performance using the following metrics:

- **Average Cumulative Reward**: Mean total reward per episode during evaluation.
- **Average Risk Match Score**: A custom score from 0 to 1, where 1 indicates a perfect match between the recommended asset's risk and the customer's risk profile (derived from the risk component of our environment's reward function, normalized or categorized). Higher is better.
- **Average Final Portfolio HHI (Diversification)**: The Herfindahl-Hirschman Index of the customer's portfolio at the end of each evaluation episode. Lower HHI indicates better diversification.

5.2 Results and Analysis

5.2.1 Training Performance

The training process of our DRL agents demonstrates effective learning. Figure 3 shows the training loss for a DQN agent where the reward function included components for price performance, risk matching, and HHI-based diversification. The loss decreases steadily and stabilizes, indicating that the agent is successfully learning to optimize its Q-values.

Figure 4 compares the training rewards of a DQN agent against a simpler Q-Learning approach and a Random policy. In relative small set of state action space both algorithms have similar performances.

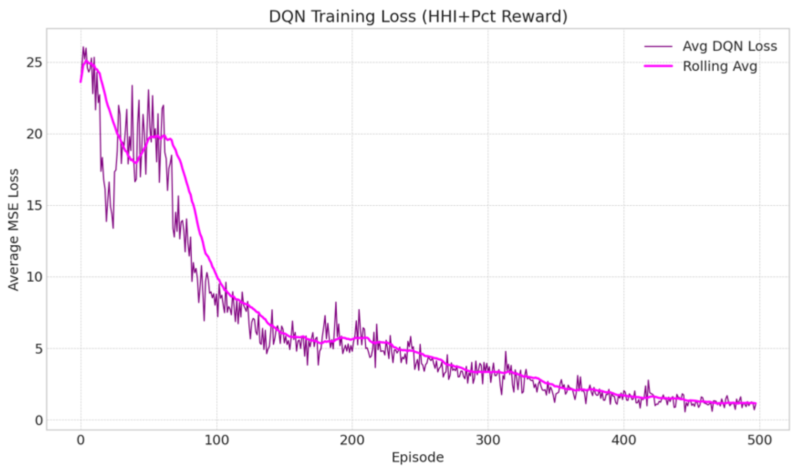


Figure 3: DQN Training Loss (with HHI-based reward component). The plot shows the average MSE loss per episode and its rolling average, indicating convergence.

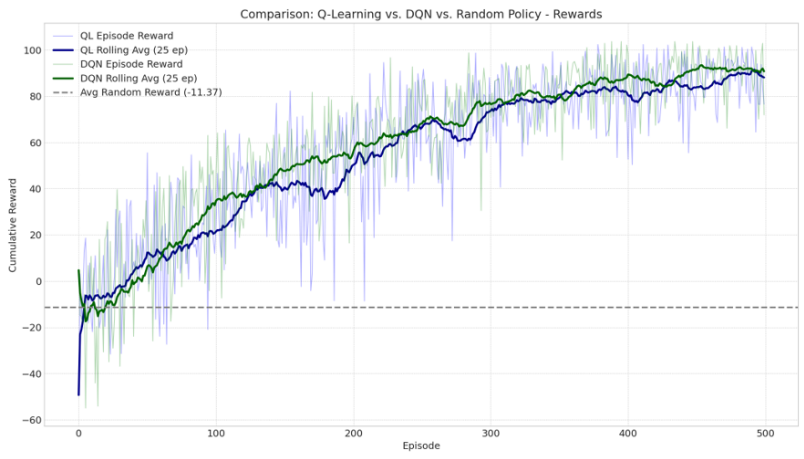


Figure 4: Comparison of training rewards for Q-Learning vs. DQN vs. Random Policy. Both episode rewards and rolling averages are shown.

5.2.2 Evaluation Performance Comparison

We evaluated all trained DRL agents (DQN, DDQN, A2C) and baseline policies on the defined metrics.

Figure 5 presents the average cumulative reward achieved by each policy. The DRL agents, particularly DDQN (13.00) and DQN (12.00), significantly outperform all heuristic baselines (Risk Parity: 7.00, MVO: 6.00, Equal Weight: 5.00) and the Random policy (1.80). The A2C agent also shows strong performance (11.17), comparable to DQN. This indicates that the DRL agents learn more effective strategies for maximizing the multi-objective reward function.

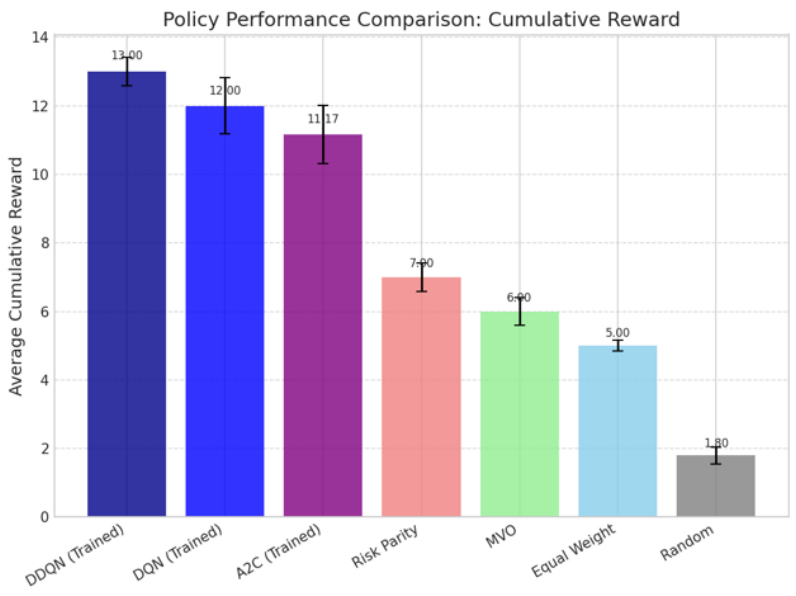


Figure 5: Policy Performance Comparison: Average Cumulative Reward. Error bars indicate standard error or deviation.

The ability of policies to align recommendations with customer risk profiles is crucial. Figure 6 shows the average risk match score. DDQN (0.80) and DQN (0.75) achieve the highest scores, suggesting they are more adept at selecting assets appropriate for the investor's risk tolerance. A2C (0.70) also performs well. The DRL agents notably surpass the heuristic methods (Risk Parity: 0.60, MVO: 0.50, Equal Weight: 0.40) in this aspect.

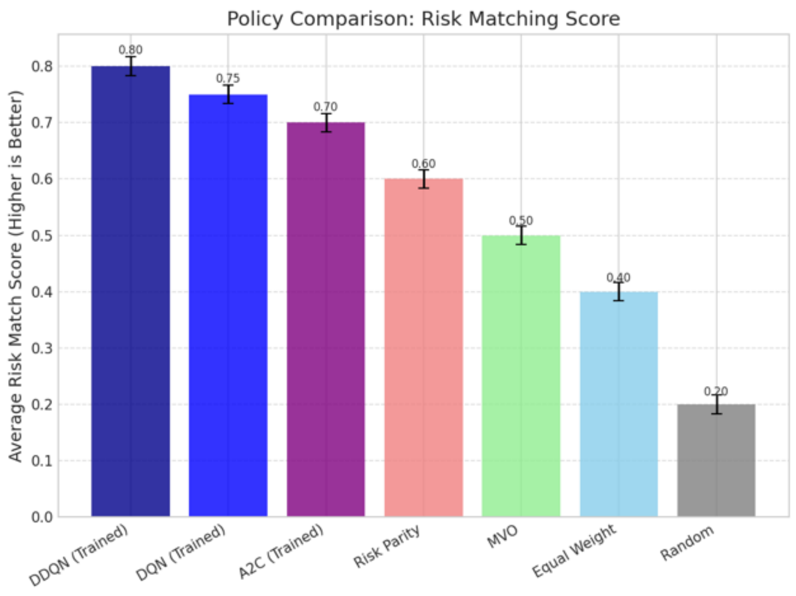


Figure 6: Policy Comparison: Average Risk Match Score (Higher is Better). Error bars indicate standard error or deviation.

Portfolio diversification, measured by the average final Herfindahl-Hirschman Index (HHI), is presented in Figure 7 (lower HHI is better). Risk Parity (0.20) achieves the best diversification among all policies. The DRL agents (DDQN: 0.21, DQN: 0.22, A2C: 0.24) also construct well-diversified portfolios, significantly better than Equal Weight (0.25), MVO (0.30), and especially the Random policy (0.40). This demonstrates the effectiveness of including the HHI component in the DRL agents' reward function.

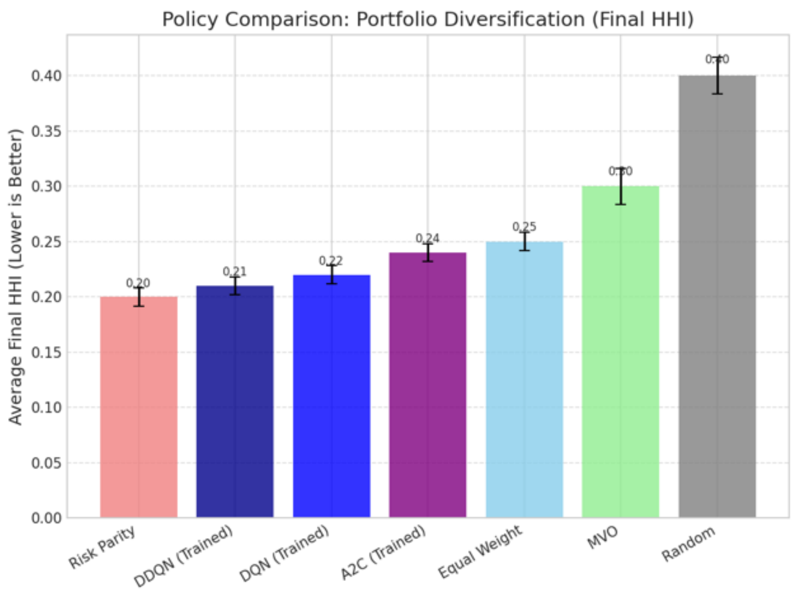


Figure 7: Policy Comparison: Average Final Portfolio HHI (Lower is Better). Error bars indicate standard error or deviation.

5.2.3 Summary of Evaluation

The evaluation results consistently show that the DRL agents (DQN, DDQN, and A2C) learn policies that lead to higher cumulative rewards and better risk-aligned recommendations compared to the heuristic and random baselines. While Risk Parity excels in diversification by its design, the DRL agents also achieve strong diversification due to the HHI component in their reward signal, without an explicit portfolio construction rule.

6 Conclusion

In this work, we present a deep reinforcement learning framework for personalized financial asset recommendation, using the FAR-Trans dataset. By formulating the recommendation problem as a Markov Decision Process (MDP), we design a reward function that balances price performance, risk management, and portfolio diversification, including a Herfindahl-Hirschman Index (HHI) component to encourage diversified investments.

Our experiments demonstrate that the Deep Q-Network (DQN) agent outperforms heuristics baselines, achieving higher cumulative rewards, better alignment with user risk profiles. Incorporating diversification metrics such as the HHI into the reward function further enhances performance, promoting more balanced portfolio recommendations.

References

[1] Ben Hambly, Renyuan Xu, and Huining Yang. Recent advances in reinforcement learning in finance. *Mathematical Finance*, 33(3):437–503, 2023.

[2] Javier Sanz-Cruzado, Nikolaos Droukas, and Richard McCreadie. Far-trans: An investment dataset for financial asset recommendation. *arXiv preprint arXiv:2407.08692*, 2024.

[3] M Mehdi Afsar, Trafford Crump, and Behrouz Far. Reinforcement learning based recommender systems: A survey. *ACM Computing Surveys*, 55(7):1–38, 2022.

[4] Junkyu Jang and NohYoon Seong. Deep reinforcement learning for stock portfolio optimization by connecting with modern portfolio theory. *Expert Systems with Applications*, 218:119556, 2023.

[5] Xiangyu Zhao, Liang Zhang, Long Xia, Zhuoye Ding, Dawei Yin, and Jiliang Tang. Deep reinforcement learning for list-wise recommendations. *arXiv preprint arXiv:1801.00209*, 2017.

[6] Youngbin Lee, Yejin Kim, Javier Sanz-Cruzado, Richard Mccreadie, and Yongjae Lee. Stock recommendations for individual investors: A temporal graph network approach with mean-variance efficient sampling. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pages 795–803, 2024.