Thresholding Bandit Problem with Both Duels and Pulls

Yichong Xu[1], **Xi Chen**[2] **Aarti Singh**[1], **Artur Dubrawski**[1]
[1]Machine Learning Department, Carnegie Mellon University [2]Stern School of Business, New York University

Abstract

The Thresholding Bandit Problem (TBP) aims to find the set of arms with mean rewards greater than a given threshold. We consider a new setting of TBP, where in addition to pulling arms, one can also *duel* two arms and get the arm with a greater mean. In our motivating application from crowdsourcing, dueling two arms can be more cost-effective and time-efficient than direct pulls. We refer to this problem as TBP with Dueling Choices (TBP-DC). This paper provides an algorithm called Rank-Search (RS) for solving TBP-DC by alternating between ranking and binary search. We prove theoretical guarantees for RS, and also give lower bounds to show the optimality of it. Experiments show that RS outperforms previous baseline algorithms that only use pulls or duels.

1 Introduction

The Thresholding Bandit Problem (TBP, (Locatelli et al., 2016)) is an important pure-exploration multi-armed bandit (MAB) problem. Specifically, given a set of K arms with different mean rewards, the TBP aims to find arms whose mean rewards are above a pre-set threshold of τ. The TBP has a wide range of applications, such as anomaly detection, candidate filtering, and crowdsourced classification. For example, a popular crowdsourced classification model (Abbasi-Yadkori et al., 2016; Chen et al., 2015) assumes that there are K items with the latent true labels $\theta_i \in \{0, 1\}$ for each item. The labeling difficulty of the i-th item is characterized by its soft label $\mu_i \in [0, 1]$, which is defined as the probability that a random crowd worker will label the i-th item as positive. It is clear that the item is easy to label when μ_i is close to 0 or 1, and difficult when μ_i is close to 0.5. In MAB, μ_i is the mean reward of arm i, and pulling this arm leads to a Bernoulli observation with mean μ_i. Moreover, it is natural to assume that the soft label μ_i is consistent with the true label, i.e., $\mu_i \geq 0.5$ if and only if $\theta_i = 1$. Therefore, identifying items belonging to class 1 is equivalent to detecting those arms with $\mu_i > \tau$ with $\tau = 0.5$.

Existing literature on TBP considers the setting that only solicits information from pulling arms directly. However, in many applications of TBP, comparisons/duels can be obtained at a much lower cost than direct pulls. In crowdsourcing, a worker often compares two items more quickly and accurately than labeling them separately. It will be cheaper and time efficient to ask a worker which image is more relevant to a query as compared to asking for an absolute relevance score of an image (see, e.g., Shah et al. (2016b)). Another example is in material synthesis, a pull will need an expensive synthesis of the material, whereas duels can be carried out easily by querying experts. In such settings, directly pulling an arm is expensive and could incur a large sample complexity since each arm needs to be pulled a number of times. This paper considers two sources of information: in addition to direct pulls of arms as in the classical TBP, one can also *duel* two arms to find out the arm with a greater mean at a lower cost. We refer to this problem as the TBP with Dueling Choices (TBP-DC), since dueling and pulling are both available in each round.

It is important to note that some direct pulls are still necessary for solving a TBP even if one can duel two arms. Without direct assessments of arms, we can at best rank all the arms with duels. However, we then cannot know the target threshold τ and therefore cannot identify a boundary on the ranking. On the other hand, using an appropriate dueling strategy, the number of required direct pulls can be much lower than that in the classical TBP setting, where only direct pulling is available. We further note that TBP-DC is also different from the top-K arm identification problem considered in whether MAB (see, e.g., Bubeck et al. (2013), Zhou et al. (2014), Chen et al. (2017)) or dueling bandits (see, e.g., Mohajer et al. (2017)), because the number of arms with means greater than the threshold τ is unknown to us.

Proceedings of the 23[rd]International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

A straightforward way to solve the TBP-DC problem is to utilize an existing ranking algorithm such as ActiveRank (Heckel et al., 2016) to rank all the arms, and then use a binary search to find the boundary. However, this method is impractical because it can be very hard to differentiate arms with similar means (e.g., equally good images, similar quality materials). These arms might be far from the threshold and it is actually unnecessary to differentiate them. We instead take an iterative approach; We develop the Rank-Search (RS) algorithm for TBP-DC, which alternates between refining the rank over all items using duels and a binary search process using pulls to figure out the threshold among ranked items. We interleave the ranking and searching step so that we do not waste time differentiating equally good arms.

Our contributions. First, in Section 3, we analyze the number of duels and pulls required for RS under the fixed confidence setting, i.e., to recognize the set of arms with reward larger than τ with probability at least $1-\delta$. To better illustrate our main idea, we further provide concrete examples, which show that the proposed RS only requires $O(\log^2 K)$ direct labels, while the classical TBP requires at least $\Omega(K)$ labels (see Section 4). Section 5 shows complementary lower bounds that RS is near-optimal in both duel and pull complexity. Finally, we provide practical experiments to demonstrate the performance of RS.

Related Works. TBP is a special case of the pure-exploration combinatorial MAB problem. As with other pure-exploration MAB problems(Bubeck et al., 2013) , algorithms for combinatorial bandits fall into either *fixed-budget* or *fixed-confidence* categories. In the former setting, the algorithm is given a time horizon of T and tries to minimize the probability of failure. In the latter setting, the algorithm is given a target failure probability and tries to minimize the number of queries. For TBP, the CLUCB algorithm (Chen et al., 2014) can solve TBP under the pull-only and fixed confidence setting, with optimal sample complexity. (Chen et al., 2014) also develops the CSAR algorithm for the fixed-budget setting which can also be used for TBP. The result was improved by recent followup work (Locatelli et al., 2016; Mukherjee et al., 2017) under the fixed budget setting. Chen et al. (2015) considered TBP in the context of budget allocation for crowdsourced classification in the Bayesian framework.

Motivated by crowdsourcing and other applications, this paper proposes a new setup since we allow both pulling one arm and dueling two arms in each round, with the underlying assumption that dueling is more cost-effective than pulling. To the best of our knowledge, this setting has not been considered in the previous work. Most close in spirit to our work is a series of recent papers (Kane et al., 2017; Xu et al., 2018, 2017), which consider using pairwise comparisons for learning classifiers. The methods in those papers are however not directly applicable to TBP-DC because their final goal is to learn a classification boundary, instead of labeling each item without feature information.

2 Problem Setup

Suppose there are K arms, which are denoted by $\mathcal{A} = [K] = \{1, 2, ..., K\}$. Each arm $i \in \mathcal{A}$ is associated with a mean reward μ_i. Without loss of generality, we will assume that $\mu_1 \leq \mu_2 \leq \cdots \leq \mu_K$. Given a target threshold τ, our goal is to identify the positive set $S_\tau = \{i : \mu_i \geq \tau\}$ and the negative set $S_\tau^c = \{i : \mu_i < \tau\}$.

Modes of interactions. Each instance of TBP-DC is uniquely defined by the tuple $(M, \boldsymbol{\mu})$, where M is the preference matrix (defined below) and $\boldsymbol{\mu} = \{\mu_i\}_{i=1}^K$ is the mean reward vector. In each round of our algorithm, we can choose one of two possible interactions:

- **Direct Queries (Pulls)**: We choose an arm $i \in \mathcal{A}$ and get a (independent) noisy reward Y from arm i. We assume that each arm i is associated a reward distribution ν_i with mean μ_i, and that ν_i is sub-Gaussian with parameter R: $\mathbb{E}_{Y\sim\nu_i}[\exp(tY - t\mathbb{E}[Y])] \leq \exp(R^2t^2/2)$ for all $t \in \mathbb{R}$. The definition of sub-Gaussian variables includes many common distributions, such as Gaussian distributions or any bounded distributions (e.g., Bernoulli distribution). We denote by $\Delta_i^l = |\mu_i - \tau|$ the gap between arm i and the threshold.

- **Comparisons (Duels)**: We can also choose to duel two arms $i, j \in \mathcal{A}$ and obtain a random variable Z, with $Z = 1$ indicating the arm i has a larger mean reward than j and $Z = 0$ otherwise. Let $M_{ij} \in [0, 1]$ characterize the probability that a random worker believes that arm i is "more positive" than arm j. The outcome of duels is therefore characterized by the matrix M. The (Borda) score of each arm in dueling is defined as

$$p_i := \frac{1}{K-1} \sum_{j\in[K]\setminus\{i\}} M_{ij}, \tag{1}$$

  i.e., the probability of arm i beating another randomly chosen arm j.

  In contrast to previous work (Shah et al., 2016b; Szörényi et al., 2015; Yue et al., 2012) that usually assumes parametric or structural assumptions on M, we allow an arbitrary preference matrix M; the only assumption is that the score of any positive arm is larger than any negative arm, i.e., $p_i > p_j, \forall i \in S_\tau, j \in S_\tau^c$. We note that this is a very weak condition since arbitrary relations within

the positive and negative sets are allowed. This assumption also holds if $(1, 2, ..., K)$ is the Borda ranking of M; or the underlying comparison model follows the Strong Stochastic Transitivity (SST, (Fishburn, 1973; Shah et al., 2016a)). We note that the problem is very difficult under this assumption: For example, even if μ_i (knowledge from pulls) are bounded away from τ by a constant, the p_i (knowledge from duels) may be arbitrarily close, hence making the problem much harder.

Taking crowdsourced binary classification as an example, $Y_i \in \{0, 1\}$ would correspond to a binary label of the i-th item obtained from a worker, where $\mu_i = \Pr_{Y_i \sim \nu_i}[Y = 1]$. For this case we have $\tau = 1/2$. Dueling outcome Z_{ij} will correspond to asking a worker to compare item i with item j and $Z_{ij} = 1$ if the worker claims that item i is "more positive" than item j.

The fixed-confidence setting. Given a target error rate δ, our goal is to recover the sets $\hat{S}_\tau$ and $\hat{S}_\tau^c$, such that $\Pr[S_\tau = \hat{S}_\tau, S_\tau^c = \hat{S}_\tau^c] \geq 1 - \delta$, with as fewer pulls and duels as possible. Since in practice duels are often cheaper than pulls, we want to minimize the number of pulls while also avoiding too many duels.

2.1 Problem Complexity

We define two problem complexities w.r.t pulls and duels separately.

Pull complexity. Following previous works on TBP and pure-exploration bandits (Chen et al., 2014; Locatelli et al., 2016), we introduce the following quantity to characterize the intrinsic problem complexity with direct pulls. In particular, recall that $\Delta_i^l = |\mu_i - \tau|$ is the gap between arm i and threshold. Then the pull complexity is defined as $H_l = \sum_{i=1}^K \frac{1}{(\Delta_i^l)^2}$. Chen et al. (Chen et al., 2014) shows that there exists an algorithm using at most $O(H_l \log(KH_l/\delta))$ pulls. Moreover, they show a lower bound that any pull-only algorithm would require at least $\Omega(H_l \log(1/\delta))$ pulls to give correct output with probability $1 - \delta$. We add another notation for a "partial" label complexity: let $H_l(m)$ be the sum of the largest m terms in H_l. Namely, we sort $\mu_1, \ldots, \mu_K$ by their gap with threshold, i.e., $\Delta_{i_1}^l \leq \Delta_{i_2}^l \leq \cdots \leq \Delta_{i_K}^l$ (cf. Figure 1 left), and $H_l(m) = \sum_{j=1}^m \frac{1}{(\Delta_{i_j}^l)^2}$.

Duel complexity. Now we define the complexity w.r.t. duels. Our goal is to use duels to infer the (positive or negative) label of arms without actually pulling them. Therefore the difficulty of inferring a positive arm $i \in S_\tau$ will depend on its difference with the "worst" positive arm, and similarly $i \in S_\tau^c$ with the "best" negative arm. Let $i_l = \arg\max_{i \in S_\tau^c} p_i$ be the best negative arm and $i_u = \arg\min_{i \in S_\tau} p_i$ be the worst positive arm, where p_i is defined in Equation (1). And for any arm $i \in S_\tau$, let $\Delta_i^c = p_i - p_{i_u}$ be the gap with arm i_u and similarly for any arm $j \in S_\tau^c$ define $\Delta_j^c = p_{i_l} - p_j$. Intuitively, the complexity of identifying arm i through duels should depend on Δ_i^c, and we therefore define $H_{c,1} = \sum_{i=1}^K \frac{1}{(\Delta_i^c)^2}$.

Moreover, it is worthwhile noting that the complexity of inferring a positive arm i using arm i_u will not only depend on $p_i - p_{i_u}$, but also on $p_{i_u} - p_{i_l}$. If the gap $p_{i_u} - p_{i_l}$ is very small, we cannot easily differentiate i_u from the other negative arms. On the other hand, we can use any positive arm j to infer about arm i, when $p_{i_u} \leq p_j < p_i$. To this end, we define

$$\bar{\Delta}_i^c = \begin{cases} \max_{j \in S_\tau} \min\{p_j - p_{i_l}, p_i - p_j\} & \text{if } i \in S_\tau, \\ \max_{j \in S_\tau^c} \min\{p_j - p_i, p_{i_u} - p_j\}, & \text{if } i \in S_\tau^c, \end{cases}$$

See Figure 1 right for a reference. And we define another duel complexity as $H_{c,2} = \sum_{i \in \mathcal{A} \setminus \{i_u, i_l\}} \frac{1}{\left(\bar{\Delta}_i^c\right)^2}$.

Relation between Δ_i^c and $\bar{\Delta}_i^c$. Although we always have $\Delta_i^c \geq \bar{\Delta}_i^c$ and thus $H_{c,1} \leq H_{c,2}$, in many situations Δ_i^c and $\bar{\Delta}_i^c$ are of similar scales. To see this, notice that $\bar{\Delta}_i^c \geq \min\{\Delta_i^c, p_{i_u} - p_{i_l}\}$. In many cases in practice, we would expect a gap between S_τ and S_τ^c, and therefore $p_{i_u} - p_{i_l}$ will be a constant. We give a formal proposition about the relation between $H_{c,2}$ and $H_{c,1}$ under Massart noise condition in Section 4.

In Section 3, we present an upper bound using $H_{c,2}$, and in Section 5, we present a lower bound using $H_{c,1}$.

3 The Rank-Search (RS) Algorithm

We present our Rank-Search algorithm in this section. We give a detailed description of the algorithm in Section 3.1, and analyze its theoretical performance in Section 3.2.

3.1 Algorithm Description

Algorithm 1 describes the Rank-Search algorithm. At a high level, RS alternates between ranking items using duels (Line 3-13), and a binary search using pulls (Line 14 and Algorithm 2). We first initialize the work set S with all arms, and comparison confidence $\gamma_0 = 1/4$. In the rank phase, we iteratively compare each arm $i \in S$ with a random arm, as an unbiased estimator for p_i. After each arm has received $\frac{\log(2/\delta_t)}{\gamma_t^2}$ comparisons, we rank the arms in S according to their win rates $\hat{p}_i$. Then Algorithm 2 performs binary search on the sorted S to find the boundary between positive and negative arms (detailed below).

Our binary search is a standard process: it starts with the middle of the sequence, and if the middle arm is positive, we move to the first half (i.e., arms with

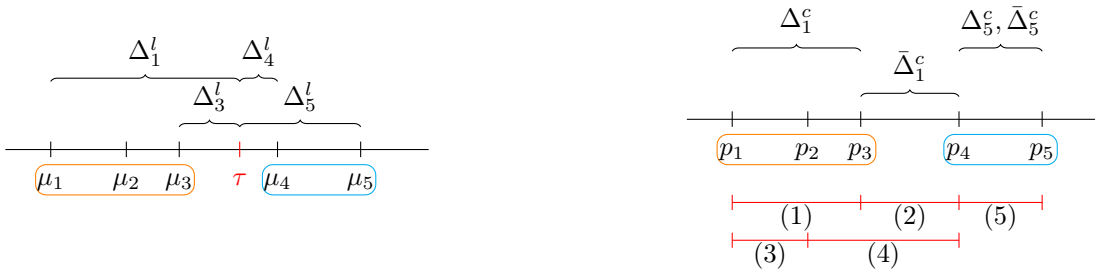


Figure 1: Graphical illustration of the quantities Δ_i^l (left) and $\Delta_i^c, \bar{\Delta}_i^c$ (right) for $K = 5$ arms, with $S_\tau = \{4, 5\}$. We have $i_u = 4$ and $i_l = 3$. $\bar{\Delta}_1^c$ is equal to the max of $\min\{(1), (2)\}$ and $\min\{(3), (4)\}$; $\bar{\Delta}_5^c$ is equal to $\min\{(2), (5)\}$.

Algorithm 1 Rank-Search (RS)

Input: Set of arms $\mathcal{A}$, noise tolerance δ, threshold τ, initial confidence level γ_0, shrinking factor κ

$S \leftarrow \mathcal{A} = [K]$, counter $t \leftarrow 0$
For every $i \in S$, let $n_i \leftarrow 0, w_i \leftarrow 0$
▷ n_i: Comparison count, w_i: Win count
while $S \neq \emptyset$ **do**
  while $\exists i \in S, n_i \leq \frac{1}{\gamma_t^2} \log\left(\frac{8|S|(t+1)^2}{\delta}\right)$ **do**
    for $i \in S$ **do**
      Draw $i' \in [K]$ uniformly at random, and compare arm i with arm i'
      If arm i wins, $w_i \leftarrow w_i + 1$
      $n_i \leftarrow n_i + 1$
    end for
  end while
  Compute $\hat{p}_i \leftarrow w_i/n_i$ for all $i \in S$
  Rank arms in S according to their $\hat{p}_i$: $S = (i_1, i_2, ..., i_{|S|}), \hat{p}_{i_1} \leq \hat{p}_{i_2} \leq \cdots \leq \hat{p}_{i_{|S|}}$
  Get $(k, T) =$ Binary-Search$(S, \tau, \delta/4(t+1)^2)$
  If $k < |S|$, let $\overline{S} = \{i \in S : \hat{p}_i - \hat{p}_{i_{k+1}} > 2\gamma_t\}$; for $i \in \overline{S}$, set $\hat{y}_i = 1$
  If $k > 0$, let $\underline{S} = \{i \in S : \hat{p}_i - \hat{p}_{i_k} < -2\gamma_t\}$; for $i \in \underline{S}$, set $\hat{y}_i = 0$
  $S \leftarrow S - \overline{S} - \underline{S} - T$
  $\gamma_{t+1} \leftarrow \gamma_t/\kappa$
  $t \leftarrow t + 1$
end while

Output: $\hat{S}_\tau = \{i : \hat{y}_i = 1\}, \hat{S}_\tau^c = \mathcal{A} \setminus \hat{S}_\tau$

smaller estimated means), and otherwise, we move to the second half (i.e., arms with larger estimated means). Algorithm 2 just behaves as if S is perfectly ranked. It is worthwhile noting that since S is not ranked according to the real p_i's, there might be negative samples larger than positive samples in S. However, we show that RS can still run effectively even with a misranked S. We figure out the label of the middle point using Figure-Out-Label (Algorithm 3). Figure-Out-Label aims to solve the simple TBP in the one-arm setting: We keep a confidence interval $\hat{\mu}_i \pm \gamma$ in each round and return the label once τ is not in the interval.

Binary-Search returns the boundary k. Let $\overline{S} = \{i \in S : \hat{p}_i - \hat{p}_{i_{k+1}} > 2\gamma_t\}$ be the arms that are separated from arm i_{k+1}, and we label $i \in \overline{S}$ as positive; we do similarly for negative arms. Then we update working set S with all the unlabeled arms, and we shrink the confidence level by a constant factor $\kappa > 1$.

Algorithm 2 Binary-Search

Input: Sequence $S = (i_1, i_2, ..., i_{|S|})$, threshold τ, confidence δ_0

$k_{\min} \leftarrow 0, k_{\max} \leftarrow |S|, T = \emptyset$
while $k_{\min} < k_{\max}$ **do**
  $k = \lceil (k_{\min} + k_{\max})/2 \rceil$
  $\hat{y}_{i_k} =$ Figure-Out-Label$(i_k, \tau, \frac{\delta}{\log |S|})$
  $T = T \cup \{i_k\}$
  if $\hat{y}_{i_k} = 1$ **then**
    $k_{\max} = k - 1$
  else
    $k_{\min} = k$
  end if
end while

Output: Boundary $k_{\min}$, labeled arms T

Algorithm 3 Figure-Out-Label

Input: Arm i, threshold τ, confidence δ_1

$t \leftarrow 0$
Define $m_i \leftarrow 0, s_i \leftarrow 0$
repeat
  while $m_i \leq 2^t$ **do**
    Query Y_i, and let $s_i \leftarrow s_i + Y_i, m_i \leftarrow m_i + 1$
  end while
  Compute $\hat{\mu}_i \leftarrow s_i/m_i$
  $\gamma = R\sqrt{\frac{2\log(4(t+1)^2/\delta_1)}{m_i}}$
  $t \leftarrow t + 1$
until $|\hat{\mu}_i - \tau| > \gamma$

Output: Predicted label $\hat{y}_i = I(\hat{\mu}_i > \tau)$

3.2 Theoretical Analysis

We now present the theorem about performance of RS.

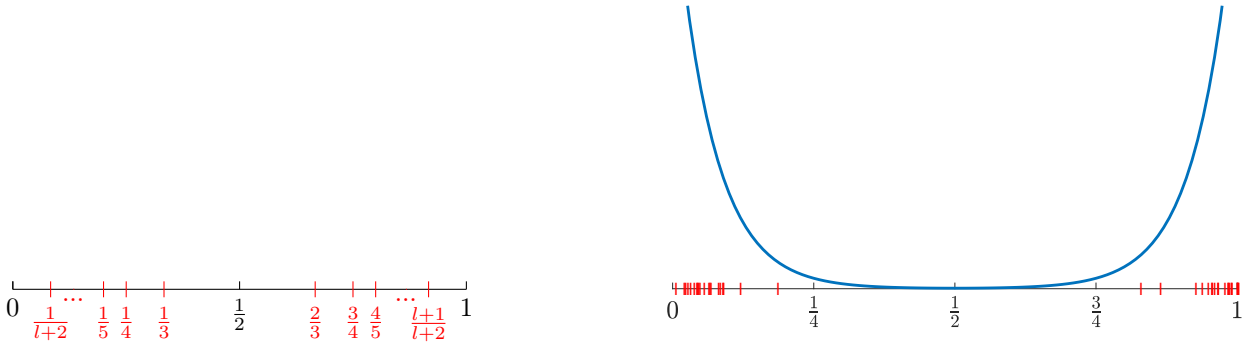


Figure 2: Graphical illustration of the examples. Each red vertical line corresponds to one arm i, and $\tau = 1/2$. Left: Example 1 with fixed means. Right: Example 2 with $K = 40$ arms. The blue curve illustrates the pdf of all arm means.

Theorem 1. *Let* $\gamma^* = \min_{i\in\mathcal{A}\setminus\{i_u,i_l\}} \bar{\Delta}_i^c$ *and* $\Delta^* = \min_i \Delta_i^l$. *Then with probability* $1-\delta$ RS *succeeds, and the number of duels and pulls it uses are bounded by*

$$n_{duel} \le 32 H_{c,2} \log \frac{4K \log(1/\gamma^*)}{\delta},$$
$$n_{pull} \le 16 R^2 H_l(n_l) \log\left(\frac{n_l \log(1/\Delta^*)}{\delta}\right),$$

where n_l *is the number of times* Figure-Out-Label *is called, and we have* $n_l = O(\log K \log(1/\gamma^*))$.

Remark. First, the results in (Chen et al., 2014) correspond to using $O\left(H_l(K)\log\left(\frac{H_l(K)K}{\delta}\right)\right)$ pulls to get δ confidence. In terms of number of direct pulls, RS can reduce K dependence to $\log K$ dependence when γ^* is a constant, an exponential improvement.
Second, in terms of number of duels, RS has a requirement based on dueling complexity $H_{c,2}$ instead of H_l. In many cases, $H_{c,2}$ is close to H_l, and we point out several such cases in Section 4. Thus, we see that in the Dueling-choice framework, the number of pulls required improves exponentially in dependence on K at the expense of requiring a number of duels proportional to number of pulls in pull-only case.

4 Implications of Upper Bounds in Special Cases

We provide two examples to compare our theoretical upper bounds with the classical pull-only TBP. Throughout this section, we will assume that all the observations follow Bernoulli distributions, and $\tau = 1/2$. The examples we raise in this section all follow the Massart noise condition, i.e., $|\mu_i - \tau| \ge c$ that is well known in classification analysis (Massart and Nédélec, 2007). We first give the following proposition to show that RS is optimal under Massart noise.

Proposition 2. *Suppose* $\Delta_i^l \ge c$ *for some* c *for all arm* i, *and* $M_{ij} = \frac{1}{2} + \sigma(\mu_i - \mu_j)$ *for some increasing link function* $\sigma : \mathbb{R} \to [-1/2, 1/2]$. *Also assume for any* $x, y \in [\mu_1, \mu_K]$ *we have* $\sigma(x-y) \ge L(x-y)$ *for some constant* L. *Then we have i)* $p_{i_u} - p_{i_l} \ge 2Lc$, *ii)* $\bar{\Delta}_i^c \ge \min\{2Lc, \Delta_i^c\}$, *and iii)* $H_{c,2} \le \frac{1}{4L^2c^2} H_{c,1}$.

Proposition 2 shows that $H_{c,2} = O(H_{c,1})$ under Massart noise and the assumption that a link function exists. The assumption of such a link function is satisfied by several popular comparison models including the Bradley-Terry-Luce (BTL) (Bradley and Terry, 1952) and Thurstone models (Thurstone, 1927). We now give two positive examples that RS will lead to a gain compared with the pull-only setting. For simplicity we will suppose duels follow a comparison model given as follows: $M_{ij} = \Pr[i \succ j] = \frac{1+\mu_i-\mu_j}{2}$. This is known as the linear link function since it linearly relates the duel win probability with the reward means. Routine calculations show that under a linear link function we have $p_i - p_j = \Theta(\mu_i - \mu_j)$. We require that both our method and pull-only method succeed with probability $1-\delta$, with a small constant δ (e.g., $\delta = 0.05$). Both of our positive examples assume that the means are dense near the boundaries given by $\mu = 0$ and $\mu = 1$, while a very small fraction of arms have means near $1/2$, so that there is a significant gap between the arms i_u and i_l closest to the threshold, as well as any arm i and arm i_u or i_l that is closest to it(cf. Figure 2). Although these examples can look artificial at first sight, we note that such a bowl-shaped distribution is common in practice, and is similar to Tsybakov noise (Tsybakov et al., 2004) assumption used to characterize classification noise in the machine learning literature.

Example 1. Suppose $K = 2l$, and $\mu_i = \frac{1}{(l+3)-i}$ for $1 \le i \le l$, and $\mu_i = 1 - \frac{1}{i-(l-2)}$ for $l+1 \le i \le 2l$ (see Figure 2 left). We will have $\Delta_i^l = \bar{\Delta}_i^c = \Omega(1)$ for all arms $i \in \mathcal{A}$. Then the previous state-of-art CLUCB algorithm requires $O(K \log K)$ pulls, and their lower bounds show that any pull-only algorithm needs at least $\Omega(K)$ pulls. On the other hand, our algorithm requires $O(K \log K)$ duels and only $O(\log^2 K)$ pulls.

When pulls are more expensive than duels, there is a significant cost saving when using our RS algorithm.

Example 2. Suppose $K = 2l$. Sample $x_1, ..., x_K$ from an exponential distribution with parameter $\lambda = 4\log(4l/\delta)$, and let $\mu_i = x_i$ for $1 \le i \le l$, and $\mu_i = 1 - x_i$ for $l + 1 \le i \le 2l$ (see Figure 2 right). Then with probability $1 - \delta$: i) $\mu_i \in [0, 1]\ \forall i \in [K]$; ii) $\Delta_i^l = \Omega(1)$, and $H_{c,2} = H_{c,1}$; iii) CLUCB requires $O(K \log K)$ pulls, and any pull-only algorithm requires at least $\Omega(K)$ pulls; iv) Our algorithm requires $O(K \log^3 K)$ duels and $O(\log^2 K)$ pulls.

We provide proofs of the results for these two examples in the appendix.

5 Lower Bounds

In this section, we give lower bounds that complement our upper bounds. We first give an arm-wise lower bound in Section 5.1 to show that RS is almost optimal in terms of the total number of queries to each individual arm. Then, we discuss the optimality of both n_{duel} and n_{pull} in Section 5.2.

For simplicity, in this section we suppose all rewards follow a Gaussian distribution with parameter R, i.e., $\nu_i = \mathcal{N}(\mu_i, R^2)$. Our results can be easily extended to other sub-Gaussian distributions (e.g., when all rewards are binary).

5.1 An Arm-Wise Lower Bound

The following theorem gives a lower bound on the number of pulls and duels on a particular arm k.

Theorem 3. *Suppose an algorithm $\mathcal{A}$ recovers S_τ with probability $1 - \delta$ for any problem instance $(M, \boldsymbol{\mu})$ and $\delta \le 0.15$. For any arm i, let $D_i^{\mathcal{A}}$ be the number of times that arm i is selected for a duel, and $L_i^{\mathcal{A}}$ be the number of times that arm i is pulled. Let $c = \min\{\frac{1}{10}, \frac{R^2}{2}\}$. Then for any problem instance $(M, \boldsymbol{\mu})$ with $M_{ij} \ge \frac{3}{8}$ for every arm $i, j \in [K]$, and a specific arm $k \in S_\tau$, we have*

$$\mathbb{E}_{M,\boldsymbol{\mu}}[(\Delta_k^c)^2 D_k^{\mathcal{A}} + (\Delta_k^l)^2 L_k^{\mathcal{A}}] \ge c \log(\frac{1}{2\delta}). \quad (2)$$

Theorem 3 shows an arm-wise lower bound that the sum of duels and pulls (weighted by their complexity) must satisfy (2). In the pull-only setting, this agrees with the known result that number of pulls needed for an arm k is $\Omega((\Delta_k^l)^{-2})$. And for duel-choice setting, it shows that if we never pull arm k, number of duels involving arm k (against some known arm) is at least $\Omega((\Delta_k^c)^{-2})$. From our proof of Theorem 1, we can easily show the following proposition for the upper bound that RS achieves:

Proposition 4. *For any problem instance $(M, \boldsymbol{\mu})$ and arm k, Algorithm RS succeeds with probability at least $1 - \delta$ and there exists a constant C such that the RS algorithm achieves that*

$$\mathbb{E}_{M,\boldsymbol{\mu}}[(\bar{\Delta}_k^c)^2 D_k^{\mathsf{RS}} + (\Delta_k^l)^2 L_k^{\mathsf{RS}}] \le C \log\left(\frac{K \log(\frac{K}{\gamma^* \Delta^*})}{\delta}\right). \quad (3)$$

Comparing (3) with (2), our RS algorithm is arm-wise optimal except for the difference of Δ_k^c vs. $\bar{\Delta}_k^c$, and the log factors. This shows that RS is near optimal in the sum $\mathbb{E}_{M,\boldsymbol{\mu}}[(\Delta_k^c)^2 D_k^{\mathcal{A}} + (\Delta_k^l)^2 L_k^{\mathcal{A}}]$.

5.2 Optimality of n_{duel} and n_{pull}

In this subsection, we analyze the lower bound of TBP-DC under the case when duels are much cheaper than pulls. In this case, we would like to minimize the number of pulls, and then minimize the number of duels. Intuitively, RS algorithm is optimal in n_{pull} as it uses roughly $O(\log K)$ pulls; this is necessary even if we know a perfect ranking of all arms (due to the complexity of binary search). We consider an extreme case, where we know the label of arm i_u and i_l from pulls, and wish to infer all other labels using duels. The following corollary of Theorem 3 shows a lower bound in this case:

Corollary 5. *Suppose an algorithm $\mathcal{A}$ is given that $i_u \in S_\tau$ and $i_l \in S_\tau^c$, and uses only duels. Under the same assumption as in Theorem 3, the number of duels of $\mathcal{A}$ is at least $\mathbb{E}[n_{duel}^{\mathcal{A}}] \ge cH_{c,1} \log(1/2\delta)$.*

Combining Corollary 5 with the fact that $O(\log K)$ is necessary for TBP-DC, we show that RS is near optimal in both n_{duel} and n_{pull}.

6 Experiments

To verify our theoretical insights, we perform experiments on a series of settings to illustrate the efficacy of RS, on both synthetic and real-world data. For comparison, we include the state-of-art CLUCB in the pull-only setting, and several naive baselines.

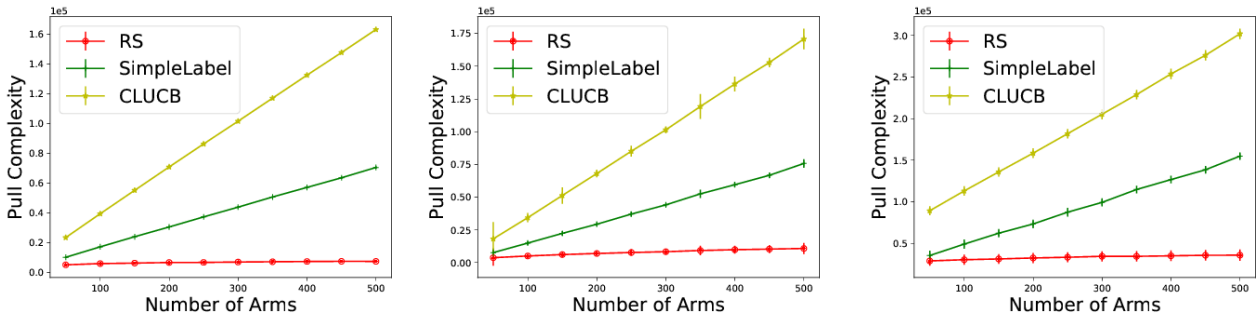
6.1 Data Configuration

For synthetic data, we vary the number of arms K from 50 to 500, and set threshold $\tau = 0.5$. The duels follows from a linear link function $\Pr[i \succ j] = \frac{1+\mu_i-\mu_j}{2}$ [1]. Let $K = 2l$, the mean rewards are given as below:

Experiment 1 (harmonic): This is Example 1 from Section 4.

Experiment 2 (exponential): This is Example 2

[1] We include the results with a BTL model in the appendix.



(a) harmonic (b) exponential (c) 3 groups

Figure 3: Empirical results comparing RS and other baselines. Error bars represent standard deviation across 500 experiments.

from Section 4.

Experiment 3 (3groups): This is similar to the example in (Locatelli et al., 2016). Let $\mu_i = 0.1$ for $i = 1, 2, ..., l-2$, $\mu_{(l-1):(l+2)} = (0.35, 0.45, 0.55, 0.65)$, and $\mu_i = 0.9$ for $i = l+3, ..., K$.

For real-world data, we use the reading difficulty dataset collected by Chen et al. (2013). The data consists of 491 passages, each with a reading difficulty level ranged in 1-12. We randomly take K passages from the whole set, with K varying from 50 to 491. Let $\mu_i = l_i/13$, where l_i is the difficulty level of passage i. The goal here is to identify the difficult passages with level at least 7, or equivalently $\tau = 0.5$. Although the original dataset from Chen et al. (2013) comes with comparisons, it does not cover all pairs and we therefore use a probabilistic model to generate comparison feedbacks. Specifically, we experiment with two types of comparison models: i) linear link function $\Pr[i \succ j] = \frac{1+\theta(\mu_i-\mu_j)}{2}$; ii) BTL model: $\Pr[i \succ j] = \frac{1}{1+e^{(\mu_j-\mu_i)\theta}}$. For both model, we find the θ that maximizes the log likelihood based on comparisons data provided in (Chen et al., 2013). Hypothesis testing against a null hypothesis ($\Pr[i \succ j] = 1/2$) gives p-values less than 1×10^{-4} for both models.

6.2 Baselines and Implementation Details

We compare performance of the following methods.

CLUCB(Chen et al., 2014): We implement the CLUCB algorithm which only queries for selective direct pulls in a TBP setting.

SimpleLabel: This is a simple pull-only baseline where we apply Figure-Out-Label to all the arms $i \in \mathcal{A}$ with confidence δ/K.

RankThenSearch: As we discussed in introduction, we compare to the baselines where we first use a ranking algorithm to rank all the arms, and then perform a binary search to find the boundary. We consider two methods for the first ranking step. i) ActiveRank(Heckel et al., 2016): An active ranking algorithm that achieves optimal rates based on Borda scores. ii) PLPAC-AMPR(Szörényi et al., 2015): Another ranking algorithm that focuses on BTL model. After the ranking algorithm we run a single binary search on the sorted sequence, using Figure-Out-Label to identify labels.

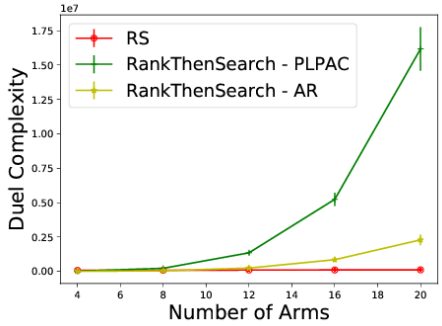
RankSearch: Our algorithm. The parameters of our algorithms are the initial confidence γ_0 and shrinking factor κ. Both of them decides how aggressive we decrease our confidence: A small γ_0 will lead to a starting point with high confidence, and a large κ will increase the confidence level quickly. Both of them will lead to a higher number of duels. In our implementation, we pick γ_0 adaptively so that $\max \hat{p}_i - \min \hat{p}_i \geq 2\gamma_0$ (see Appendix for details), and use $\kappa = 2$.

We note that previous works on TBP in the fixed budget setting (Locatelli et al., 2016; Mukherjee et al., 2017) cannot be implemented in our fixed-confidence setting.

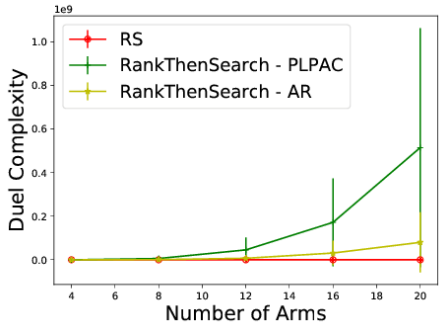
We run all the methods with varying number of arms, and compare their performance to reach confidence $\delta = 0.95$. For complexity notion, since there is no predefined cost ratio between duels and pulls, we compare the pull and duel complexity of RS separately with the baselines. Specifically, we compare pull complexity with SimpleLabel and CLUCB, and compare duel complexity with RankThenSearch (since the other two baselines are pull-only algorithms). Each experiment is repeated 500 times, and we compute the mean and standard deviation of each baseline's performance.

6.3 Experiment Results

Results on synthetic data. In Figure 3, we plot the empirical pull complexity of RS along with the baselines of CLUCB and SimpleLabel. As expected, the number of pulls of RS is much lower than the baseline algorithms in all three experiments we consider. Interestingly, SimpleLabel also has an advantage over CLUCB in the pull-only setting. We note that

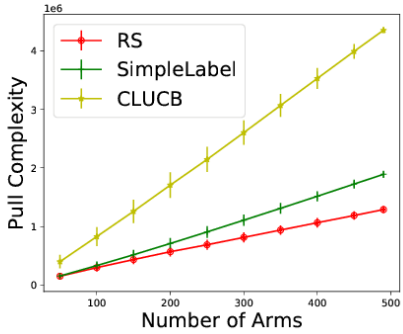


(a) harmonic

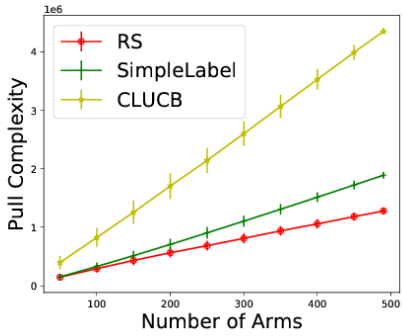


(b) exponential

Figure 4: Empirical results comparing RS and RankThenSearch. Error bars represent standard deviation across 500 experiments. PLPAC is short for PLPAC-AMPR.



(a) Linear Link Function



(b) BTL Link Function

Figure 5: Empirical results comparing RS and other baselines. Error bars represent standard deviation across 500 experiments.

CLUCB's $O(H_l \log(\frac{H_l}{\delta}))$ is only optimal up to $\log(H_l)$ factors, and SimpleLabel might have an advantage because its pull complexity is $O(H_l \log(\frac{K \log \Delta^*}{\delta}))$ in the pull-only setting, slightly better than CLUCB. This advantage and the optimal rate for the pull-only setting is of independent interest and we leave it as future work.

We then compare the duel complexity with RankThenSearch in Figure 4. Since RankThenSearch needs to differentiate between every pair of arms, the algorithms take extremely long to run and we have to limit the arms to be at most 20 (as is done in Szörényi et al. (2015)). Note that since in **3groups** the arms are not separable, we only compare to RankThenSearch in the first two settings. The results show that RankThenSearch with ActiveRank and PLPAC-AMPR both acquires an incredible number of duels in order to rank the arms: to rank 20 arms they acquire hundreds of millions (1×10^8) of duels, for the **exponential** arm setup. This prohibitive cost makes it impossible to adopt the RankThenSearch method. We also observed a very large variance in performance for RankThenSearch, because differentiating arms close to each other is very unstable. Dueling complexity of RS is much lower and more stable than the above methods, and therefore RS achieves a balance between duels and pulls.

Results on real-world data. Finally, we compare the pull complexity between RS and the pull-only baselines on real-world data in Figure 5. RS still performs better than both baselines for the real data, but the advantage of RS over the baselines are lower than on synthetic data. This is possibly because the data contains many passages near the boundary (i.e., grade 6 and 7), and RS have to use pulls to identify their label. We verify this empirically in the appendix.

7 Conclusion

We formulate a new setting of the Thresholding Bandit Problem with Dueling Choices, and provide the RS algorithm, along with upper and lower bounds on its performance. For future work, it would be interesting to tighten the upper and lower bounds to match them; We believe it should be possible to improve the lower bound by randomizing the arms closest to the threshold. It would also be interesting to develop algorithms adapting to varying noise levels in comparisons.

Acknowledgements

This work has been supported in part by DARPA FA8750-17-2-0130, NSF CCF-1763734 and IIS-1845444, and AFRL FA8750-17-2-0212.

References

Abbasi-Yadkori, Y., Bartlett, P., Chen, X., and Malek, A. (2016). Large-scale markov decision problems with kl control cost and its application to crowdsourcing. In *In Proceedings of the International Conference on Machine Learning (ICML)*.

Bradley, R. A. and Terry, M. E. (1952). Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Bubeck, S., Wang, T., and Viswanathan, N. (2013). Multiple identifications in multi-armed bandits. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Chen, J., Chen, X., Zhang, Q., and Zhou, Y. (2017). Adaptive multiple-Arm identification. In *In Proceedings of the International Conference on Machine Learning (ICML)*.

Chen, S., Lin, T., King, I., Lyu, M. R., and Chen, W. (2014). Combinatorial pure exploration of multi-armed bandits. In *Advances in Neural Information Processing Systems*, pages 379–387.

Chen, X., Bennett, P. N., Collins-Thompson, K., and Horvitz, E. (2013). Pairwise ranking aggregation in a crowdsourced setting. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 193–202. ACM.

Chen, X., Lin, Q., and Zhou, D. (2015). Statistical decision making for optimal budget allocation in crowd labeling. *The Journal of Machine Learning Research*, 16(1):1–46.

Fishburn, P. C. (1973). Binary choice probabilities: on the varieties of stochastic transitivity. *Journal of Mathematical psychology*, 10(4):327–352.

Heckel, R., Shah, N. B., Ramchandran, K., and Wainwright, M. J. (2016). Active ranking from pairwise comparisons and when parametric assumptions don't help. *arXiv preprint arXiv:1606.08842*.

Kane, D. M., Lovett, S., Moran, S., and Zhang, J. (2017). Active classification with comparison queries. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 355–366. IEEE.

Kaufmann, E., Cappé, O., and Garivier, A. (2016). On the complexity of best-arm identification in multi-armed bandit models. *The Journal of Machine Learning Research*, 17(1):1–42.

Locatelli, A., Gutzeit, M., and Carpentier, A. (2016). An optimal algorithm for the thresholding bandit problem. In *Proceedings of the 33rd International Conference on Machine Learning-Volume 48*, pages 1690–1698. JMLR. org.

Massart, P. and Nédélec, É. (2007). Risk bounds for statistical learning. *arXiv Mathematics e-prints*, page math/0702683.

Mohajer, S., Suh, C., and Elmahdy, A. (2017). Active learning for top-k rank aggregation from noisy comparisons. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2488–2497. JMLR. org.

Mukherjee, S., Purushothama, N. K., Sudarsanam, N., and Ravindran, B. (2017). Thresholding bandits with augmented ucb. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 2515–2521. AAAI Press.

Shah, N., Balakrishnan, S., Guntuboyina, A., and Wainwright, M. (2016a). Stochastically transitive models for pairwise comparisons: Statistical and computational issues. In *International Conference on Machine Learning*, pages 11–20.

Shah, N. B., Balakrishnan, S., Bradley, J., Parekh, A., Ramchandran, K., and Wainwright, M. J. (2016b). Estimation from pairwise comparisons: Sharp minimax bounds with topology dependence. *The Journal of Machine Learning Research*, 17(1):2049–2095.

Szörényi, B., Busa-Fekete, R., Paul, A., and Hüllermeier, E. (2015). Online rank elicitation for plackett-luce: A dueling bandits approach. In *Advances in Neural Information Processing Systems*, pages 604–612.

Thurstone, L. L. (1927). A law of comparative judgment. *Psychological review*, 34(4):273.

Tsybakov, A. B. et al. (2004). Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32(1):135–166.

Xu, Y., Muthakana, H., Balakrishnan, S., Singh, A., and Dubrawski, A. (2018). Nonparametric regression with comparisons: Escaping the curse of dimensionality with ordinal information. In *International Conference on Machine Learning*, pages 5469–5478.

Xu, Y., Zhang, H., Miller, K., Singh, A., and Dubrawski, A. (2017). Noise-tolerant interactive learning using pairwise comparisons. In *Advances in Neural Information Processing Systems*, pages 2431–2440.

Yue, Y., Broder, J., Kleinberg, R., and Joachims, T. (2012). The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556.

Zhou, Y., Chen, X., and Li, J. (2014). Optimal PAC multiple arm identification with applications to crowdsourcing. In *In Proceedings of the International Conference on Machine Learning (ICML)*.