Stochastic Multi Armed Bandits: Frequentist and Bayesian Analysis

Shivam Bhat
bhat41@purdue.edu

Shyamvanshikumar Singh
sing1012.edu

Raghav Ram
rram@purdue.edu

Abstract

The Multi-Armed Bandit (MAB) problem, a cornerstone in the study of decision-making under uncertainty, presents a scenario akin to a gambler choosing from multiple slot machines, each with unknown payout rates. This dilemma encapsulates the critical challenge of balancing exploration and exploitation . Rooted in probability and statistics, MAB problems have far-reaching implications in various domains, from optimizing online algorithms to guiding strategic choices in business and healthcare, where decision-makers must continually adapt to evolving environments with incomplete information.This research presents a detailed empirical analysis of Frequentist and Bayesian strategies in addressing the Stochastic Multi-Armed Bandit problem, with a particular focus on the mathematical intricacies involved in decision-making under uncertainty. By employing advanced mathematical models and simulations across diverse scenarios, the study offers profound insights into the performance and applicability of these algorithms. We demonstrate how different strategies fare in terms of reward maximization, best-arm identification, and computational efficiency, offering a comprehensive understanding of the exploration-exploitation trade-off, highlighting the practical applications of these approaches in fields like online advertising, economics and healthcare. The study offers a detailed comparison, guiding future research in optimizing decision-making algorithms for real-world problems. Code and results for experiments can be found at [1]

1 Introduction and Motivation

The study of Multi-Armed Bandits (MAB) holds significant relevance in various domains, including computer science, operations research, statistics, and economics. In essence, this problem entails the allocation of finite resources to a set of actions with the aim of maximizing outcomes. These outcomes are not known in advance and can only be learned over a period of time.

Multi-Armed Bandits serve as a classic example embodying the exploration-exploitation trade-off. In formal terms, a stochastic bandit is defined as having the following structure: at any time t, there are k possible actions, indicated by $\mathcal{A} = \{a_1, ..., a_k\}$, each of which results in a reward $r[a] \sim \mathcal{D}_a$, sampled from a distribution $\mathcal{D}_a$. The primary objective is to optimize the cumulative rewards and/or determine the best rewarding action over time T.

Stochastic Multi-Armed Bandits algorithms strive to estimate the parameters $\{\theta_a^i\}$ of the reward distributions $\mathcal{D}_a$. Similar to other statistical estimation problems, bandit algorithms can be formulated from two main perspectives: Frequentist and Bayesian. The Frequentist approach has historically dominated the analysis of stochastic MAB problems, leading to the development of algorithms such as ϵ-greedy and Upper Confidence Bound (UCB), which have been extensively studied and applied in various contexts[6][2]. In contrast, the Bayesian formulation introduces the concept of prior beliefs regarding the reward distribution, making it increasingly popular in industrial applications[4]. Algorithms like Thompson sampling have shown to achieve optimal bounds in such settings[7][3, 5].

Preprint. Under review.

However, the deployment of Thompson sampling in complex problems is often constrained by its use of samples from posterior distributions, which are often difficult to generate in regimes where the posteriors do not have closed forms. A solution to this challenge has emerged in the form of approximate sampling techniques, which enable the generation of samples from approximated posterior distributions[4].

2 Background

The name, "Multi-armed bandits", comes from an imagined gambling scenario where a gambler is faced with a row of slot machines, also known as one-armed bandits, that appear identical but offer different rewards not known a-priori. With the goal of maximizing the total reward, the gambler has to choose which machines to play, the order of playing them, and whether to try different machines or to continue with the same machine at the next rounds. This crucial dilemma between 'exploration' of new machines and 'exploitation' of the best-known machine, called the exploration-exploitation trade-off, is embodied by multi-armed bandits, and is common in machine learning applications. This gambling scenario is used to imagine and model similar real-life problems as multi-armed bandits. Online advertisements (choosing which ad to display to maximize revenue), medical trials (choosing which drug to prescribe to maximize health outcome), web search (choosing which search results to display to maximize user satisfaction), are examples of multi-armed bandit problems.

Multi-armed bandits is a simple and powerful framework for algorithms that make decisions over time under uncertainty. The basic version of the algorithm is tasked with choosing from K actions (a.k.a. arms) in T rounds, while collecting the reward for the chosen arm at each round. The reward is drawn independently from a fixed distribution specific to each arm, not known to the algorithm a-priori. The objective of the algorithm is to maximize the total reward collected. A good MAB algorithm strives to balance between exploration of new arms and exploitation of the best-known arm, and tries to find the best arms without spending a long time exploring.

Multi-armed bandits can be modelled across several dimensions based on the feedback the algorithm might receive (other than the chosen arm's reward), the modelling of rewards, and the prior knowledge of contexts that the algorithm might use to choose an arm[6]. The scope of this project is limited primarily to the basic model with independent and identically distributed (IID) rewards, called stochastic multi-armed bandits, wherein the reward for each arm is drawn independently from a fixed distribution that depends on the arm a but not on the round t.

Stochastic MAB assumes the following.

- The algorithm observes only the reward for the selected arm, and receives no other feedback.
- The reward for each arm is IID and initially unknown to the algorithm. Each arm a has a reward distribution $\mathcal{D}_a$, and the reward is sampled independently from $\mathcal{D}_a$ every time arm a is selected.
- Per-round rewards are bounded to the interval [0, 1].

Algorithm 0 General Protocol of Stochastic bandits

1: Parameters: K arms, T rounds (both known); reward distribution $\mathcal{D}_a$ for each arm a (unknown).
2: **for** $t = 1, \ldots, T$ **do**
3: Pick some arm a_t
4: Independently sample the reward r_t from distribution $\mathcal{D}_a, a = a_t$
5: Collect reward r_t
6: **end for**

To quantify the performance of algorithms, the following terminologies are introduced.

- The mean reward of arm a is given by, $\mu(a) = \mathbb{E}[\mathcal{D}_a]$.
- The best mean reward is given by, $\mu^* := max_{a \in A}\ \mu(a)$, where the set of all arms is denoted by $\mathcal{A}$.
- The optimal arm, a^*, is the arm with $\mu(a) = \mu^*$.
- The gap of arm a, $\triangle(a) := \mu^* - \mu(a)$, describes how bad arm a is compared to μ^*.

To evaluate the performance of an algorithm across different problem instances, the concept of regret is universally used in MAB literature. The regret of an algorithm at round T, is defined as the difference between the algorithm's cumulative reward and the best-arm benchmark, μ^*T (the expected reward of always playing the optimal arm).

$$R(T) = \mu^*T - \sum_{t=1}^{T} \mu(a_t) \tag{1}$$

Since the arm chosen at round t, a_t, is a random quantity, $R(T)$ is also a random variable, hence the expected regret $\mathbb{E}[R(T)]$ is used towards performance evaluation. In particular, the dependence of $\mathbb{E}[R(T)]$ on the time horizon T is of interest.

Before analysing any algorithm, we will state a theorem[6] that provides a lower bound to the regret of a stochastic bandit problem:

Theorem 1 *For a fixed time horizon T, number of arms K and any bandit algorithm, there exists a problem instance such that,* $\mathbb{E}[R(T)] \geq \Omega(\sqrt{KT})$.

The reader is referred to Slivkins[6] for a detailed proof of the theorem.

3 Methodology

3.1 ϵ-Greedy

The epsilon-greedy algorithm 1 is a quintessential heuristic for addressing the exploration-exploitation dilemma in reinforcement learning, especially in the context of the Multi-Armed Bandit (MAB) problem. In this scenario, each option or 'arm' represents a different action that can be taken, and the goal is to maximize the cumulative reward over time.

The core of the epsilon-greedy strategy lies in its two-fold approach:

1. With probability $1 - \epsilon$, the algorithm exploits the current knowledge, selecting the arm with the highest estimated reward—this is based on the experimental mean, a running average of the rewards received from that arm so far.

2. With probability ϵ, it explores by randomly selecting any arm, allowing for the acquisition of new information that could lead to a better understanding of the optimal action.

Mathematically, the experimental mean after T turns for a given machine i is calculated as:

$$\text{Experimental Mean} = \frac{\sum_{t=1}^{T} r_{i,t}}{p_{i,T}} \tag{2}$$

where $r_{i,t}$ is the reward received from machine i at time t, and $p_{i,T}$ is the number of times that machine has been chosen up to time T.

This simple yet effective method ensures that over time, each action is sampled enough to provide an increasingly accurate estimate of its true reward value. The choice of epsilon (ϵ)) is critical—too high, and the algorithm wastes too many trials on exploration; too low, and it may not explore enough to find the best option. Balancing this parameter is key to the algorithm's success in various domains, from online recommendation systems to clinical trials. Below, the pseudocode delineates the epsilon-greedy approach.

Algorithm 1 Epsilon-Greedy Algorithm for Action Selection

Require: Set of actions A, number of trials N, exploration rate ϵ
Ensure: Action selection strategy for N trials

Initialize action value estimates:
for each action $a \in A$ **do**
    $Q(a) \leftarrow$ initial value (typically 0)
    $N(a) \leftarrow 0$ ▷ Number of times action a was chosen
end for
Main loop:
for trial $= 1$ to N **do**
    $p \leftarrow$ random number between 0 and 1
    if $p < \epsilon$ **then**
        $a \leftarrow$ random action from A
    else
        $a \leftarrow$ action from A with highest $Q(a)$
    end if
    Take action a, observe reward r
    Update action value estimates:
    $N(a) \leftarrow N(a) + 1$
    $Q(a) \leftarrow Q(a) + \frac{1}{N(a)} \times (r - Q(a))$
end for
return Strategy based on Q values

3.2 UCB

The UCB (Upper Confidence Bound) algorithm 2 strategically balances the exploitation of known rewards and the exploration of less certain options within the multi-armed bandit framework. It is particularly effective due to its use of the optimism in the face of uncertainty principle.

3.2.1 Mathematical Formulation

The selection criterion for an arm a using the UCB approach is given by the following expression:

$$UCB(a) = \hat{\mu}_a + \sqrt{\frac{2\ln(T)}{n_a}} \tag{3}$$

where:

- $\hat{\mu}_a$ represents the empirical mean reward of arm a.
- T denotes the total number of trials conducted.
- n_a is the count of selections for arm a.

The UCB algorithm excels in decision-making by balancing two key components:

- The exploitation of arms based on the empirical mean rewards.
- The exploration of arms, encouraged by the confidence interval represented by the second term of the UCB formula. This term decreases as the arm selection count n_a increases, reducing the drive for exploration and favoring exploitation.

By dynamically adjusting the exploration-exploitation trade-off, the UCB algorithm navigates the uncertain environment of the multi-armed bandit problem, improving the decision-making process to maximize the cumulative reward over time.

3.3 Bayesian Bandits and Thompson sampling

Any Multi-Armed Bandit problem instance $\mathcal{I}$ is completely defined by the mean reward vector $\mu \in [0,1]^K$ of the reward distribution $\mathcal{D}_a$ of each of the K arms. The Bayesian bandit problem adds

Algorithm 2 UCB Algorithm for Action Selection

Require: Set of actions A, number of trials N
Ensure: Action selection strategy for N trials
Initialize action value estimates:
for each action $a \in A$ **do**
    $Q(a) \leftarrow$ initial reward obtained by running action a
    $N(a) \leftarrow 1$ ▷ Number of times action a was chosen
    $\texttt{UCB}(a) \leftarrow Q(a) + \sqrt{\frac{2ln(T)}{N(a)}}$ ▷ Upper confident bound calculation
end for
Main loop:
for trial $= 1$ to T **do**
    $\hat{a} \leftarrow$ action with highest action value estimate $\texttt{UCB}(\hat{a})$
    Take action $\hat{a}$, observe reward r
    Update action value estimates:
    $N(a) \leftarrow N(a) + 1$
    $Q(a) \leftarrow Q(a) + \frac{1}{N(a)} \times (r - Q(a))$
    $\texttt{UCB}(a) \leftarrow Q(a) + \sqrt{\frac{2ln(T)}{N(a)}}$ ▷ Upper confident bound calculation
end for
return Strategy based on Q values

the Bayesian assumption that problem instance $\mathcal{I}$ is drawn from know prior probability distribution over a finite support $[0,1]^K$. The goal changes to minimizing the Bayesian regret which is the expectation of the previously defined regret over all problem instance:

$$\texttt{BR}(T) := \mathop{\mathbb{E}}_{\mathcal{I}\sim\mathbb{P}}[\mathbb{E}[R(T)|\mathcal{I}]] \tag{4}$$

3.4 Bayesian updates or posterior calculation

For any Bayesian MAB algorithm the input is a history of sequence of *action-value* pair over round t, which is called a t-history:

$$H_t = ((a_1, r_1), (a_2, r_2), ..., (a_t, r_t)) \in (\mathcal{A} \times \mathbb{R}) \tag{5}$$

Given a feasible t-history H, Bayesian update equation for Bernoulli Bandits is given as:

$$\begin{aligned}\mathbb{P}_H(\mathcal{M}) &:= Pr[\mu \in \mathcal{M}|H_t = H], \quad \forall \mathcal{M} \subset [0,1]^K \\ &= \frac{P(H|\mu)P(\mu)}{P(H)}\end{aligned}$$

It is important to consider following two lemma's related to Bayesian updates:

Lemma 2 *The posterior probability distribution $\mathbb{P}_H$ is independent of the algorithm.*

Lemma 3 *The posterior robability distribution $\mathbb{P}_H$ calcutated at round t can be used as the prior probability distribution $\mathbb{P}$ at round $t+1$.*

3.4.1 Thompson Sampling

Thompson sampling is a very simple algorithm for Bayesian bandits. At each time step the algorithm calculates the posterior distribution $\mathbb{P}_H$ and samples the mean reward vector from the posterior. Then it chooses the best arm based on this sampled. Formally the algorithm is given in 3

3.4.2 The Bernoulli and Beta Distributions

For ease of computation in our experiments, in the context of MAB, each 'arm' or option is assumed to follow a Bernoulli distribution with parameter θ_{arm}, and the number of successes in trials is modeled by a binomial distribution. The Beta distribution is adopted to represent the belief about the success probability of each arm before any data is observed, with parameters α and β corresponding to the counts of prior successes and failures, respectively.

Algorithm 3 Thompson sampling

$H \leftarrow \emptyset$
for $t = 1, \ldots, T$ **do**
    Receive context x_t
    Draw μ_t according to $P_H(\mu|H)$
    Select $a_t = \arg\max_a \mathbb{E}[r_t|x_t, a, \mu_t]$
    Observe reward r_t
    $H \leftarrow H \cup \{(x_t, a_t, r_t)\}$
    $P \leftarrow P_H$
end for

3.4.3 Posterior Updates

The posterior distribution after observing the data also follows a Beta distribution, with updated parameters $\alpha_{\text{posterior}}$ and $\beta_{\text{posterior}}$ reflecting the evidence from successes and failures in trials:

$$\alpha_{\text{posterior}} = \alpha_{\text{prior}} + k$$
$$\beta_{\text{posterior}} = \beta_{\text{prior}} + N - k$$

where k is the number of successes and N the total number of trials.

4 Experiments and Results

In our experiment, we set up a model with five bandit arms, similar to flipping a coin where you can get heads or tails. This setup uses the Bernoulli distribution, which is a simple way to decide how often each arm gives a reward. We focus on two things: Counts, which keep track of how many times each arm is pulled, and Values, which show the average reward from each arm.

We test two different situations in our simulation. In one, the arms have similar rewards, like most arms giving a reward of 0.1 and one giving 0.9. In another, the arms have more varied rewards, like four arms giving 0.8 and one giving 0.9. This helps us see how well the algorithm converges in different cases.

We conduct 5000 independent simulations for each of five epsilon values in our Epsilon Greedy experiments. The large number of simulations ensures reliable average performance metrics, countering the stochastic nature of individual runs and providing a more accurate assessment of the algorithm's effectiveness. Through these experiments, we closely examine how quickly various algorithms identify the best arm, their efficiency in maximizing rewards, and how they minimizes regret over time

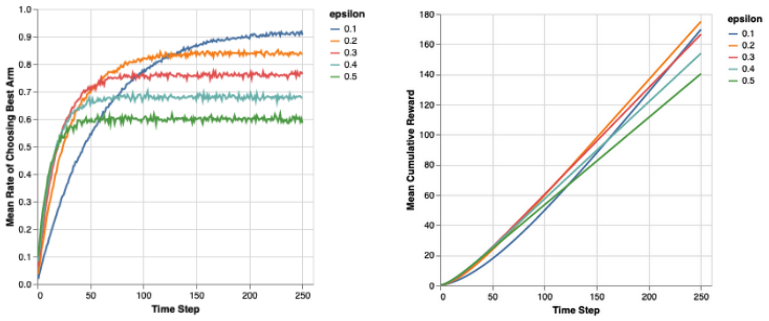

Figure 1: Results for bernoulli arms with distinguishable reward for ϵ-greedy algorithm

From the graphs in Fig: 1 we notice that the epsilon greedy algorithm's performance notably varies with its epsilon value. A higher epsilon (0.5) leads to 60% asymptotic performance, while a lower epsilon (0.1) reaches up to 92%, balancing exploitation and exploration. The algorithm's speed in identifying the best option is faster with higher exploration but varies non-linearly with different epsilon values.

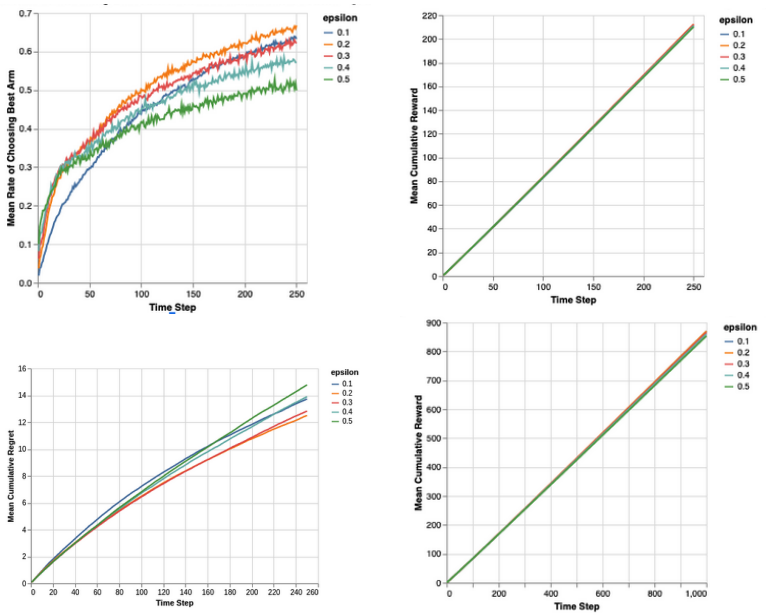

Figure 2: Results for bernoulli arms with similar reward for ϵ-greedy algorithm

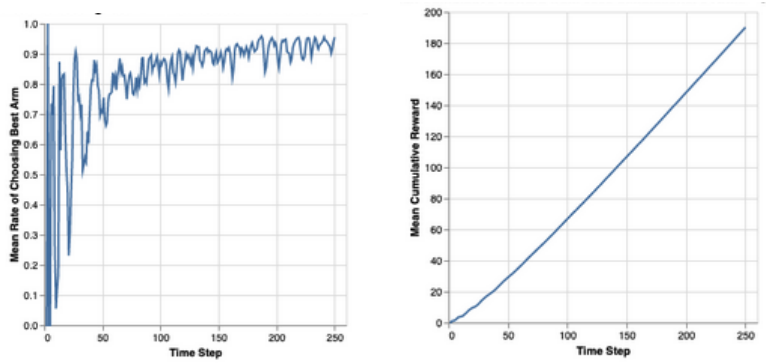

Figure 3: Results for bernoulli arms with distinguishable reward for UCB algorithm

In practical applications, a moderate epsilon (around 0.2) is often most effective, offering a balanced convergence speed and a high reward rate.The rate at which the algorithm identifies the best arm is directly proportional to the level of exploration. Higher exploration rates lead to quicker discovery, but this process is not linear, as seen in the differing convergence speeds of various epsilon values.

In the extended analysis of the epsilon greedy algorithm as Shown in Fig: 2, the focus shifted to a scenario of five arms with close means, four having a mean reward of 0.8 and the best one at 0.9. With these closely clustered rewards, the algorithm understandably struggled to identify the best arm within the 250-step time horizon. Notably, an epsilon of 0.1 was slower in discovering the best arm, and cumulative rewards across different epsilon values were almost indistinguishable due to the close reward means.

The analysis of cumulative regret showed that an epsilon value of 0.2 was most effective, with overall regret ranging from 12.3 to 14.8. Extending the time horizon to 1000 steps led to more distinct results and signs of convergence, highlighting the impact of a longer evaluation period on the algorithm's performance.

For the UCB (Upper Confidence Bound) algorithm, we observe in Fig: 3 that its early phase, especially between time steps 0 to 60, is marked by notable fluctuations in selecting the best arm.

This behavior results from the algorithm's initial strategy of exploring all arms, evident from the larger UCB components for each arm during this period.

As the trial progresses, these components reduce, aligning the UCB values closer to the average reward of each arm. This leads to the more frequent selection of the arm with the highest mean, causing the rate of choosing the best arm to gradually converge towards 1. However, the rate of convergence slows as it approaches 1, and the limited time horizon of the experiment precluded observation of complete convergence. On observing the reward plot we notice an initial curvature, reflecting the early fluctuations in arm selection. However, Over time, as the algorithm consistently identifies the best arm, the plot becomes more linear, suggesting a steady state of picking the best arm, which is expected to yield a reward close to 0.9.

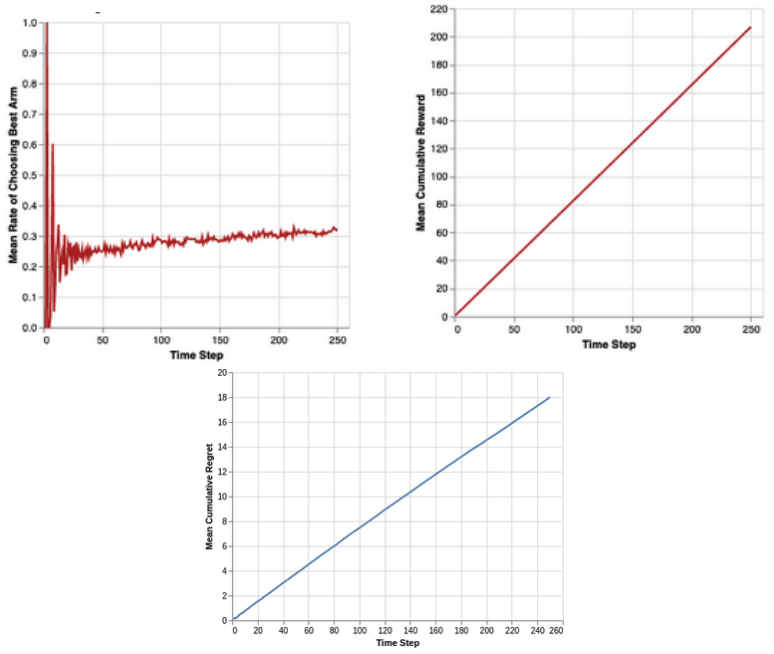

Figure 4: Results for bernoulli arms with similar reward for UCB algorithm

Fig: 4 shows a simulation with closely spaced rewards (five arms, four at 0.8 and one at 0.9). We notice that the UCB algorithm's performance declined, with the best arm chosen only about 32% of the time. This contrasts with the Epsilon Greedy algorithm, which performed better in this scenario, selecting the best arm $50 - 70\%$ of the time. The UCB algorithm's cumulative regret is found to be higher than Epsilon Greedy, suggesting it accumulates more regret as time progresses, especially in situations with closely matched arm rewards.

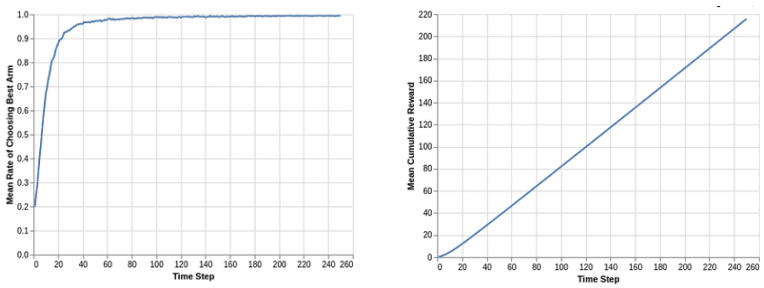

Figure 5: Results for TS with distinguishable arms

The Thompson Sampling algorithm demonstrates remarkable efficiency in a simulation with arms having varied rewards as demonstrated in Figure 5 . It achieves quick convergence, identifying the best arm with around 95% accuracy within just 40 trials. Initially, all arms are viewed equally, starting from a 20% chance of selecting the best arm out of five. However, as the trial progresses, the algorithm's reliance on the playthrough history and updates to the posterior Beta distribution swiftly pinpoints the best arm, displaying a smoother progression than the UCB1 algorithm.

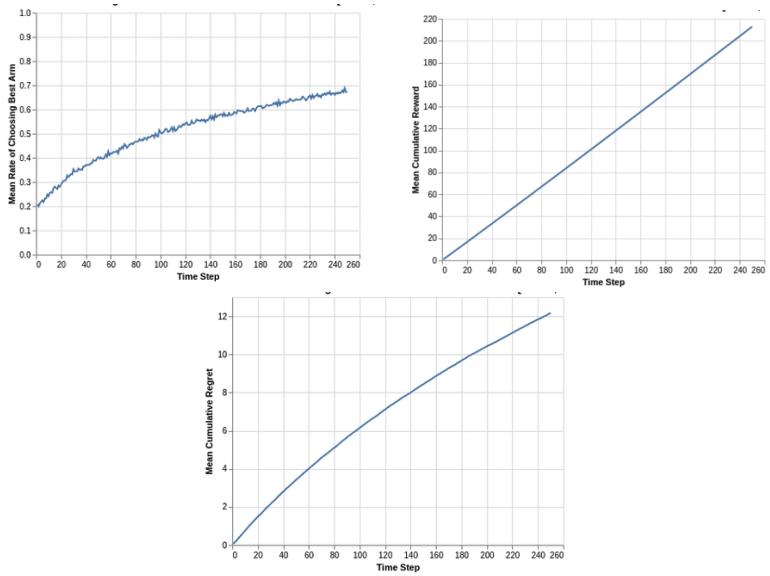

Figure 6: Results for TS with similar arms

In a scenario with arms having closer reward means, the Thompson Sampling algorithm outperforms others UCB, maintaining a higher selection rate of the best arm. This is demonstrated in Figure 6. Its overall performance, in terms of cumulative rewards, also surpasses most algorithms, suggesting its superiority in both rapidly identifying the best option and accumulating rewards effectively over time.

5 Insights and Conclusion

In conclusion, our study compared three Multi-Armed Bandit (MAB) algorithms - ϵ-greedy, UCB, and Thompson Sampling (TS).The initial theoretical expectations suggested that UCB should outperform ϵ-greedy, primarily due to its adept balance between exploration and exploitation, particularly in scenarios with fewer arms and higher reward distribution standard deviations. However, our empirical observations revealed nuanced outcomes.

UCB, as anticipated, excelled in scenarios with fewer arms and higher standard deviations. Nevertheless, a noteworthy decline in performance surfaced as the number of arms increased, introducing a complexity not fully captured by theoretical expectations. In contrast, ϵ-greedy, at times, demonstrated superior practical results, highlighting the importance of empirical validation.In diverse scenarios, Thompson Sampling (TS) demonstrated robust performance, effectively handling arms with varying reward averages. However, potential bottlenecks in TS's sampling from posterior distributions may compromise overall performance, requiring further investigation. Each MAB algorithm—ϵ-greedy, UCB, and TS—has unique strengths and weaknesses. Theoretical expectations didn't consistently align with empirical outcomes, underscoring the need for practical validations.

6 Future Work

Besides the three algorithms that we discussed, other strategies for solving the MAB problem exist.

- Approximate sampling methods: Analyse different approximate sampling methods and observe the performance of Thompson sampling algorithm
- Boltzmann exploration (SoftMax): Selects arms with a probability proportional to the average reward of each arm.
- Value Difference Based Exploration (VDBE): An adaptive epsilon greedy method that dynamically balances exploration and exploitation.
- LinUCB: Tailored for the contextual MAB problem, it uses feature vectors to inform the selection of arms.
- Reinforcement Learning with Neural Networks: A suggested area for future exploration to approximate the value of each action, especially useful in high-dimensional spaces.

7 Contribution

In our project, each team member played a vital role, ensuring a collaborative effort. Shyamvanshikumar took the lead in conducting the literature review, delving into Bayesian bandits and Thompson sampling. Meanwhile, Raghav led the exploration of the ϵ-greedy and UCB algorithms, overseeing the formulation of the problem's motivation and approach. Shivam was in charge of the mathematical formulation and modelling the code along with the undertaking of comparative experiments and results.

References

[1] Code repository. `https://github.com/apoc146/MAB.git`. Accessed: 2023-11-30.

[2] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47:235–256, 2002.

[3] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[4] Eric Mazumdar, Aldo Pacchiano, Yian Ma, Michael Jordan, and Peter Bartlett. On approximate thompson sampling with langevin algorithms. In *International Conference on Machine Learning*, pages 6797–6807. PMLR, 2020.

[5] Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

[6] Aleksandrs Slivkins et al. Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning*, 12(1-2):1–286, 2019.

[7] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.