Outlier Detection in Multi-Armed Bandits

A Project Report

submitted by

HANUMATH PRANAV BHARADWAJ KORRAPATI

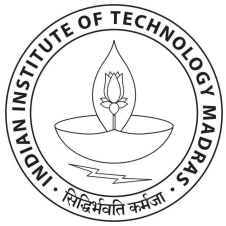
in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

&

MASTER OF TECHNOLOGY



DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

JUNE 2021

THESIS CERTIFICATE

This is to certify that the thesis titled **Outlier Detection in Multi-Armed Bandits**, submitted by **Hanumath Pranav Bharadwaj Korrapati**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology & Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Srikrishna Bhashyam
Research Guide
Professor
Dept. of Electrical Engineering
IIT-Madras, 600036

Date: 18th June 2021

Place; Chennai

ACKNOWLEDGEMENTS

I am very grateful to my research guide **Professor Srikrishna Bhashyam** for giving me the opportunity to work with him. The mini project I did under his guidance laid a great foundation of interest for me in this area. His guidance and support have been invaluable throughout the project. Learnt a great lesson from this project that I should always cross check results and run simulations for a higher number of episodes for desirable results. I would also like to thank my family wholeheartedly without whom I couldn't have finished my project during these tough times.

ABSTRACT

Multi-Armed Bandits problem is a very prominent field of study for the past 50 years. It's use cases are in Internet traffic routing, Marketing, Medical technology, Recommendation systems etc. In this work we will be talking about a specific problem in MAB setting called outlier detection. Our target is to find the outlier arms with least number of samples in a fixed confidence setting.This is different from the **"Top-K"** arm problem and the **"Threshold"** problem as neither the number of outliers nor the threshold value are known before hand. We need to estimate mean values of arms simultaneously along with the threshold.

Then we will be exploring novel algorithms developed in the recent years to solve the classic "best arm" problem. We will be comparing performance of these novel algorithms with the existing traditional algorithms. Simulations will be done using synthetic data along with suitable collected data available on the internet. All my implementations are here: `https://github.com/pranav973/Outlier-detection-in-Bandits`

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

MAB	Multi armed Bandit
RR	Round Robin
WRR	Weighted Round Robin
ADE	Adaptive Double Estimation
RBMLE	Reward biased Maximum Likelihood estimation
LUCB	Lower Upper Centrality Bound
HDOC	Hybrid Dilemma Of Confidence

CHAPTER 1

INTRODUCTION

Multi-armed bandit:

The classic multi-armed bandit setting has a fixed number of arms with underlying stationary distributions. At each round arms are drawn according to some heuristic and a reward is obtained by sampling the arm chosen. The samples are considered to be independent. In a regular setting the reward received at each time step is viewed as a feedback to optimize exploration and exploitation. But the problem we have considers the number of pulls as cost and is therefore a pure exploration problem. All the algorithms we study in outlier detection are $\delta - PAC$. The goal is to predict the outlier set accurately in as less samples as possible.

1.1 Outlier definition

There have been a lot of instances of "Outliers" in the field of data science. A large amount of work is done in the area of identifying outlier data points from the observed data points in a data set. We concentrate on finding the arms which generate data that are substantially different from other sources. (Carpentier and Valko, 2014) defines extreme bandits by looking at the sampled rewards in each round and choosing the arm with highest sample at every instant. The algorithm "Extreme hunter" favours the arm with a heavy tail function as it's most probably the one which can produce large samples. On contrary our work is to identify arms with highest expected reward in a sense.

1.2 Problem setting

We study to identify outlier arms with extremely high reward expectations compared to other arms. Our setting has n arms with stationary distributions and all the samples

drawn are independent from each other. We denote $x_i^{(j)}$ to denote the j^{th} sample of arm i. The true expected rewards are represented by $y_i$for i^{th}arm. To define the outliers we use the standard $k\sigma$ rule over the expected rewards. Let μ_y and σ_y be the mean and standard deviation of expected rewards.

$$\mu_y = \frac{\sum_i y_i}{n}, \sigma_y = \sqrt{\frac{\sum_i (\mu_y - y_i)^2}{n}}$$

We define the the threshold for an outlier to be :

$$\theta = \mu_y + k\sigma_y$$

Mathematically we define the set of outliers as

$$\Omega = \{i \epsilon [n] \, | y_i > \theta\}$$

We use mean and standard deviation estimates at every time step to have an estimated threshold and thus classify arms as either active or inactive based on the threshold value at that instant.

$$\widehat{y_i} = \frac{\sum_{t=1}^{m_i} x_i^{(t)}}{m_i}, \widehat{\mu}_y = \frac{\sum_{i=1}^{n} \widehat{y_i}}{n}$$

”

$$\widehat{\sigma}_y = \sqrt{\frac{\sum_{i=1}^{n} (\widehat{y_i} - \widehat{\mu}_y)^2}{n}}$$

$$\widehat{\theta} = \widehat{\mu}_y + k\widehat{\sigma}_y$$

Where m_i is the number of times i^{th}arm is drawn until that time instant.

CHAPTER 2

ROUND ROBIN ALGORITHM

We would be exploring the first algorithm from (Zhuang *et al.*, 2017) in this chapter.

2.1 Procedure

This simple algorithms samples all active arms one by one in a round robin sense until the termination condition is met. Intuitively the algorithm must stop when all the arms are either classified as outlier or non-outlier. In order to this we construct confidence intervals for estimated means and threshold at every time instant. We classify an arm when it's confidence interval doesn't overlap with threshold's confidence interval. If there is an overlap it's considered active for next iteration.

2.2 Construction of confidence intervals

In order to have the overall algorithm to be $\delta - PAC$ we use δ^1 for Hoeffding bounds.

$$\delta^1 = \frac{6\delta}{\pi^2(n+1)T^2}$$

For a confidence level of δ^1 after T number of total pulls we have

$$P(\widehat{y}_i - y_i > \beta_i(m_i, \delta^1)) < \delta^1, P(\widehat{y}_i - y_i < \beta_i(m_i, \delta^1)) < \delta^1$$

For threshold confidence interval we have

$$P(\widehat{\theta} - \theta > \beta_\theta(\mathbf{m}, \delta^1)) < \delta^1, P(\widehat{\theta} - \theta < \beta_\theta(\mathbf{m}, \delta^1)) < \delta^1$$

where the functions β_i, β_θ are dependent on range of the reward distributions and $\mathbf{m}$ is the vector of arm means. In order to have the overall algorithm to be $\delta - PAC$ we need to set

2.3 Pseudo code

Round Robin algorithm(RR):

Input: n, k

Output: An expected set of outlier arms $\hat{\Omega}$

1. Pull each arm once for i$\epsilon[n]$.
2. Update the values of $\mathbf{m}$,m_θ, β_i, β_θ, $\widehat{\mathrm{y}}_\mathrm{i}$, $\widehat{\theta}$.
3. Construct the set A, the set of active arms.
4. Set i = 1, T = n.
5. While $A \neq \phi$:
6. i = i%n+1
7. Sample arm i
8. T = T+1
9. Update the values of $\mathbf{m}$,m_θ, β_i, β_θ, $\widehat{\mathrm{y}}_\mathrm{i}$, $\widehat{\theta}$
10. Return $\hat{\Omega}$

2.4 Sample complexity

The algorithm can be shown to return the correct set of outliers with probability $1 - \delta$. The sample complexity can be upper bounded from (2) as below:

$$T \leq 8R^2 H_{RR} \left[log(\frac{2R^2\pi^2(n+1)H_{RR}}{3\delta}) + 1 \right] + 4n$$

where $H_{RR} = \frac{n}{min_i(y_i-\theta)^2}(1 + \sqrt{l(k)})^2$, where $l(k) = \Theta(n)$

CHAPTER 3

WEIGHTED ROUND ROBIN ALGORITHM

This algorithm from (Zhuang *et al.*, 2017) is an extension to the regular Round Robin algorithm which is more accurate and takes almost 60% less samples for an identical setting.

3.1 Introduction

In this algorithm instead of sampling all arms equally, we pull the active arms more frequently compared to classified arms. We use a predefined constant ρ and we pull the active arms until they aren't active anymore or the totals pulls exceeds ρ times the number of rounds.

3.2 Ideal value of ρ

Our target is to find the ρ that minimizes the number of samples compared to regular round robin. This value is instance dependent on means of arms. We therefore optimize the following problem

$$\rho^* = argmin_{\rho \geq 1} sup \frac{H_{WRR}}{H_{RR}}$$

Where the supremum is taken over all possible y.

The optimal value of ρ is

$$\rho^* = \frac{(n-1)^{\frac{2}{3}}}{l^{\frac{1}{3}}(k)}$$

CHAPTER 4

BASELINE AlGORITHM

In order to compare our novel algorithms performance we use a strong baseline algorithm CLUCB from (S. Chen and Chen, 2014). The CLUCB algorithm is a pure exploration algorithm which identifies the "best" set of arms in the given decision set in a fixed confidence setting. We modify this algorithm to return best arm. After every iteration of algorithm we remove the best arm from given set and continue until the current best arm is not a part of $\hat{\Omega}$.

In the CLUCB algorithm from (S. Chen and Chen, 2014) we take the decision set as simply the set of individual arms. The maximising Oracle must return the "best" arm at any point. We remove the best arm once the algorithm converges and remove the arm from further sampling. We run the algorithm again on rest of the arms.

4.1 Simulation settings

We tabulate the accuracy values of RR,WRR and compare them with the baseline: $1 - \delta = 0.9$. We don't compare the accuracy of Iterative baseline algorithm as it's not a valid PAC algorithm. It's correctness is also not proved.

Table 4.1: Accuracy are shown for different value of "k" to not make it look cumbersome.

k	RR	WRR
2	0.98	1.0
2.5	0.98	0.97
3	0.96	0.98

For the simulations we use N = 20,30,40,50,75,100,120,150,200. All the means are generated from Uniform distribution [0,1]. For every instance "n" we have averaged the results over 10 different data sets. We have also used 3 "k" values, [2,2.5,3]. The sample

complexity is averaged over 3 "k" values.The plot is a semi log plot with logarithmic axis as Y axis.

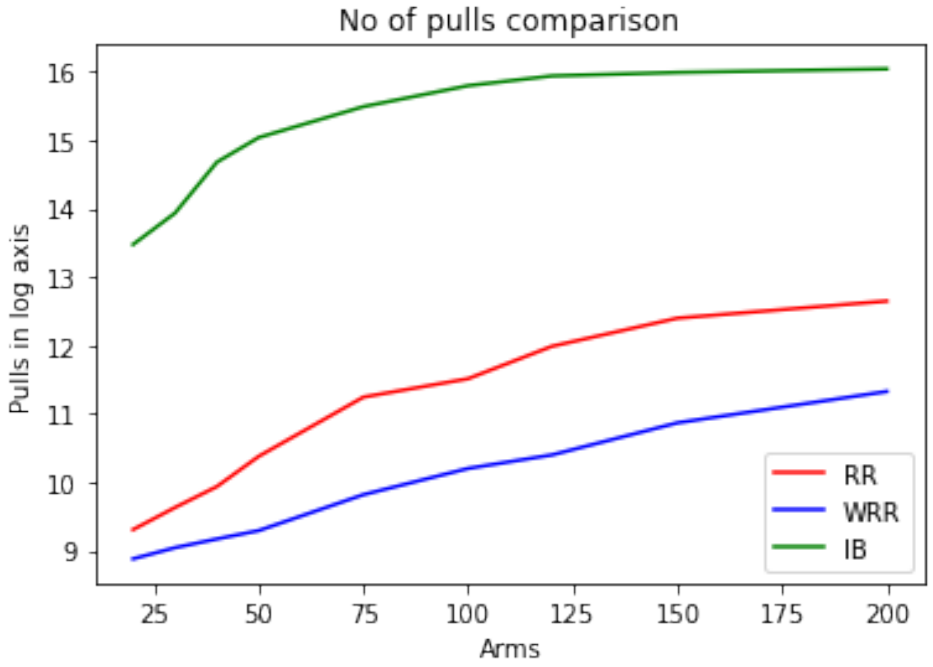


Figure 4.1: Comparison of number of pulls for each of the 3 algorithms RR,WRR,IB

We have verified our expression for weight in WRR for n = 15 arms and k = 2.5. Weights are taken from 1.5 to 5 in steps of 0.5. All the instances repeated for 10 times. From our expression we are supposed to have 2.5 as ideal weight which is exactly the case.

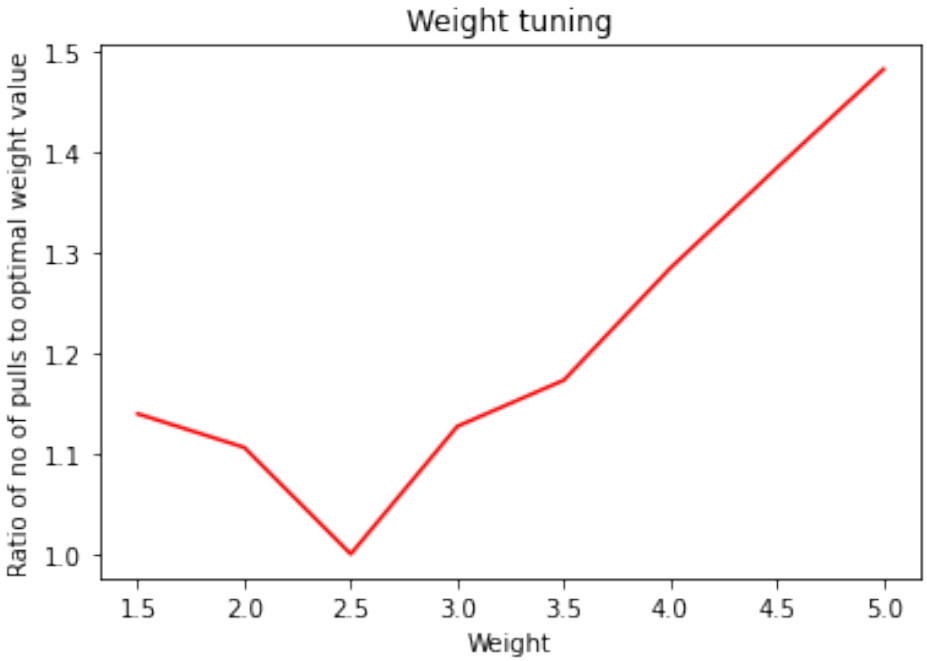


Figure 4.2: Weight tuning in steps of 0.5 to check for weight that minimizes pulls

CHAPTER 5

ADAPTIVE DOUBLE ESTIMATION

5.1 Introduction

The RR and WRR algorithms are state of the art in terms of accuracy and number of samples for a smaller number of arms. The radius of confidence and the sample complexity are heavily dependent on the number of arms. In both the algorithms we keep sampling all the arms until the termination condition even though we have a fair idea which arms are possibly the outliers and which aren't. The radius of confidence of threshold is also dependent on harmonic mean of estimated means of all the arms. We therefore propose ADE from (Zhang and Zhou, 2020) which only samples active arms at any point of time. The sample complexity is also almost independent of number of arms involved. We sample separately to estimate the radius of threshold thus making it independent of estimated means of arms.

5.2 The Algorithm

The algorithm uses two sampling strategies, one for estimating means another for estimating the threshold. We choose strategy based on the radius of confidence. If the arms radius is more then it's more uncertain so we sample arms else we sample for the threshold.

Sequential Sample for The arms: We sample the arm "i" once and the sample is used to update the estimated mean.
Random Sample for The Threshold: We pull an arm uniformly at random from the "n" arms abd we sample it twice. The samples are used to estimate the threshold.

Let $m_{i,t}, y_{i,m_{i,t}}$ be the number of times arm i has been sampled until round $t$and the $m_{i,t}^{th}$ sample respectively.

Let $m_{\theta,t}$be the number of times we sample for threshold before round t. Let the $m_{\theta,t}^{th}$ samples be $(x_{m_{\theta,t},1}, x_{m_{\theta,t},2})$.

We have:

$$\widehat{y}_{i,t} = \frac{1}{m_{i,t}} \sum y_{i,m_{i,t}}$$

$$\widehat{\mu}_{y,t} = \frac{1}{m_{\theta,t}} \sum_{l=1}^{m_{\theta,t}} x_{l,1}$$

$$\widehat{\sigma}_{y,t} = \sqrt{|\frac{1}{m_{\theta,t}} \sum_{l=1}^{m_{\theta,t}} x_{l,1} x_{l,2} - \frac{1}{m_{\theta,t^2}} \sum_{l=1}^{m_{\theta,t}} \sum_{h=1}^{m_{\theta,t}} x_{m_{l,l}} x_{h,2}}$$

$$\widehat{\theta}_y = \widehat{\mu}_{y,t} + k\widehat{\sigma}_{y,t}$$

$$r_{a,t} = R\sqrt{\frac{log(\frac{1}{\delta_t})}{2m_{a,t}}}$$

$$\delta_t = \frac{3\delta}{(n+4)\pi^2 T^2}$$

We can clearly observe the dependence of radius of confidence is clearly limited to logarithmic factors here. Therefore radius of confidence is very small even though we are dealing with high number of arms.

5.3 Sample complexity

The sample complexity for ADE is $\delta - PAC$ with:

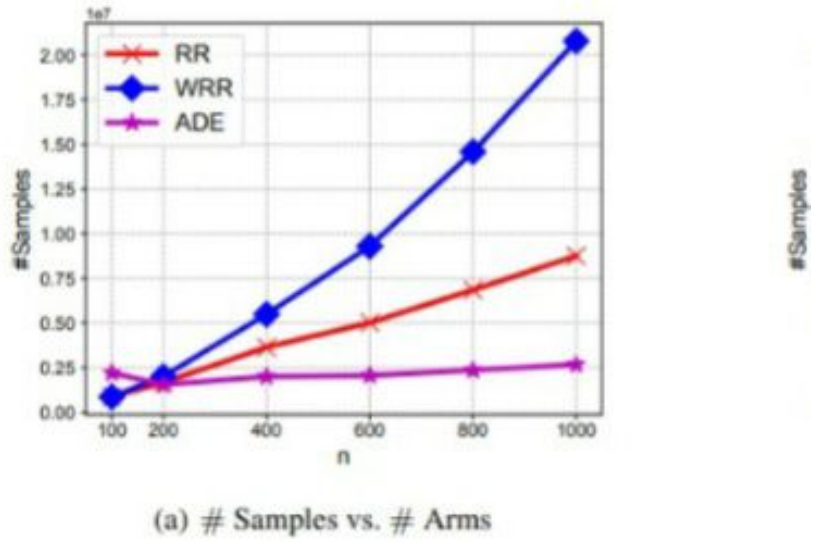
$$T = O(\sum \frac{1}{\Delta_i^2} log(\sqrt{\frac{n}{\delta}} \frac{1}{\Delta_i^2}) + \frac{1}{\Delta_{min}^2} log(\sqrt{\frac{n}{\delta}} \frac{1}{\Delta_{min}^2})$$

where $\Delta_i = |y_i - \theta|$ and $\Delta_{min} = min\Delta_i$

Compared to RR and WRR the sample complexity also only depends on “n” upto logarithmic terms where as both RR and WRR have $H = O(nlogn)$.

5.4 Simulation settings

For the simulation settings we use n = 100,200,300,400,500,600,700,800,900,1000 with each ’n" averaged over 10 different data sets. The means are sampled randomly from a uniform distribution [0,1]. The RR and WRR are taking significantly much time even for single episode. So as a reference we will be using the results of the author with the same settings. In our plot we will be using 1e7 and 2e7 as baselines for comparison.



(a) # Samples vs. # Arms

Figure 5.1: Authors simulation for similar setting

We vary Δ_{min} as [0.06,0.08,0.09,0.1,0.12] to find relationship between Δ_{min} and No of pulls. All the experiments are averaged over 10 episodes.

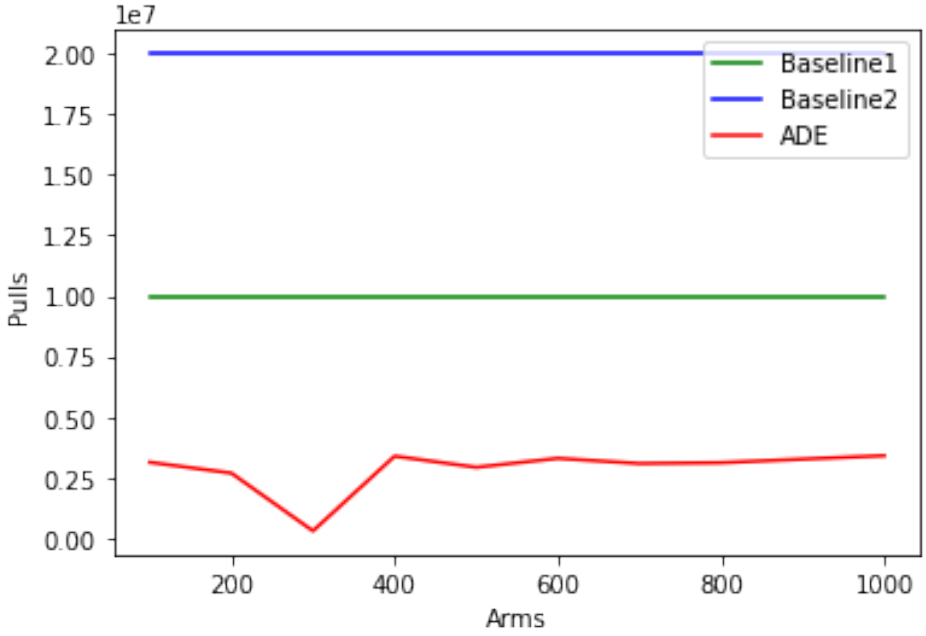


Figure 5.2: Our simulation for ADE

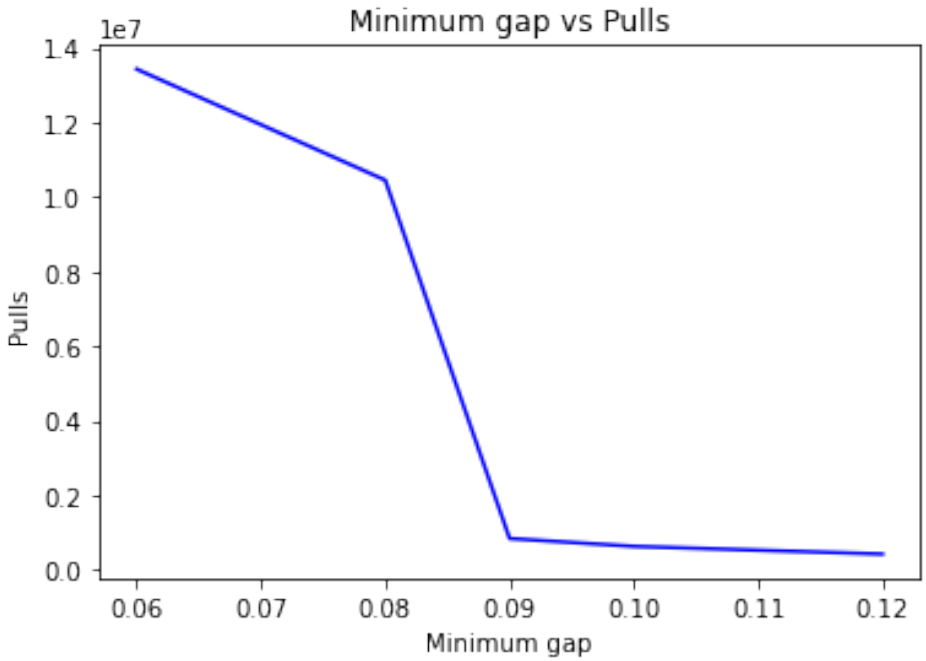


Figure 5.3: Δ_{min} vs Pulls

CHAPTER 6

ROBUST OUTLIER ARM DETECTION

6.1 Introduction

The algorithms from (Yinglun Zhu and Nowak, 2020) are an extension to all the previous algorithms we discussed along with LUCB algorithm from (Shivaram Kalyanakrishnan, 2012). According to extensive work done on robust parameters to detect outliers (Leys and Licata, 2013), turns out Median and Median absolute deviation found the outliers that mean and standard deviation missed. We will be discussing two adaptive algorithms using these for threshold estimation.

6.2 Confidence Intervals

For simplicity purpose we are subscript everything that belongs to median arm with subscript (m). Define $AD(i) = |y_i - y_{(m)}|$. Since we do not know which arm is the median arm, all the confidence intervals of assumed median arm are median of all arms confidence intervals.

$$\beta_s = \sqrt{\frac{log(\frac{4ns^2}{\delta})}{2s}}$$

$$L_{y_{(m)},t} = median\{L_{y_i,t}\}$$

$$U_{y_{(m)},t} = median\{U_{y_i,t}\}$$

$$L_{AD_i,t} = max\{L_{y_{(m)},t} - U_{y_i,t}, L_{y_i,t} - U_{y_{(m)},t}\}$$

$$U_{AD_i,t} = max\{U_{y_{(m)},t} - L_{y_i,t}, U_{y_i,t} - L_{y_{(m)},t}\}$$

$$L_{AD_{(m)},t} = median\{L_{AD_i,t}\}$$

$$U_{AD_{(m)},t} = median\{U_{AD_i,t}\}$$

$$L_{\theta,t} = L_{y_{(m)},t} + kL_{AD_{(m)},t}$$

$$U_{\theta,t} = U_{y_{(m)},t} + kU_{AD_{(m)},t}$$

$$\widehat{\theta}_t = \frac{L_{\theta,t} + U_{\theta,t}}{2}$$

$$\widehat{AD}_{i,t} = \frac{L_{AD_i,t} + U_{AD_i,t}}{2}$$

6.3 ROAElim

The stopping rule for both the algorithms is same. We stop when there is no intersection between threshold confidence interval and any of the arms confidence interval. Now we need to find the arms that overlap with any part of the threshold expression. So we will find arms whose estimated mean interval overlap with interval of median arm's estimated mean. Similarly we find it for absolute deviation of median arm and the threshold itself. We take union of all these sets and find the common arms from the previous union set and sample them all. This looking back at previous set ensures we don't unnecessarily sample arms with minimal doubt and the number of arms sampled at every iteration is non increasing.

6.4 ROALUCB

The LUCB algorithm from (Yinglun Zhu and Nowak, 2020) is special in terms of its sampling strategy. In order to find the "k" best arms we split the set of arms to disjoint sets of size 'm","n-m". We consider the "active" arms as the ones at the boundary of both the sets. We take the arm with highest upper bound the lower empirical average set and the arm with lowest lower bound from the higher empirical average set. We sample both the arms the difference between the bounds is less than some threshold.

Similar to LUCB we sample arms that are on the boundary to median estimated mean, AD of median arm and the threshold. Since we do not clearly know if "m" or "m-1" is the actual median we split the sets accordingly and sample both arms in each case.

$$A_{L,t+1}^{median} = argmin_{i\epsilon J_{m-1}}\{L_{i,t}\} \cup argmin_{i\epsilon J_m}\{L_{i,t}\} \cup argmin_{i\notin J_{m-1}}\{U_{i,t}\} \cup argmin_{i\notin J_m}\{U_{i,t}\}$$

$$A_{L,t+1}^{MAD} = argmin_{i\epsilon J_{m-1}^{MAD}}\{L_{AD_i,t}\} \cup argmin_{i\epsilon J_m^{MAD}}\{L_{AD_i,t}\} \cup argmin_{i\notin J_{m-1}^{MAD}}\{U_{AD_i,t}\} \cup argmin_{i\notin J_m^{MAD}}$$

$$A_{L,t+1}^{\theta} = argmin_{i\epsilon\widehat{S}_{o,t}}\{L_{i,t}\} \cup argmin_{i\epsilon\widehat{S}_{n,t}}\{U_{i,t}\} \cap \{i\epsilon[n]|I_{y_i,t} \cap I_{\theta,t} \neq \Phi\}$$

We will sample the arms from the union sets of these 3 sets. Therefore at every instant we will sampling at most 10 arms similar to LUCB.

6.5 Sample Complexity

$\Delta_i^{\theta} = |y_i - \theta|$, $\Delta_*^{\theta} = min\Delta_i^{\theta}$, $\Delta_i^{median} = |y_i - y_{(m)}|$,$\Delta_i^{MAD} = |AD_i - AD_{(m)}|$

$$\Delta_i^* = max\{\Delta_*^\theta, min\{\Delta_i^\theta, \Delta_i^{median}, \Delta_i^{MAD}\}\}$$

$$T \leq O(Ck^2 \sum_{i=1}^{n} log(\frac{\frac{nk}{\delta\Delta_i^*}}{(\Delta_i^*)^2})$$

Simulation settings For the simulations we take Gaussian system with fixed standard deviation 0.5. We take 17 arms 5 of which have means uniformly distributed between [0,2]. The threshold is fixed as 2.837 with k = 2.95. The two outlier arms have a mean gap of 0.2. They are moved from Threshold such that Δ_*^θ is varied from 0.2 to 0.6 in steps of 0.05. We expect the correlation between the variables to be a straight line as the complexity is tightly bound in both the directions.

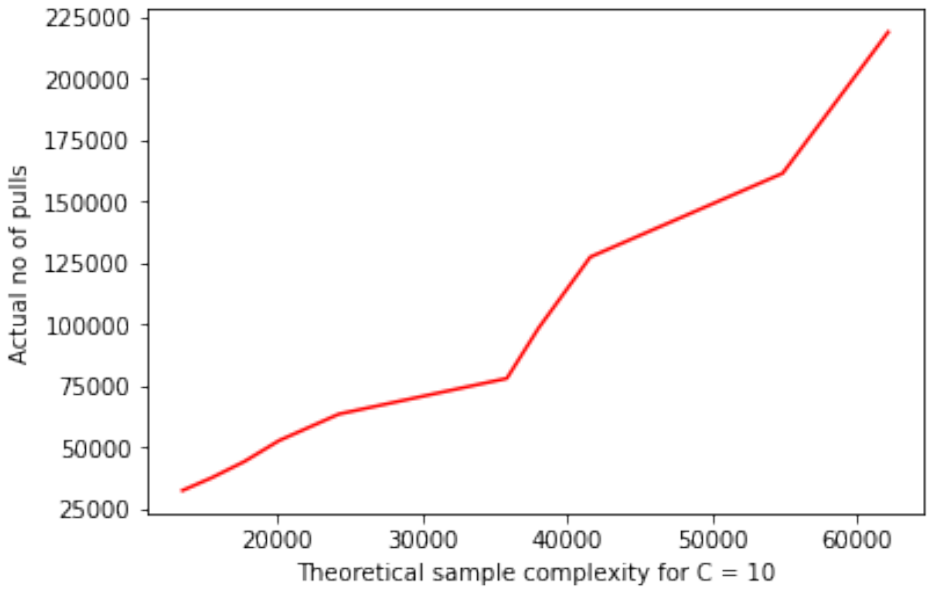


Figure 6.1: Theoretical vs Actual complexity for ROALUCB

CHAPTER 7

GOOD ARM IDENTIFICATION

The best arm identification problem and the threshold problem consume a lot of sample to do a task that is very pinpointed in a sense. Consider the case where we don't have a lot of data to go with, like a case where we have to find a drug out of a pool of drugs that cures a certain rare disease or a classic recommendation system in an e-commerce website. In the second instance a seller who is rated well now might not be rated well in the future hence a decision on whether an arm is "good" should be made rather than finding the best seller or partition all the sellers into good or bad. We will now discuss an elimination style algorithm that no only intends to minimize the stopping time for a threshold problem but also minimizes individual round times of elimination.

7.1 Hybrid dilemma of confidence

The HdoC algorithm from (Kano *et al.*, 2019) consists of two parts, 1. The sampling strategy 2. The identification strategy. The sampling strategy is exactly same as that of UCB. We sample the arm that has maximum UCB score. For the identification we use LUCB algorithm identification rule.

7.2 LUCB-G

In LUCB-G algorithm we use the same metric derived from Hoeffding's inequality for both identification and sampling.

7.3 APT-G

In the APT-G algorithm we use the absolute difference between the estimated mean and the threshold for sampling. In order to not be biased towards arms that are less sampled

we multiply the score function with number of times an arm is pulled. The identification problem is same as LUCB rule.

The proposed lower and upper bounds for sample complexity are very close. The sample complexity for HdoC is $\delta - PAC$ with:

$$E[T_\lambda] \geq \sum_{i=1}^{\lambda}(\frac{1}{d(\mu_i,\xi)}log\frac{1}{2\delta})-\frac{m}{d(\mu_\lambda,\xi)}$$

$$E[T_\lambda] \leq (log\frac{1}{\delta})\sum_{i=1}^{\lambda}(\frac{1}{2\Delta_i^2})$$

where T_λ is the sampling time taken for λ^{th} good arm, Δ_i is the absolute gap from the threshold to the i^{th} arm and $d(\mu_i,\xi)$ is the binary entropy function.

The identification score for all 3 algorithms are:

$$\psi_i = \widehat{\mu}_i \pm \sqrt{\frac{log(\frac{4kN_{a^*}^2}{\delta})}{2N_{a^*}(t)}}$$

if the current sampled arm has $\widehat{\mu}_i - \sqrt{\frac{log(\frac{4kN_{a^*}^2}{\delta})}{2N_{a^*}(t)}} > \xi$ then add it to the set of good arms, else if $\widehat{\mu}_i + \sqrt{\frac{log(\frac{4kN_{a^*}^2}{\delta})}{2N_{a^*}(t)}} < \xi$ then discard the arms the set of current sampled arms.

The sampling scores for different algorithms are:

1. HdoC: $\widehat{\mu}_i + \sqrt{\frac{logt}{2N_i}}$

2. LUCB-G: same as ψ_i

3. APT-G: $min\sqrt{N_i(t)}|\xi - \widehat{\mu}_i|$

7.4 Simulation settings

We have 4 simulation environments here. The first 3 are synthetic and the last one is a real time data set for finding appropriate dosage of a drug used to treat Rheumatoid Arthritis. For all the simulation we use $\delta = 0.05$.

1. The first simulation setting has 10 Bernoulli arms with $\mu_{1:3} = 0.1$, $\mu_{4:7} = 0.35 + (0:3)^{*}0.1$, $\mu_{8:10} = 0.9$ and threshold $\xi = 0.5$.

Table 7.1: Samples for Setting1

$Algo$	τ_1	τ_2	τ_3
$HdoC$	122.06 ± 31.02	173.71 ± 29.92	210.25 ± 28.65
$LUCB-G$	128.33 ± 28.45	178.86 ± 36.70	225.32 ± 34.33
$APT-G$	7084.83 ± 1439.67	7267.07 ± 1456.46	7504.56 ± 1457.22

$Algo$	τ_4	τ_5	$\tau_{stopping}$
$HdoC$	$849.72 + 229.34$	6347.21 ± 1556.90	11689.11 ± 1976.44
$LUCB-G$	880.98 ± 210.42	6521.33 ± 1358.44	11955.21 ± 1863.02
$APT-G$	9338.87 ± 1646.80	11330.79 ± 2016.89	11424.32 ± 2017.65

2. This simulation setting has 6 Bernoulli arms (Arithmetic progression setting) with $\mu_{1:6} = (1:6)^{*}0.1$ and threshold $\xi = 0.35$.

Table 7.2: Samples for Arithmetic setting

τ_1	τ_2	τ_3	τ_{stop}
249.21 ± 47.38	660.23 ± 195.56	6130.20 ± 1272.37	10906.45 ± 1567.98
265.71 ± 48.85	799.86 ± 93.39	5895.32 ± 1355.30	9953.22 ± 1535.22
7778.02 ± 1571.66	8898.98 ± 1592.81	10822.59 ± 2098.20	10910.14 ± 2078.08

3. This simulation setting has 6 Bernoulli arms (Close-to-Threshold setting) with $\mu_{1:3} = 0.55$, $\mu_{4:10} = 0.45$ and threshold $\xi = 0.5$.

Table 7.3: Close-to-Threshold setting

τ_1	τ_2	τ_3	τ_{stop}
10142.22 ± 2976.30	14734.12 ± 2840.56	17890.36 ± 2823.18	51487.37 ± 4697.58
11067.92 ± 2885.99	13910.29 ± 2913.90	17475.21 ± 2600.05	50592.11 ± 5123.67
48321.5 ± 4918.07	49323.93 ± 4907.30	49762.61 ± 4754.758	49818.93 ± 4758.75

4. This simulation setting has 5 Bernoulli arms (Medical data for testing dosage of Secukinumab) with $\overline{\mu} = [0.36, 0.34, 0.469, 0.465, 0.537]$ and threshold $\xi = 0.5$.

Table 7.4: Medical data setting

$Algo$	τ_1	τ_{stop}
$HdoC$	10325.43 ± 3393.25	34838.07 ± 6700.04
$LUCB-G$	10764.77 ± 2195.36	33457.49 ± 5223.42
$APT-G$	33851.35 ± 6534.39	34440.95 ± 6624.07

CHAPTER 8

RBMLE

All the above algorithms except for the Hdoc algorithms we studied are pure exploratory problems. We were only interested in identifying the a set of arms depending on the context and we never really cared about sampled rewards. Now we will see a novel algorithm based on regret minimization that is indexable like the UCB algorithm. This algorithm takes advantage of family of bandits and gives us a separate indexable function for every family. It is faster than all the state of the art MAB algorithms.

8.1 Introduction

We pose the Multi armed bandit problem as an Markov Decision Process with an unknown vector of parameters on which the transition probabilities depend on every step. The Certainity Equivalence approach goes by taking the MLE estimate of Likelihood function as the parameter value at every state and follows the optimal policy for that parameter set. This has been shown to be sub-optimal in terms of Long time reward. The paper uses "Reward biased Maximum likelihood estimation" technique to overcome the problem.

Consider an MDP with state space Ω and action space Θ..Assume the transition probabilities are stationary and depend on parameter vector η. We have $p(i, j, u; \eta)$ as the transition probability from state i to state j under action u with parameter η. The reward for this transition be $r(i, j, u)$.

Let η^0 be the true parameter and let $J(\phi, \eta)$ be the reward for policy ϕ. We consider optimal reward for every possible η as $J_{opt}(\eta) = max_\phi J(\phi, \eta)$. For the CE approach we have

$$\eta_t^* = max_\eta \prod_{t=0}^{T} p(s(t), s(t+1), u(t), \eta)$$

The action taken in CE approach is $u_t = \phi_{\eta_t^*}(s(t))$.Both the parameter estimate and

the long term regret estimated by CE process are non optimal. But in the long term the probability transitions are optimal. i.e

$$p(i, j, \phi_{\eta^*(t)}(i), \eta_t^*) = p(i, j, \phi_{\eta^*(t)}(i), \eta^0)$$

$$\eta^{RBLM} = argmax_\eta [f(J_{opt}(\eta))^{\alpha(t)} \prod_{t=0}^{T} p(s(t), s(t+1), u(t), \eta)]$$

where f is a strictly monotonically increasing function and $\alpha(t)$ is a process which is $O(t)$.

The indexing function for Gaussian and Sub Gaussian family is:.

$$I_i(t) = p_i(t) + \frac{\alpha(t)}{2N_i(t)}$$

where $p_i(t), N_i(t)$ are the estimated mean and number of times i^{th}arm is sampled until round t.

$$\alpha(t) = C(t)\sqrt{logt}$$

For Gaussian distributions: Assume same variance for all arms, and $C \geq \frac{256\sigma^2}{\Delta}$, we have

$$R(t) \leq O(\sum_{a=2}^{n} \Delta_a) + O(nClogt)$$

We need to have a minimum value of "C" for our regret to be bounded. Therefore we adaptively estimate the constant every iteration.

8.2 Simulation settings

We use a 10 arm Gaussian bandit family with variance 1. The set of means are equal to those one taken in (Mete *et al.*, 2021). For benchmarking we used UCB, Thompson, UCB-normal and MOSS algorithms. We have plotted the cumulative regret until 1e5. We have also measured the time taken for a decision in every algorithm

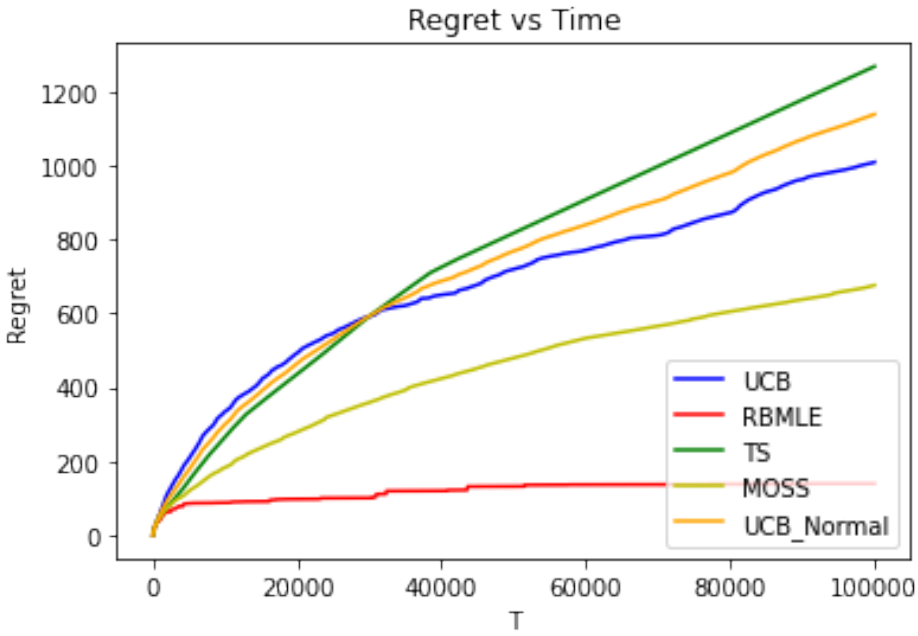


Figure 8.1: Cumulative regret for RBMLE

Table 8.1: Time taken per decision

$Algo$	$Time/decision$
$RBMLE$	0.001266
UCB	0.00038062
TS	0.0008261
$MOSS$	0.0009708
$UCB-Norm$	0.001008

CHAPTER 9

GAI with RBMLE as score function for sampling

9.1 Introduction

In place of UCB score as mentioned in (Kano *et al.*, 2019), I have tried using RBMLE score for sampling as RBMLE has less regret therefore more possibility to exploit. Like mentioned in the RBMLE section we need to maintain a minimum "C" to have our regret bounded. The "C" is estimated step wise using estimated mean at that particular instant. We do not use the "classified" arms for estimating the "C" value. The identification score has to be updated for Gaussian distribution by using Cramer's bound as mentioned in (Kano *et al.*, 2019).

9.2 Simulation settings

4. This simulation setting has 10 Gaussian arms with mean values identical to setting 1 in GAI simulations and threshold $\xi = 0.5$. We take variance $\sigma = 1$ for all the amrs.

Table 9.1: HdoC vs HdoC RBMLE

$Algo$	τ_1	τ_2	τ_3
$HdoC$	405.43 ± 152.94	650.59 ± 171.32	948.88 ± 286.136
$HdoC - RBMLE$	372.52 ± 132.44	611.32 ± 154.59	875.45 ± 220.12

$Algo$	τ_4	τ_5	τ_{stop}
$HdoC$	4002.12 ± 1672.09	26716.55 ± 7918.30	47234.22 ± 10280.42
$HdoC - RBMLE$	3872.51 ± 1378.53	24234.63 ± 7683.22	43598.23 ± 10092.87

CHAPTER 10

CONCLUSIONS AND FUTURE WORK

The RR and WRR algorithms are simple but they keep sampling arms unnecessarily even though we know with confidence in the middle if some of the arms are outliers. The sample complexity also increases drastically with the number of arms which is not desirable for recommendation systems considering there are hundreds of items to be suggested. The adaptive double estimation samples for threshold separately so that we don't have to depend on estimated means for threshold estimate. This has diminished the effect of number of arms on sample complexity. But ADE still depends on mean and standard deviation to define an outlier and is limited to bandit families with finite range reward. ROALim, ROALUCB have stable measures to identify outliers and have performed sufficiently well compared to all other state of the art algorithms. GAI has shown why it's better than outlier detection problem in the real world. It needs very less samples to output a subset of outlier arms which is satisfactory for several problems including Drug testings and crowd sourcing algorithms.

As part of the future work to our project we can explore using the ADE algorithm to infinite range families (e.g Gaussian). Though the RBMLE-HdoC performed better than regular HdoC algorithm we need to mathematically prove that the algorithm is $\delta - PAC.$ and we can obtain bounds on sample complexity. The bounds given for ROALim, ROALUCB are not tight and are difficult to be obtained with the regular techniques. We can try to improve these bounds mathematically.

REFERENCES

1. **Carpentier, A.** and **M. Valko**, Extreme bandits. NIPS, 2014.

2. **Kano, H.**, **J. Honda**, **K. Sakamaki**, **K. Matsuura**, **A. Nakamura**, and **M. Sugiyama** (2019). Good arm identification via bandit feedback. *Mach. Learn.*, **108**(5), 721–745. ISSN 0885-6125. URL `https://doi.org/10.1007/s10994-019-05784-4`.

3. **Leys, L. C. K. O. B. P., C.** and **L. Licata** (2013). Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, **49**(4), 764–766.

4. **Mete, A.**, **R. Singh**, **X. Liu**, and **P. R. Kumar** (2021). Reward biased maximum likelihood estimation for reinforcement learning.

5. **S. Chen, I. K. M. R. L., T. Lin** and **W. Chen**, Combinatorial pure exploration of multi-armed bandits. In NIPS, 2014.

6. **Shivaram Kalyanakrishnan, P. A. P. S., Ambuj Tewari**, Pac subset selection in stochastic multi-armed bandits. Proceedings of the 29thInternational Confer-ence on Machine Learning, Edinburgh, Scotland, UK, 2012.

7. **Yinglun Zhu, S. K.** and **R. Nowak**, Robust outlier arm identification. Proceedings of the 37th International Conference on MachineLearning, Online, PMLR 119, 2020.

8. **Zhang, Z. H. Z. S., X.** and **Y. Zhou**, Adaptive double-exploration tradeoff for outlier detection. The Thirty-Fourth AAAI conference, 2020.

9. **Zhuang, H.**, **C. Wang**, and **Y. Wang**, Identifying outlier arms in multi-armed bandit. In Advances in Neural Information Processing Systems, 2017.