Proceedings of CONF-MLA 2024 Workshop: Semantic Communication Based Complexity Scalable Image Transmission System for Resource Constrained Devices
DOI: 10.54254/2755-2721/83/2024GLG0077

Applications and Advances of UCB Algorithms in Dynamic and Contaminated Environments

Mengxuan Du

School of Science and Engineering, The Chinese University of Hong Kong, Shen Zhen, China

122090093@link.cuhk.edu.cn

Abstract. The Upper Confidence Bound (UCB) algorithm is a widely used approach in the Multi-Armed Bandit (MAB) problem, where the goal is to maximize cumulative rewards over time by selecting the best possible action among several options. The UCB algorithm uses confidence bounds to balance exploration and exploitation, guiding its decision-making. In recent years, researchers have identified significant challenges in applying the UCB algorithm to dynamic and contaminated environments. In such scenarios, the underlying conditions may change over time, making it difficult for standard UCB to adapt, or the data may be polluted by noise and outliers, leading to incorrect estimations of reward distributions. To address these challenges, several variants of the UCB algorithm have been developed. These new approaches are designed to better handle the complexities of changing environments and data contamination, ensuring more robust and reliable performance in these difficult settings. This paper aims to provide a comprehensive review of Robust-UCB (cr-UCB), Sliding Window UCB (SW-UCB) and bandit-over-bandit UCB (BOB-UCB). Focusing on their theoretical foundations, practical applications, and empirical performance. By examining how these algorithms have been adapted to handle the complexities of dynamic and contaminated environments, we found that the adaptability of these algorithms in dynamic environments is significantly improved, and they can effectively reduce decision-making errors caused by data pollution, thus providing a more reliable solution to the multi-armed bandit problem.

Keywords: UCB contaminated, contaminated environments, dynamic.

1. Introduction

The Upper Confidence Bound (UCB) algorithm is a fundamental technique in addressing the exploration-exploitation trade-off inherent in the Multi-Armed Bandit (MAB) problem. The algorithm has seen widespread adoption and extension, particularly in challenging environments where conditions are either dynamic or contaminated by noise. This paper aims to review and synthesize the recent advances in UCB algorithms specifically designed for these complex scenarios. The UCB algorithm, introduced by Auer, Cesa-Bianchi, and Fischer in 2002, established the foundation for exploration-exploitation strategies by employing the principle of optimism under uncertainty. The authors offer a unified proof technique to analyze these algorithms [1] . In the case of C-UCB, the correlated bandit variant of the Upper Confidence Bound algorithm, a thorough analysis shows that the algorithm pulls certain sub-optimal arms, referred to as non-competitive, only O(1) times, unlike traditional bandit

DOI: 10.54254/2755-2721/83/2024GLG0077

algorithms like UCB or TS, which require O(logT) pulls. One significant line of research has extended UCB to adversarial settings, where an opponent can influence the rewards to mislead the learning agent. Algorithms like EXP3 and EXP3++ incorporate UCB principles with robust mechanisms to maintain performance even under adversarial attacks. Recent work by Bogunovic introduced the Robust-UCB (crUCB) algorithm, specifically designed to handle ε-contaminated environments [2]. This algorithm employs robust mean estimators to mitigate the impact of outliers and adversarial contamination, demonstrating strong performance both theoretically and empirically. In 2011, Garivier and Cappé introduced KL-UCB, which enhances the confidence bounds using Kullback-Leibler (KL) divergence [3]. For instance, KL-UCB-Switch, proposed by Menard, dynamically adjusts between different exploration strategies based on the observed data, achieving optimal regret bounds from both distribution-dependent and distribution-free perspectives [4]. In non-stationary environments, where the reward distribution of each arm may change over time, standard UCB algorithms can perform poorly. Variants such as SW-UCB and D-UCB have been developed to address these challenges by incorporating mechanisms to detect and adapt to changes in the reward distribution [5]. Similarly, contextual bandits, where the agent has access to additional contextual information, have led to the development of algorithms like LinUCB, which integrates linear models with UCB principles to efficiently exploit the context [6]. Recent studies have also explored trade-offs between exploration and exploitation in UCB-based algorithms. Restless bandit problems, where the state of each arm evolves over time according to a Markov process, present additional challenges. The Restless-UCB algorithm, proposed by Wang, Huang, and Lui adapt UCB principles to this dynamic setting [7] and introduce a novel algorithm, Rotting Adaptive Window UCB (RAW-UCB), that achieves near-optimal regret in both rotting rested and restless bandit, without any prior knowledge of the setting (rested or restless) and the type of non-stationarity (e.g., piece-wise constant, bounded variation) [8].

Mridul Agarwal and his coworkers present Limited Communication Collaboration- Upper Confidence Bound (LCC-UCB), a doubling-epoch based algorithm where each agent communicates only after the end of the epoch and shares the index of the best arm it knows Mridul Agarwal and his coworkers present the Limited Communication Collaboration-Upper Confidence Bound (LCC-UCB) algorithm, a novel approach where agents communicate only at the end of each doubling epoch, sharing the index of the best arm they have identified. This method significantly reduces communication overhead while maintaining effective collaboration among agents.

Despite the numerous UCB algorithm variants that have been proposed in recent years to tackle the challenges posed by dynamic and polluted environments, there remains a significant gap in the systematic evaluation and comparison of these algorithms. Many studies have focused on developing and refining specific UCB variants, yet there has been less emphasis on understanding how these algorithms perform relative to each other across different scenarios and environments. This lack of comprehensive assessment leaves open questions about the strengths and weaknesses of each variant, their practical applicability, and the contexts in which they are most effective.

This paper aims to fill this gap by providing a thorough review of the existing literature, synthesizing the key advancements in UCB algorithms for dynamic and contaminated environments, and critically evaluating their performance. By highlighting the comparative strengths and weaknesses of different UCB variants, this review seeks to identify not only the current state of the art but also the areas where further research is needed. This will help guide future developments and ensure that UCB algorithms can be more effectively applied in increasingly complex and challenging real-world scenarios.

2. UCB Algorithm Overview

2.1. Basic UCB Framework

The UCB algorithm operates on the principle of optimistic exploration. At each step, the algorithm selects the action with the highest upper confidence bound, which is calculated based on the estimated reward and a term that accounts for the uncertainty of this estimate. The UCB action selection rule is typically expressed as:

DOI: 10.54254/2755-2721/83/2024GLG0077

$$a_t = argmax_a \left(\hat{}\mu_a + \sqrt{\frac{2logt}{n_a}} \right) \quad (1)$$

where $\hat{}\mu_a$ is the estimated mean reward of action a, t is the current time step, and $n_a$is the number of times action a has been selected. The term $\sqrt{\frac{2logt}{n_a}}$ represents the exploration bonus, which decreases as the action is chosen more frequently.2 Comparison with Other Algorithms

The UCB algorithm stands as one of the most widely used methods in the Multi-Armed Bandit (MAB) problem, where the goal is to identify the optimal strategy for selecting among KKK arms, each offering a random reward from an unknown distribution. UCB is often compared with other popular MAB algorithms, particularly Thompson Sampling (TS) and Epsilon-Greedy (EG), which have different approaches to balancing the exploration-exploitation trade-off.

Thompson Sampling (TS): TS employs a Bayesian approach to decision-making. At each round, it samples from the posterior distribution of the reward for each arm and selects the arm with the highest sampled value. This probabilistic nature makes TS inherently adaptive and often leads to more efficient exploration, as it takes into account both prior knowledge and observed data. One advantage of TS is its strong empirical performance, often achieving lower regret in practice, especially when the reward distributions are highly non-stationary or when prior information is available.

Epsilon-Greedy (EG): The Epsilon-Greedy algorithm introduces randomness into the selection process by choosing a random arm with a small probability ϵ\epsilonϵ, and otherwise selecting the best-known arm. This ensures exploration but can lead to inefficiencies, particularly if ϵ\epsilonϵ is not well-tuned. In environments where exploration is less critical, such as when the optimal arm consistently yields significantly higher rewards than the others, EG can perform adequately. However, its performance tends to degrade in more complex settings where continuous exploration is necessary to track changes in the environment.

The UCB algorithm is distinct due to its deterministic nature, where decision-making relies on calculated upper confidence bounds derived from estimated rewards. This deterministic approach makes UCB highly interpretable, as it provides clear criteria for exploration versus exploitation decisions. The algorithm ensures that each arm is explored sufficiently to establish a reliable estimate of its expected reward, making it robust across various scenarios.

However, UCB can require careful tuning, particularly when dealing with environments where the reward distribution of the arms evolves over time. In such non-stationary environments, simple versions of UCB may continue exploring suboptimal arms for too long, leading to higher regret compared to algorithms like TS, which can more quickly adapt to changing conditions. This is why variants of UCB, such as SW-UCB (Sliding Window UCB) and D-UCB (Discounted UCB), have been developed to enhance performance in dynamic environments by emphasizing recent rewards over older data.

The performance of UCB in KKK-armed bandit problems varies depending on several factors, including the number of arms, the disparity in reward distributions, and the presence of non-stationarity.

- Small KKK and Large Reward Differences: When the number of arms is small and there is a significant gap between the best and suboptimal arms, UCB performs very well. The algorithm quickly identifies the optimal arm and focuses on exploitation, leading to low regret.
- Large KKK and Similar Rewards: In scenarios with a large number of arms where the rewards are similar, UCB might struggle more compared to TS. This is because UCB could waste time exploring arms that are close in performance to the optimal arm. TS, with its probabilistic nature, tends to balance exploration more efficiently, especially when arms have similar expected rewards.
- Non-Stationary Environments: In environments where the reward distributions change over time, UCB's deterministic strategy might lead to slower adaptation. Variants like SW-UCB or D-UCB can help mitigate this by focusing more on recent data, but in highly dynamic settings, TS often outperforms UCB due to its ability to continuously sample and adapt to changes.

In summary, UCB excels in environments where the rewards are relatively stable, and the differences between arms are pronounced. However, in more complex, non-stationary environments or when the

DOI: 10.54254/2755-2721/83/2024GLG0077

number of arms increases, TS often demonstrates superior adaptability and lower regret. The deterministic nature of UCB remains a strength in terms of interpretability, but it necessitates careful tuning and modification to remain competitive with more adaptive algorithms like TS.

2.2. Characteristics of UCB

UCB's primary advantage lies in its ability to provide a theoretical guarantee on the regret, which measures the difference between the cumulative reward of the best possible policy and that achieved by the algorithm. UCB algorithms have sub-linear regret bounds, ensuring that as the number of time steps increases, the average regret per step decreases. The UCB algorithm is well-regarded not only for its practical utility but also for its strong theoretical guarantees. It has been shown to achieve logarithmic regret in stochastic bandit settings, which is optimal up to constant factors. The regret of UCB is defined as the difference between the expected reward of the optimal strategy and the reward actually obtained by the algorithm: $R(T)=\sum_{t=1}^{T}(\mu^{*}-\mu_{a_t})$. In 2010, Auer and Ronald Ortner modified UCB algorithm achieves a tighter regret bound of $O(\sum_i:\Delta_i> O\frac{\log(T\Delta_i^2)}{\Delta_i})$, [10]This represents a significant improvement, especially for arms with expected rewards close to the optimal arm, as the dependence on Δ_i inside the logarithmic term allows for finer distinctions between arms.

3. UCB in Contaminated Environments

In contaminated environments, where the data is subject to noise, outliers, or even adversarial attacks. Standard UCB algorithms often struggle due to the incorrect estimation of reward distributions. Such environments can severely impact the performance of traditional UCB algorithms, which rely on accurate reward estimates to balance exploration and exploitation effectively. The presence of contaminated or corrupted data can lead to overly optimistic or pessimistic confidence bounds, causing the algorithm to either overexplore suboptimal arms or underexploit potentially optimal ones.

To address these challenges, Niss and Tewari proposed the Contaminated Robust UCB (crUCB) algorithm,[2] which integrates robust statistical techniques designed to handle contaminated data. The crUCB algorithm enhances the traditional UCB framework in significant ways. First, it introduces methods that are specifically designed to be resilient to outliers, ensuring that extreme or atypical reward observations do not disproportionately influence the learning process. Additionally, crUCB incorporates techniques to handle noise in the reward observations, allowing the algorithm to maintain robust performance even when the data is not clean or reliable.

This is achieved by leveraging robust estimators, such as the median or trimmed mean, instead of the sample mean, to calculate the expected rewards for each arm. These robust estimators are less sensitive to extreme values, making the crUCB algorithm more reliable in environments with high levels of contamination.

The crUCB algorithm also incorporates robust confidence bounds that adjust for the potential contamination of data. By doing so, it ensures that the exploration-exploitation trade-off is managed more effectively, even when the data is noisy or adversarial. This approach not only improves the algorithm's resilience to corrupted data but also ensures that it remains competitive in both clean and contaminated environments.

Empirical results in their paper demonstrate that the crUCB algorithm significantly outperforms traditional UCB algorithms in contaminated settings, maintaining lower regret and better adaptability. This makes crUCB a valuable tool in practical applications where data contamination is a concern, such as in recommendation systems, financial markets, and sensor networks, where adversarial interference or measurement noise can distort reward signals.

3.1. crUCB Algorithm

The crUCB algorithm modifies the standard UCB by adjusting the reward estimation process to account for potential contamination. This is achieved through techniques like trimming or Winsorizing, $\hat{\theta}_t$ which

DOI: 10.54254/2755-2721/83/2024GLG0077

reduce the influence of extreme values. The algorithm still follows the UCB selection rule but with a robust estimator of the mean reward:

$$a_t = argmax_a \left(\hat{\mu}_a^{robust} + \sqrt{\frac{2logt}{n_a}} \right) \quad (2)$$

where $\hat{\mu}_a^{robust}$ is a robust estimate of the mean reward.

Niss and Tewari present the contamination robust-UCB (crUCB) algorithm for-CSB with sub-Gaussian rewards, shown as below:

Algorithm 1: crUCB

Input: number of actions K, time horizon T, upper bound on fraction contamination α, upper bound on sub-Gaussian constant σ_0, mean estimate function (α trimmed or shorth mean) f.

1. **For** t ≤ K **do**
   Pick action a when t = a.
   end
2. **For** t > K **do**:
   For a ∈ [K], compute **do**
   $f(x_a(t))$ ← mean estimate of rewards.
   $N_a(t)$ ← number of times action has been played.
   end
   Pick action $A_t = \text{argmax}_{a \in [K]} (f(x_a(t)) + (\sigma_0 / (1 - 2\alpha)) * \text{sqrt}(4 \log(t) / N_a(t)))$.
   Observe reward $x_{at}(t)$.
   end

This offers uncontaminated regret assurances for the trimmed and shorth means, respectively, for crUCB below. We point out that implementing our algorithms is not difficult. In real-world applications, upper bounds on the parameters that are resistant to underestimates are frequently simple to locate. Our algorithms exhibit unambiguous tuition to their actions and are statistically stable. These algorithms' drawback is that they necessitate prior knowledge. Decisions about could need independent research, but they could also stem from domain expertise. We considered a completely adaptive adversarial contamination in this work, with the overall fraction of contamination at each time step serving as the only constraint. It could be able to increase uncontaminated remorse bounds by assuming more about the opponent.

Another type of contamination is considered as a data poisoning attack, also referred as man in the middle (MITM) attack [11]. In this attack, there are three agents: the environment, the learner (MAB algorithm), and the attacker. At each discrete time-step t, the learner selects an action it among K choices, the environment then generates a reward rt(it) ∈ [0,1] corresponding to the selected action, and attempts to communicate it to the learner. However, an adversary intercepts rt(it) and can contaminate it by adding noise $\varepsilon_t(i_t) \in [-r_t(i_t), 1 - r_t(i_t)]$. It follows that the learner observes the contaminated reward $r_t^0(i_t) = r_t(i_t) + \varepsilon_t(i_t)$, and $r_t^0(i_t) \in [0,1]$. Hence, the adversary acts as a "man in the middle" between the learner and the environment. This can be solved by Secure UCB (for Secure Upper Confidence Bound), which integrates verification into the classical UCB algorithm, and also enjoys similar order-optimal regret bounds and order optimal expected number of verifications.

3.2. Applications of crUCB

The crUCB algorithm has demonstrated strong performance in scenarios where data is vulnerable to contamination, making it highly suitable for real-world applications where noise, outliers, and adversarial disruptions are prevalent. For instance, in financial markets, crUCB can help navigate irregular events, such as market shocks, crashes, or unexpected news that could skew traditional reward

DOI: 10.54254/2755-2721/83/2024GLG0077

estimations. By applying robust statistical techniques, crUCB reduces the impact of such anomalies, ensuring that investment strategies remain more reliable even in volatile and unpredictable conditions.

Another critical application of crUCB is in sensor networks, where faulty sensor readings can distort the data, leading to suboptimal decisions. In environments where sensors are prone to malfunction due to harsh conditions or technical issues, crUCB can mitigate the influence of erroneous data, ensuring that the system continues to make accurate predictions and decisions. This makes crUCB a valuable tool in areas such as environmental monitoring, healthcare, and smart infrastructure, where maintaining data integrity is crucial for effective operation.

In addition to these, crUCB can be applied in online recommendation systems, where user behavior might be subject to unusual or adversarial actions, such as click fraud or spamming. By making the learning process less sensitive to these anomalies, crUCB enables more robust and reliable recommendations, even in the presence of data contamination. This improves user experience and ensures that the system adapts effectively to changing conditions without being misled by outliers. While crUCB offers substantial improvements over traditional UCB algorithms in contaminated environments, it is not without limitations. One of the primary challenges is that the incorporation of robust statistical methods can introduce additional computational complexity. Techniques such as the trimmed mean or median are computationally more intensive than simple averaging, especially as the number of arms or the amount of data increases. This can lead to slower processing times, making crUCB less suitable for scenarios that require real-time decision-making with limited computational resources.

Additionally, crUCB's performance may still degrade in environments with extreme levels of contamination or where the nature of the contamination is highly unpredictable. Robust statistical methods, while effective, are not foolproof and can struggle when the proportion of contaminated data becomes very large or when the contamination patterns are complex and non-standard. In such cases, more advanced or tailored robust methods might be necessary to maintain performance

4. UCB in Dynamic and Non-Stationary Environments

Dynamic and non-stationary environments present unique challenges for learning algorithms, particularly in scenarios where reward distributions change over time. Unlike static environments, where the expected rewards for each action remain constant, dynamic environments are characterized by shifts in the underlying reward structure due to factors such as time-dependent trends, external interventions, or evolving system states. Two of the main obstacles to making decisions in many scientific fields, such as engineering, economics, social science, neurology, and ecology, are uncertainty and the nonstationary of the environment. In such environments, an effective strategy necessitates balancing multiple trade off: remembering-versus-forgetting (i.e., using more but possibly outdated information or using less but recent information) and exploration-versus-exploitation (i.e., choosing between the most informative and the empirically most rewarding alternatives).[12] An official way to express the exploration-versus-exploitation trade off is the stochastic MAB problem. An agent chooses one among K alternatives in an MAB issue each time, and it is rewarded accordingly. It is assumed that each option's reward sequence is the result of an unknown, i.i.d. random process. The MAB formulation has been applied in many scientific and technological areas. These changes violate the stationarity assumptions of standard UCB algorithms, which typically rely on the assumption that the reward distribution for each arm remains fixed throughout the learning process. As a result, standard UCB algorithms can become slow to adapt, leading to suboptimal performance and higher cumulative regret in such settings.

To address these challenges, researchers have developed several UCB variants that are specifically designed to handle dynamic and non-stationary environments. These variants modify the standard UCB framework to allow for more flexible adaptation to changing conditions, ensuring that the exploration-exploitation trade-off is continuously optimized as the environment evolves. In conclusion, these UCB variants enhance the adaptability of the standard UCB algorithm, enabling it to perform effectively in dynamic and non-stationary environments. By incorporating mechanisms such as sliding windows, discount factors, and continuous adjustment of strategies, these algorithms ensure that the learning process remains responsive to changes in the underlying reward structure, ultimately leading to better

DOI: 10.54254/2755-2721/83/2024GLG0077

decision-making and lower regret in complex, evolving environments. However, careful tuning of parameters such as window size or discount factor is essential to ensure optimal performance across different types of non-stationary environments

4.1. Restless Bandit Problems and Restless-UCB Algorithm

The Restless Bandit problem is a specific formulation of the multi-armed bandit problem that is well-suited to dynamic environments. In this problem, the reward distributions of the arms evolve over time, regardless of whether they are selected. Wang introduced the Restless-UCB algorithm, which adapts the traditional UCB framework to this setting by continuously adjusting the selection strategy to account for the changing environment [7]. Restless-UCB demonstrates strong adaptability, making it effective in complex scenarios where both selected and unselected arms can change over time. This flexibility allows it to outperform traditional UCB algorithms in highly dynamic and non-stationary environments, particularly in applications such as online recommendations, dynamic pricing, and real-time decision-making.

In this section, we present our Restless-UCB policy, shown as below:

Algorithm 2: Restless-UCB Policy

1: **Input**: Time horizon *T*, learning function m(*T*).
2: **for** i=1,2, … , N **do**
3: Choose arm I until there are m(T) times we observe $s_i(t) = k$ for all states k.
4: **end for**
5: Let $\hat{P}_i(j,k)$'s and $\hat{r}_i(i,k)$'s be the empirical values of $P_i(j,k)$'s and *r(i,k)'s*.
6: Construct instance R' with $r'(i,k) = \hat{r}(i,k) + rad(T).P_i'(k,k+1) = \hat{P}_i(k,k+1) - rad(T)$ $P_i'(k,k) = \hat{P}_i(k,k), P_i'(k,k-1) = \hat{P}_i(k,k-1) + rad(T)$ Specifically, $P_i'(1,1) = \hat{P}_i(1,1) + rad(T)$ and $P_i'(M,M) = \hat{P}_i(M,M) - rad(T)$.
7: Find the optimal policy $\pi^{*'}$ for problem $R', i.e., \pi^{*'} = Oracle\left(R'\right)$.
8: **while true do**
9: Follow $\pi^{*'}$ for the rest of the game.
10: **end while**

There are two sections in Restless-UCB: (i) the investigating section (lines 2-4) and (ii) the exploiting section (lines 5–10). Accurately determining the parameters {Pi, i = 1,2…. N} and {r(i, s), i, s} is the aim of the exploration phase. For any action i and state k, we witness the next transition and the offered reward for at least m(T) number of times (m(T) to be determined later). Restless-UCB accomplishes this by pulling each arm until there are sufficient observations.

Then, without any prior information of the environment (rested or restless) and the type of non-stationarity, we offer a unique method, Rotting Adaptive Window UCB (RAW-UCB) [10], shown as below, that yields nearly optimal regret in both rotting relaxed and restless bandit scenarios.

Algorithm 3: RAW-UCB

Input: K, σ, α

1. **For** t ← 1, 2, ..., K **do**: ▷ Pull each arm once
2. PULL $i_t \leftarrow$ t; RECEIVE o_t; $N_{i_t} \leftarrow 1$
3. $\left\{\hat{u}_{i_t}^h\right\}_h \leftarrow$ UPDATE($\left\{\hat{u}_{i_t}^h\right\}_h$, o_t) ▷ cf. (1)
4. **End for**
5. **For** t ← K + 1, K + 2, ... **do**:
6. PULL $i_t \in$ argmax$_i min_h \leq N_i \hat{u}_{i_t}^h$ + c(h, δ_t) ▷ cf. (3)

DOI: 10.54254/2755-2721/83/2024GLG0077

7. RECEIVE o_t; $N_{i_t} \leftarrow N_{i_t} + 1$
8. $\hat{u}_{i_t}^h \leftarrow$ UPDATE($\hat{u}_{i_t}^h$, o_t) ▷ cf. (1)
9. **End for**

.

The RAW-UCB algorithm is designed to address the bias-variance trade-off inherent in choosing the window size for computing the Upper Confidence Bound. A smaller window size increases variance because it relies on fewer data points, making the estimates more sensitive to recent fluctuations. Conversely, a larger window size reduces variance but introduces bias, as it includes outdated information that may not reflect the current state of the environment. The objective of RAW-UCB is to dynamically select the optimal window size to achieve the tightest possible UCB.

To do this, RAW-UCB calculates UCB indexes for all possible slices of each arm's history, focusing on segments that include the most recent pull. When rewards are decaying or "rotting," these indexes act as upper confidence bounds on the next reward value. RAW-UCB then selects the tightest (i.e., minimum) index for each arm, making it a pure UCB-based algorithm.

In environments where rewards can increase over time, the situation becomes more complex. In such cases, UCB indexes based on past values become less informative about future rewards. This is why algorithms designed for restless and non-stationary environments often incorporate additional mechanisms, such as change detection subroutines, active random exploration strategies, or passive forgetting techniques. These mechanisms allow the learner to adapt to changes in the reward distribution, ensuring that the algorithm remains responsive to shifts in the environment while balancing exploration and exploitation effectively

4.2. SW-UCB, D-UCB Algorithms and BOB-UCB

In non-stationary environments, SW-UCB and discounted UCB (D-UCB) are two notable variants that adjust to changing reward distributions by dynamically tuning their exploration-exploitation balance. The SW-UCB algorithm, for instance, uses a sliding window to focus on recent rewards, discarding outdated information. The D-UCB algorithm, on the other hand, adjusts its confidence bounds based on detected changes in the environment.

Below describes the details of the SW-UCB algorithm. Following its design guideline, the SW-UCB algorithm selects a positive regularization parameter (>0), and initializes V0= I*λ. In each round t, the SW-UCB algorithm first computes the estimate θ_t for θ_t, and then finds the action x_t with largest UCB by solving the optimization problem. Afterwards, the corresponding reward y_t is observed. The pseudocode of the SW-UCB algorithm is shown as below.

Algorithm 4: SW-UCB algorithm

Input: Sliding window size w, dimension d, variance proxy of the noise terms R, upper bound of all the actions' ℓ_2 norms L, upper bound of all the θ_t's ℓ_2 norms S, and regularization constant λ.

1. **Initialization**: $V_0 \leftarrow \lambda I$.
2. **For** t = 1, ..., T **do**:
3. $\widehat{\theta_t} \leftarrow V_{t-1}^{-1} \left(\sum_{s=1V(t-w)}^{t-1} X_s Y_s\right)$.
4. $X_t \leftarrow argmax_{x \in D_t} \{(x^T \widehat{\theta_t} + \|x\|_{V_{t-1}^{-1}} [R\sqrt{dln(\frac{1+\frac{wL^2}{\lambda}}{\delta})} + \sqrt{\lambda}S]\}$
5. $Y_t \leftarrow \langle X_t, \theta_t \rangle + \eta_t$.
6. $V_t \leftarrow \lambda I + \sum_{s=1V(t-w)}^{t-1} X_s X_s^t$
7. **End for**

D-UCB and SW-UCB are both designed to handle non-stationary environments in multi-armed bandit problems, but they approach the problem in different ways. D-UCB uses a discount factor to reduce the

DOI: 10.54254/2755-2721/83/2024GLG0077

influence of older rewards, emphasizing more recent data, which helps it adapt to gradual changes in the environment. In contrast, SW-UCB maintains a fixed-size sliding window of recent rewards and only considers data within this window, effectively "forgetting" past information outside of it. Both methods share the goal of adapting UCB to non-stationary settings, but D-UCB offers a continuous decay of older data, while SW-UCB applies a more abrupt cutoff.

We are motivated to employ the SW-UCB algorithm as a subroutine and "hedge" against the modifications of the previously mentioned observations and established outcomes. The SW-UCB algorithm's details are described below. Finding a suitable fixed window length is the first step in adhering to its design guidelines. In order to do this, we outline the fundamental principle of the Bandit-over-Bandit (BOB) algorithm. The BOB method designates a set J (∈[H]) from which each is taken after dividing the entire time horizon into [T/H] blocks of equal length H rounds (the final block may have less rounds than H rounds). The BOB method first chooses a window length w_i(∈J) for each block i2 [dT/He], and then it starts a fresh instance of the SW-UCB algorithm using the chosen, shown as figure 5.

Algorithm 5: BOB algorithm

1.**Input**: Time horizon T, the dimension d, variance proxy of the noise terms R, upper bound of all the actions' ℓ2 norms L, upper bound of all the θt's ℓ2 norms S, and a constant λ.
2.**Initialize** H, Δ, J by eq. (22), γ, $\{s_j, l\}$Δ=0 by eq. (23).
3.**for** i = 1, 2, ..., ⌊T/H⌋ **do**
4. Define distribution $(p_{j,i})_{j=0}^{\Delta}$
5. Set $j_t \leftarrow$ j with probability $p_{j,i}$.
6. $w_i \leftarrow \lfloor H^{j_t/\Delta} \rfloor$.
7. $V_{(j-1)H} = \lambda I$.
8. **for** t = (i−1)H + 1, ..., i H ∧ T **do**
9. $\widehat{\theta_t} \leftarrow V_{t-1}^{-1} \left(\sum_{s=1V(t-w)}^{t-1} X_s Y_s\right)$
10. Pull arm $X_t \leftarrow argmax_{x \in D_t} \{(x^T \widehat{\theta_t} + \|x\|_{V_{t-1}^{-1}} [R \sqrt{dln(\frac{1+\frac{wL^2}{\lambda}}{\delta})} + \sqrt{\lambda} S]\}$
.
11. Observe $Y_t = \langle X_t, \theta_t \rangle + \eta_t$
12. $V_t \leftarrow \lambda I + \sum_{s=1V(t-w)}^{t-1} X_s X_s^t \top$.
13. **end for**
14. Define $s_{j,i+1}$ according to eq. (25).
15. Define $s_{u,i+1} \leftarrow s_{u,i}$ ∀u ≠ j_i
16. **end for**

BOB-UCB, D-UCB, and SW-UCB all address non-stationary environments but in different ways. BOB-UCB takes a hierarchical approach, where a top-level bandit chooses between various strategies, including variants like D-UCB and SW-UCB, based on the environment's dynamics. D-UCB uses a discount factor to gradually reduce the influence of older data, adapting to gradual changes. SW-UCB, on the other hand, relies on a fixed sliding window to focus on recent rewards, effectively discarding older data. The key connection is that BOB-UCB can incorporate both D-UCB and SW-UCB strategies, dynamically selecting the most suitable one for different phases of non-stationarity.

Based on these successful experiences, researchers have further explored how to combine contextual information to optimize the decision-making process. Next, we will introduce the application of contextual bandits in non-stationary environments.

DOI: 10.54254/2755-2721/83/2024GLG0077

4.3. Contextual Bandits in Non-Stationary Environments

In non-stationary environments, contextual bandits have also benefited from the UCB framework. Zhou introduced the Contextual Sliding Window UCB (CSW-UCB) algorithm, which adapts to changing environments by considering context-specific reward distributions. The algorithm employs a sliding window approach, focusing on recent contexts and their associated rewards, making it particularly effective in environments where both context and reward distributions are subject to change.

Building on this, KL-UCB (Kullback-Leibler Upper Confidence Bound) [4] and KL-Neutral UCB algorithms have further advanced the UCB framework's application in non-stationary settings. KL-UCB calculates confidence bounds based on the Kullback-Leibler divergence, which allows for a more precise adjustment of estimates for unknown reward distributions, addressing the challenges posed by changes in reward distributions. Meanwhile, KL-Neutral UCB combines Kullback-Leibler divergence-based confidence bounds with a neutral treatment of newly observed contexts, enhancing adaptability in dynamic environments. These advancements enable UCB algorithms to maintain higher exploration efficiency and accuracy in evolving scenarios.

In contextual bandit problems, Neutral-UCB (Neutral Upper Confidence Bound) is designed to handle environments where the distribution of rewards or the context itself may shift over time. Unlike traditional UCB methods that might overly rely on past observations, Neutral-UCB incorporates a strategy to handle new contexts more effectively.

The Neutral-UCB algorithm operates by maintaining a balance between exploration and exploitation. It uses a neutral approach to integrate information from newly encountered contexts without disproportionately favoring or penalizing them based on limited data. This is achieved by adjusting the confidence bounds in a way that accommodates changes in the context space while ensuring that recent, less explored contexts are given fair consideration.

In practical terms, Neutral-UCB dynamically updates its estimates and confidence bounds to reflect the most current information. This adaptability is crucial in non-stationary environments where the reward distributions may evolve. By applying Neutral-UCB, practitioners can improve the performance of contextual bandit algorithms in scenarios where traditional methods might struggle due to rapid or unpredictable changes in context or reward distributions.

In summary, the UCB variants discussed in Chapter 4, including Restless-UCB, D-UCB, SW-UCB, and BOB-UCB, all offer unique approaches to addressing the challenges of dynamic and non-stationary environments. Restless-UCB adapts to evolving reward distributions by continuously adjusting its strategy, making it well-suited for highly dynamic scenarios. D-UCB and SW-UCB employ different mechanisms—discounting and sliding windows, respectively—to manage the trade-off between recent and historical data, each with its strengths depending on the rate and nature of environmental changes. BOB-UCB stands out with its hierarchical structure, allowing it to dynamically select the most appropriate strategy among these and other algorithms based on the current phase of the environment.

While these algorithms have shown considerable promise, there remain opportunities for further research. Future work could explore hybrid approaches that combine the strengths of multiple UCB variants, develop more adaptive mechanisms for environments with unpredictable changes, or apply these algorithms to new, complex real-world scenarios. Additionally, deeper theoretical analyses could help to better understand the limits and potentials of these algorithms in various non-stationary contexts.

5. Conclusion

This paper provides a comprehensive review of the applications of UCB algorithms in dynamic and polluted environments, with key findings including the following:

Current UCB algorithms face several limitations when applied to polluted and non-stationary environments. In polluted environments, even algorithms like crUCB, which are designed to handle noise and outliers, may struggle with extreme cases of data contamination or highly skewed distributions. These algorithms can also suffer from increased computational complexity and may not effectively address all types of pollution due to their assumptions about noise characteristics. In non-stationary environments, UCB variants such as Restless-UCB, SW-UCB, and D-UCB may not adapt quickly

DOI: 10.54254/2755-2721/83/2024GLG0077

enough to rapid or frequent changes, potentially leading to suboptimal performance. These algorithms might struggle to balance exploration and exploitation effectively and may face scalability issues as the complexity of the environment increases.

Future research directions could involve developing more robust methods for handling extreme data contamination, enhancing adaptation mechanisms to better respond to rapid changes, and optimizing computational efficiency to manage large-scale and complex environments more effectively. These improvements will be crucial for advancing the application of UCB algorithms in increasingly challenging real-world scenarios.

References

[1] Cappé, O., Garivier, A., Maillard, O. A., Munos, R., & Stoltz, G. (2013). Kullback-Leibler upper confidence bounds for optimal sequential allocation. The Annals of Statistics, 1516-1541.

[2] Niss, L., & Tewari, A. (2020, August). What You See May Not Be What You Get: UCB Bandit Algorithms Robust to $\varepsilon $-Contamination. In Conference on Uncertainty in Artificial Intelligence (pp. 450-459). PMLR.

[3] Garivier, A., & Cappé, O. (2011, December). The KL-UCB algorithm for bounded stochastic bandits and beyond. In Proceedings of the 24th annual conference on learning theory (pp. 359-376). JMLR Workshop and Conference Proceedings.

[4] Song, Y. (2022). Truncated LinUCB for Stochastic Linear Bandits. arXiv preprint arXiv:2202.11735. 103

[5] Garivier, A., Hadiji, H., Menard, P., & Stoltz, G. (2022). KL-UCB-switch: optimal regret bounds for stochastic bandits from both a distribution-dependent and a distribution-free viewpoints. Journal of Machine Learning Research, 23(179), 1-66.

[6] Agarwal, M., Aggarwal, V., & Azizzadenesheli, K. (2022). Multi-agent multi-armed bandits with limited communication. Journal of Machine Learning Research, 23(212), 1-24.

[7] Wang, S., Huang, L., & Lui, J. (2020). Restless-UCB, an efficient and low-complexity algorithm for online restless bandits. Advances in Neural Information Processing Sys

[8] Gupta, S., Chaudhari, S., Joshi, G., & Yağan, O. (2021). Multi-armed bandits with correlated arms. IEEE Transactions on Information Theory, 67(10), 6711-6732.

[9] Seznec, J., Menard, P., Lazaric, A., & Valko, M. (2020, June). A single algorithm for both restless and rested rotting bandits. In International Conference on Artificial Intelligence and Statistics (pp. 3784-3794). PMLR.

[10] Wang, S., Huang, L., & Lui, J. (2020). Restless-UCB, an efficient and low-complexity algorithm for online restless bandits. Advances in Neural Information Processing Systems, 33, 11878-11889.

[11] Rangi, A., Tran-Thanh, L., Xu, H., & Franceschetti, M. (2022, June). Saving stochastic bandits from poisoning attacks via limited data verification. In Proceedings of the aaai conference on artificial intelligence (Vol. 36, No. 7, pp. 8054-8061)

[12] Cheung, W. C., Simchi-Levi, D., & Zhu, R. (2019, April). Learning to optimize under non-stationarity. In The 22nd International Conference on Artificial Intelligence and Statistics (pp. 1079- 1087). PMLR.

[13] Bozgan, Kerem. Robust optimization of multi-objective multi-armed bandits with contaminated bandit feedback. Diss. bilkent university, 2022.