Optimizing Data Filtering in Multi-Armed Bandit Algorithms for Reinforcement Learning

Shengshi Zhang*

Math Department, Stony Brook University, NY11790, New York, United States

Abstract. This study investigates the performance of data filtering algorithms in multi-armed bandit (MAB) problems for reinforcement learning applications. It focuses on five algorithms: Epsilon-Greedy (ε-Greedy), Upper Confidence Bound (UCB), Linear Upper Confidence Bound (LinUCB), Thompson Sampling, and Linear Thompson Sampling (LinTS). The algorithms were evaluated in static and dynamic environments using the MovieLens dataset and transferred to binary rewards to measure performance. Each algorithm was tested in simulations with 1,000 interactions, and compared with cumulative reward, accuracy, and adaptability. The experiments involved multiple arms, each with a unique reward distribution, and simulated static (fixed rewards) and dynamic (periodically changing rewards) environments. Result shows that LinUCB and LinTS achieved the highest cumulative rewards in static settings. In dynamic environments, Thompson Sampling demonstrated superior adaptability by adjusting quickly to changing reward structures. Overall, LinUCB and LinTS maintained an accuracy rate of over 36%. This research provides critical insights into the trade-offs between exploration and exploitation, offering theoretical and experimental support for optimizing MAB algorithms in real-world applications such as recommendation systems.

1 Introduction

Bandit problems are a type of optimization problem used to make decisions in uncertain situations. The algorithms for data screening and cleaning need to be improved to obtain more accurate results. Optimization problems do not have to be new; the best solution is the one that meets the screening criteria and works well in the appropriate context. This paper discusses several MAB algorithms: ε-Greedy, UCB, LinUCB, Thompson Sampling, and LinTS. These algorithms deal with the trade-offs between exploration and exploitation in different ways and adapt to changing reward distributions in various manners. Comparing how well they work in different scenarios can identify the contexts in which each algorithm excels and understand their advantages and disadvantages.

These elements slow down the algorithm and waste valuable computational resources. This paper uses data filtering to make the algorithm focus on valid information related to rewards, thereby reducing unnecessary exploration. After analyzing the adaptability of

* Corresponding author: Shengshi.zhang@stonybrook.edu

different algorithms, using different algorithms allows the improvement of decision-making efficiency to directly and then translates into higher cumulative rewards.

Many studies and experiments have shown that Thompson Sampling outperforms UCB and ε-Greedy [1]. Thompson Sampling is based on Bayesian reasoning, constantly updating the reward distribution of each arm to effectively handle uncertainty and ultimately use existing knowledge to make the best decision. However, its limitation is that it relies on prior assumptions, such as the Beta distribution [2]. In solving the contextual multi-armed bandit problem, Thompson Sampling combined with a linear reward model formed the LinTS algorithm. In contrast, LinUCB controls the degree of exploration by adjusting the confidence interval and does not use Bayesian updating [3]. The findings of this paper show that each algorithm has its place to be used. Compared with basic algorithms such as UCB and Thompson Sampling, their optimized algorithms such as LinUCB and LinTS have greatly changed their applicability.

2 Related work

2.1 Algorithm explanations

The MAB problem is a decision-making challenge in which a player chooses between multiple options (arms), each with an unknown reward probability. The goal is to maximize total rewards over time by balancing exploration (trying different arms to learn their rewards) and exploitation (choosing the best-known arm based on current information). If reinforcement learning is defined as a machine learning method that requires continuous updating after interaction between the agent and the environment, then the MAB problem is integrated into identifying the optimal behavioral strategy [4, 5]. The differences between MAB and Reinforcement Learning are highlighted in Table 1.

Table 1. Compare multi-armed bandit and reinforcement learning.

	Multi-Armed Bandit	**Reinforcement Learning**
State dependence	independent decision in each action selection	each action will change the environment and will affect the future decisions
Rewards	immediate (once)	delayed (long term gains)
Objectives	pull as many arms as possible to get the highest reward	focus on achieving long term goals and finding a policy or policy optimization
Complexity	isolating to a finite set of independent arms	changing over time, the decision space is complex

Although these differences are easily understood, the balance between exploration and exploitation remains a common challenge in both. The MAB problem can be considered a simplified version of reinforcement learning, and the algorithms developed for MAB can also be integrated into reinforcement learning frameworks.

2.2 Algorithm details

The ε-Greedy and UCB algorithms are more suited to static environments where the reward probability remains constant over time, while Thompson Sampling and LinUCB are better suited for dynamic environments due to their adaptive nature, especially when reward distributions change or contextual information can assist decision-making [6]. This paper will analyze the accuracy of different algorithms by dynamically adjusting them to assess their ability to prioritize datasets, which are evaluated based on real-time feedback, thereby continuously optimizing data selection and processing efficiency.

ε-Greedy algorithm maintains a fixed epsilon value for the entire decision-making process. Its limitation is that this fixed epsilon does not adapt from period to period, which may result in missed information and less efficient exploration. However, an Adaptive ε-Greedy Method has been proposed in, and although not discussed here, it demonstrates ongoing efforts to optimize exploration and exploitation balance in reinforcement learning [7].

UCB and LinUCB are foundational strategies for addressing the exploration-exploitation trade-off in decision tree analysis and optimization problems. The UCB algorithm explores the relationship between new actions to obtain more information and actions that have historically performed better by constructing estimated rewards and corresponding uncertainty bounds for each action [8]. At each step, UCB selects the action with the highest upper confidence bound, based on the known reward and uncertainty of that action. While UCB works well in many scenarios, since it has drawbacks, particularly in scenarios with high exploration costs. Its computational complexity is high, then it may not be able to converge to a better strategy in complex reward structures.

Thompson Sampling has become a popular method for balancing exploration and exploitation, especially in multi-armed bandit settings. Thompson Sampling was introduced by William R. Thompson in 1933, applies Bayesian principles to estimate the likelihood of success for each option. By sampling from the posterior distributions, Thompson Sampling allows the model to prioritize promising arms while maintaining a degree of exploration. This ensures efficient convergence to the optimal arm, even under uncertainty [2].

Recent studies have highlighted the adaptability of Thompson Sampling to real-world problems, including optimization in high-dimensional spaces. Klarich et al. (2024) demonstrated Thompson Sampling's utility in chemistry, where it was used to search through ultralarge synthesis-on-demand databases to identify optimal molecular structures. This application highlights Thompson Sampling's ability to handle large search spaces and make informed, probabilistic selections [9].

LinTS, an extension of Thompson Sampling, incorporates contextual information to refine the decision-making process. By using a linear model to approximate the reward functions for different actions, LinTS makes more efficient use of feature data compared to standard Thompson Sampling [10]. This extension is particularly useful when additional contextual information can guide exploration, making it more efficient for practical applications such as recommendation systems and online advertising.

3 Methodology

To assess the performance of data filtering algorithms in MAB problems. Experiments was conducted using five algorithms: ε-Greedy, UCB, LinUCB, Thompson Sampling and LinTS. The experiments compared these algorithms in terms of cumulative reward, accuracy, exploration-exploitation balance, and adaptability across different conditions. Dynamic and static environments were simulated to evaluate algorithmic performance under varying scenarios.

3.1 Data preprocessing

The experiments took place in a simulated MAB environment with arms, each representing a unique option with a corresponding reward distribution. The reward distributions were generated randomly, and each arm was assigned a dimensional feature vector representing contextual information. In the contextual bandit setting, the context was used to determine reward probabilities. For static environments, the reward distributions remained constant. For dynamic environments, they varied periodically to simulate changing conditions over time.

Further details about the dataset used can be found in the MovieLens 1M dataset documentation, which provides over one million rating records for 3,900 movies rated by 6,040 users.

3.2 Experimental setup

The experiments were implemented in Python to test MAB and contextual bandit environments. Each algorithm was run for a significant number of iterations to ensure convergence and provide reliable results.

Static Environment. The reward distribution remained unchanged throughout the experiment, allowing for an analysis of the long-term performance of each algorithm in a stable context.

Dynamic Environment. The reward distribution changed every 500 iterations to simulate a non-stationary environment. This setup enabled the evaluation of each algorithm's ability to adapt to changes in the reward structure.

This experimental design provided a comprehensive analysis of the effectiveness of each algorithm in solving MAB problems, which identify their strengths and weaknesses in different conditions.

3.3 Cumulative reward

In this section, five MAB algorithms were tested. Each algorithm balances exploration and exploitation to maximize cumulative payoffs. ε-Greedy explores randomly based on a fixed probability, UCB selects under-explored actions using confidence intervals, and LinUCB incorporates contextual information into the selection process. Thompson Sampling uses Bayesian updates to find the best choice through sampling. LinTS improves this by using context to adapt. The algorithms were used to compare the rewards of different strategies for user-movie interaction.

3.3.1 Adaptability and accuracy

The dataset contains 1,000,209 user-movie rating records. Adaptability and accuracy were measured by converting ratings into binary rewards: a rating greater than or equal 4 was considered a reward 1, and ratings below 4 were treated as no reward 0. Performance was evaluated based on cumulative changes in returns over a rolling window, as well as the reward matching ratio.

Adaptability. The data was divided into windows of 100,000 interactions, and the cumulative rate of change in returns was calculated for each window. Algorithms showing larger rates of change demonstrated quicker adaptation to new data, reflecting their ability to adjust to changing user interactions.

Accuracy. Accuracy was evaluated by checking whether the algorithm-selected movies aligned with movies that users rated higher, indicating the algorithm correctly predicted user

preferences. This was measured by examining whether the selected movie received a reward, and the final result reflected the average accuracy across the entire dataset.

3.3.2 Static and dynamic environments

In the static environment, the reward criterion was consistent for all user-movie interactions: a rating greater than or equal to 4 was considered a reward. This setup simulated a scenario where user preferences remained constant over time. In contrast, in the dynamic environment, preference changes were simulated by altering the reward criterion after a set number of interactions [11]. For example, in some phases, a rating greater than or equal to 4 was considered a reward, while in others, a rating greater than or equal to 3 was considered a reward. This setup tested each algorithm's adaptability to changes in user preferences.

4 Experiment and results

Different algorithms exhibit distinct characteristics in terms of cumulative reward in MAB problems. The ε-Greedy algorithm normally experiences slower growth in cumulative reward during the initial stages due to frequent exploration, with increased exploitation as the algorithm gradually converges to an optimal strategy. While it can explore scenarios, it often fails to offer a substantial reward like those of more specialized algorithms. In contrast, the UCB algorithm tends to excel in cumulative reward, especially in static environments. Using upper confidence bounds to balance exploration and exploitation, then UCB effectively identifies high-reward options, making it suitable for scenarios with stable preferences. Figure 1 presents a comparison of cumulative rewards over 500 interactions, while Figure 2 shows the results for 1,000 interactions.

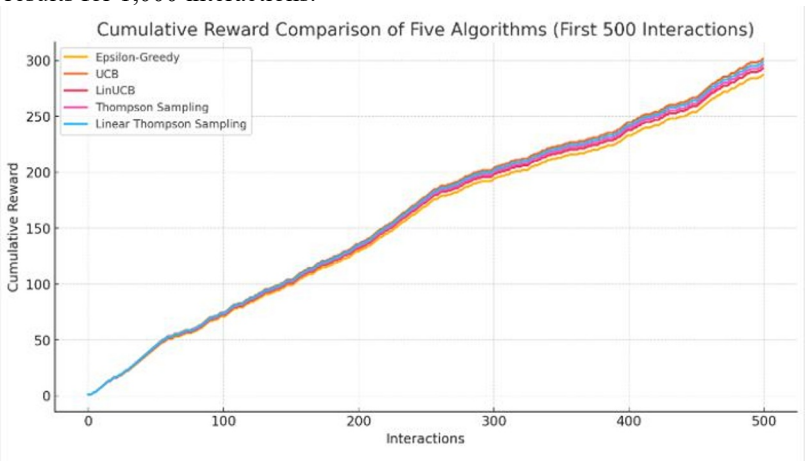

Fig. 1. Cumulative Reward Comparison Over 500 Interactions. (Photo/Picture credit: Original)

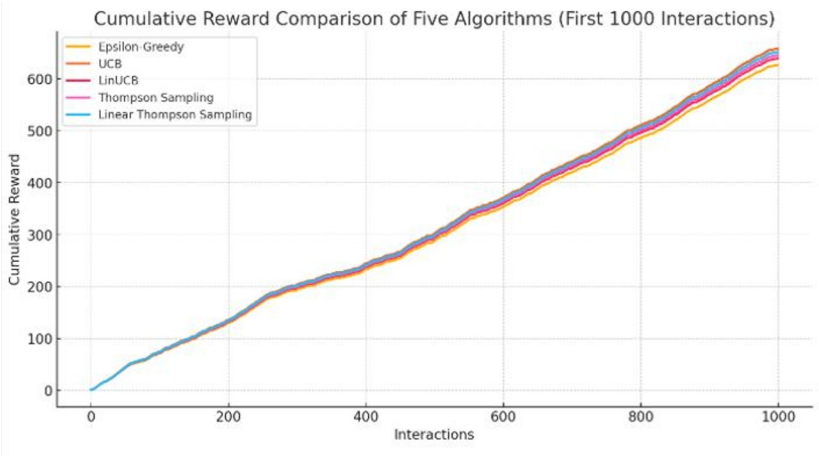

Fig. 2. Cumulative Reward Comparison Over 1000 Interactions. (Photo/Picture credit: Original)

LinUCB incorporates linear context and enhances cumulative reward in scenarios involving user or item features, therefore making it effective for consistent and stable data. On the other hand, Thompson Sampling, due to its random exploration strategy, generally demonstrates moderate cumulative reward during early exploration but converges to an efficient strategy over time. LinTS shows strong performance when contextual information is available. So LinTS utilizes user and item features effectively, also leading to higher cumulative rewards in contexts rich with contextual data. Figure 3 highlights the accuracy comparison across the five algorithms over 1,000 interactions.

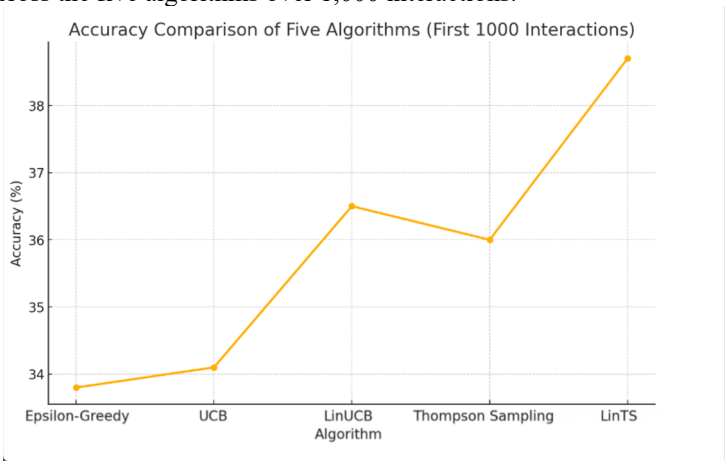

Fig. 3. Accuracy Comparison of Five Algorithms Over 1,000 Interactions. (Photo/Picture credit: Original)

These algorithms proof different behaviors in terms of cumulative reward and accuracy. ε-Greedy's slower cumulative reward growth is attributed to its frequent exploration, which gradually improves as the algorithm converges. Though useful in scenarios demanding exploration, it underperforms in cumulative reward and accuracy compared to more specialized approaches. UCB strikes a strong balance between exploration and exploitation and shows high cumulative reward and accuracy in static environments. LinUCB boosts both cumulative reward and accuracy by leveraging linear context, making it ideal for environments featuring user or item characteristics.

Thompson Sampling shows improvement over time but is characterized by fluctuating accuracy due to its randomness. Similar to the performance of LinUCB, LinTS demonstrates optimal results when contextual data is available. This approach yields higher cumulative rewards with greater accuracy, particularly in dynamic settings. Figure 4 compares the cumulative rewards of the five algorithms in static versus dynamic environments over 1,000 interactions.

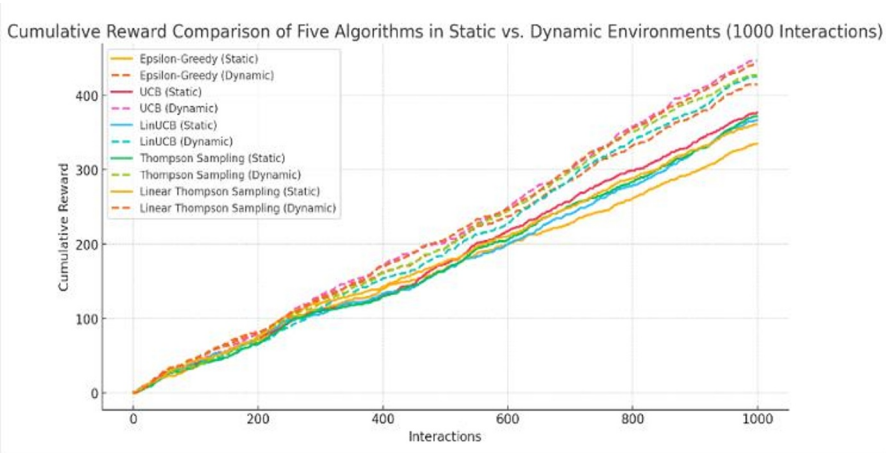

Fig. 4. Cumulative Reward Comparison of Five Algorithms in Static vs. Dynamic Environments. (Photo/Picture credit: Original)

In static and dynamic environments, the algorithms demonstrate distinct characteristics in terms of cumulative reward, accuracy, and adaptability. ε-Greedy, while initially slow in cumulative reward due to frequent exploration, improves over time as it converges into a more optimal strategy. However, it underperforms due to its limited adaptability.

UCB remains highly effective in static environments, balancing exploration and exploitation, but shows lower adaptability in dynamic settings due to its slower response to changes. LinUCB performs nicely in both environments by leveraging linear context, making it suitable for scenarios where features help track shifting preferences.

Thompson Sampling proves its high adaptability, and excels in dynamic environments by capturing preference changes but is less stable in static settings due to its inherent randomness. LinTS adapts good to changes in dynamic environments, maintaining high accuracy and reward while offering stability in static settings.

5 Conclusion

In reinforcement learning, the performance of these five algorithms varies across different tasks and environments. ε-Greedy provides a basic exploration-exploitation balance, suitable for simple tasks but performs poorly in dynamic and complex environments. UCB excels in static environments but lacks the flexibility needed in variable or complex settings. LinUCB is dynamic but relies on the assumption of a linear feature representation and is suitable for context-rich environments such as recommender systems and advertisement placement. Thompson Sampling is suited for tasks involving changing conditions in dynamic environments. LinTS is an optimal solution for real-time recommendation and advertisement placement in high-dimensional like dynamic environments. The recommendation systems of YouTube, Twitch, Amazon, and other similar platforms use deep learning technology to analyze and predict user viewing behavior with neural network models. These models process user data to update recommendations in real time, ensuring users get the latest content.

The recommendation systems in question are optimized constantly and when considered in conjunction with the MAB problem. The recommendation system optimizes the content push through a balance of exploration and exploitation. Reinforcement learning plays a principal role in this process, guiding the system in selecting the arms in a way that optimizes long-term user engagement.

The initial push systems employed UCB to balance exploration and exploitation. As demand increased, this algorithm was extended to LinUCB, which introduced linear context information to optimize recommendations. The Thompson Sampling algorithm helps recommender systems make decisions in uncertain situations. As data grew, Thompson Sampling became LinTS. This added user and content information, making recommendations more flexible and better at adapting to changing user preferences. The algorithmic optimization from UCB to LinUCB and from Thompson Sampling to LinTS reflect the main point of reinforcement learning, namely continuous updating algorithms. These algorithms adjust recommendations in real-time based on user interaction data, such as clicks or watch time, enabling personalized and dynamic content optimization.

References

1. D. Russo, B. Van Roy, Learning to Optimize Via Posterior Sampling. *Math. Oper. Res.* **39**, (2014), 1221–1243, https://doi.org/10.1287/moor.2014.0650.
2. S.K. Thompson, Sampling *2nd ed.*. *Wiley*, 2002.
3. S. Agrawal, N. Goyal, Thompson Sampling for Contextual Bandits With Linear Payoffs. *arXiv*, 2012, https://doi.org/10.48550/arXiv.1209.3352.
4. J.A. Bather, Bandit Problems: Sequential Allocation of Experiments. *J. R. Stat. Soc. A.* **149**, (1986), 271–271, https://doi.org/10.2307/2981558.
5. A.M. Andrew, Reinforcement Learning: An Introduction by Richard S. Sutton and Andrew G. Barto. *Robotica* **17**, (1999), 229–235, https://doi.org/10.1017/S0263574799211174.
6. K. H. Huang, H. T. Lin, Pairwise Regression with Upper Confidence Bound for Contextual Bandit with Multiple Actions, Conference on Technologies and Applications of Artificial Intelligence, (2013), 19-24 https://doi.org/10.1109/TAAI.2013.18
7. A. dos Santos Mignon, R.L. de Azevedo da Rocha, An Adaptive Implementation of ε-Greedy in Reinforcement Learning. *Procedia Comput. Sci.* **109**, (2017), 1146–1151, https://doi.org/10.1016/j.procs.2017.05.431.
8. Y.S. Chen, et al., LinUCB-based Handover Algorithm for Throughput Maximization in Heterogeneous Cellular Networks. *2023 IEEE 20th Consumer Communications & Networking Conference*, 734–739, https://doi.org/10.1109/CCNC51644.2023.10060709.
9. K. Klarich, et al., Thompson Sampling - An Efficient Method for Searching Ultralarge Synthesis on Demand Databases. *J. Chem. Inf. Model.* **64**,(2024), 1158–1171, https://doi.org/10.1021/acs.jcim.3c01790.
10. A. Moradipari, et al., Linear Thompson Sampling Under Unknown Linear Constraints. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing*, (2020), 3392–3396, https://doi.org/10.1109/ICASSP40776.2020.9053865.
11. M. Niimi, T. Ito, Budget-limited Multi-armed Bandit Problem with Dynamic Rewards and Proposed Algorithms. *2015 IIAI 4th International Congress on Advanced Applied Informatics*, (2015), 540–545, https://doi.org/10.1109/IIAI-AAI.2015.248.