From PAC to Instance-Optimal Sample Complexity in the Plackett-Luce Model

Aadirupa Saha [1] **Aditya Gopalan** [1]

Abstract

We consider PAC-learning a good item from k-subsetwise feedback information sampled from a Plackett-Luce probability model, with *instance-dependent* sample complexity performance. In the setting where subsets of a fixed size can be tested and top-ranked feedback is made available to the learner, we give an algorithm with optimal instance-dependent sample complexity, for PAC best arm identification, of $O\bigg(\frac{\Theta_{[k]}}{k}\sum_{i=2}^{n}\max\Big(1,\frac{1}{\Delta_i^2}\Big)\ln\frac{k}{\delta}\Big(\ln\frac{1}{\Delta_i}\Big)\bigg)$, Δ_i being the Plackett-Luce parameter gap between the best and the i^{th} best item, and $\Theta_{[k]}$ is the sum of the Plackett-Luce parameters for the top-k items. The algorithm is based on a wrapper around a PAC winner-finding algorithm with weaker performance guarantees to adapt to the hardness of the input instance. The sample complexity is also shown to be multiplicatively better depending on the length of rank-ordered feedback available in each subset-wise play. We show optimality of our algorithms with matching sample complexity lower bounds. We next address the winner-finding problem in Plackett-Luce models in the fixed-budget setting with instance dependent upper and lower bounds on the misidentification probability, of $\Omega\Big(\exp(-2\tilde{\Delta}Q)\Big)$ for a given budget Q, where $\tilde{\Delta}$ is an explicit instance-dependent problem complexity parameter. Numerical performance results are also reported.

1. Introduction

We consider the problem of sequentially learning the best item of a set when subsets of items can be tested but information about only their relative strengths is observed. This is a basic search problem motivated by applications in recommender systems and information retrieval (Hofmann et al., 2013; Radlinski et al., 2008), crowdsourced ranking (Chen et al., 2013), tournament design (Graepel & Herbrich, 2006), etc. It has received recent attention in the online learning community, primarily under the rubric of dueling bandits (e.g., (Yue et al., 2012) and online ranking in the Plackett-Luce (PL) discrete choice model (Chen et al., 2018; Saha & Gopalan, 2019; Ren et al., 2018).

Our focus in this paper is to study the instance-dependent complexity of learning the (near) best item in a subset-wise PL feedback model by which we mean the following. Each item has an a priori unknown PL weight parameter, and every time a subset of alternatives is selected, an item or items sampled from the PL probability distribution over the subset are observed by the learner. Given a tolerance ϵ and confidence level δ, the learner faces the task of sequentially playing subsets of items, and stopping and finding an ϵ-optimal arm, i.e., an arm i whose PL parameter satisfies $\theta_i \ge \max_j \theta_j - \epsilon$, with probability of error at most δ.

Existing work on best arm learning in PL models, e.g., (Saha & Gopalan, 2019), focuses on attaining the worst-case or *instance-independent* sample complexity of learning an approximately best item. By this, we mean that the typical goal is to design algorithms that terminate in a number of rounds bounded by a function of only ϵ, δ and the number of arms n, typically of the form $O\left(\frac{n}{\epsilon^2}\log\left(\frac{1}{\delta}\right)\right)$ rounds. Such worst-case results, though significantly novel, suffer from two weaknesses: (1) The termination time guarantees become vacuous in the setting where an exact best arm is sought ($\epsilon = 0$), and (2) The guarantees do not reflect the fact that some problem instances, in terms of their items' PL parameters, are easier than others to learn, e.g., the instance with parameters $(\theta_1, \ldots, \theta_n) = (1, 0.01, \ldots, 0.01)$ ought to be much easier than $(1, 0.99, \ldots, 0.99)$ since item 1 is a distinctly clearer winner than in the latter case. In this paper, we set ourselves the more challenging objective of quantifying and attaining sample complexity that depends on the inherent 'hardness' of the PL instance. In this context, we make the following contributions:

(1) We give the first instance-optimal algorithm for the problem of (ϵ, δ)-PAC learning a best item in a PL model when subsets of a fixed size can be tested in each round. This is accomplished by building a novel wrapper algorithm (Alg. 1) around an (ϵ, δ)-PAC learning algorithm used as

[1]Indian Institute of Science, Bangalore, India. Correspondence to: Aadirupa Saha <aadirupa@iisc.ac.in>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

a subroutine that we designed (Alg. 5). We also provide a matching instance-dependent lower bound on the sample complexity of any algorithm, to establish the optimality of our algorithm (Thm. 3,4,7).

(2) When richer, m length rank-ordered information is observed per subsetwise query, we show the optimal instance-dependent sample complexity lower bound is much smaller than just with the winner feedback case (Thm. 8). We also propose an optimal algorithm for this setting (Alg. 8) with an $\frac{1}{m}$-factor improved sample complexity guarantee which is shown to be optimal (Thm. 5).

(3) We also study the fixed-budget version of the best-item learning problem, where a learning horizon of Q rounds is specified instead of a desired confidence level δ, and the performance measure of interest is the probability of error in identifying a best arm. We give an algorithm for learning the best item of a Plackett-Luce instance under a fixed budget with general m-way ranking feedback (Alg. 8, Thm. 12), and also prove an instance-dependent lower bound for it (Thm. 11).

Our theoretical findings are also supported with numerical experiments on different datasets.

Related work. For classical multiarmed bandits setting, there is a well studied literature on PAC-arm identification problem (Even-Dar et al., 2006; Audibert & Bubeck, 2010; Kalyanakrishnan et al., 2012; Karnin et al., 2013; Jamieson et al., 2014), where the learner gets to see a noisy draw of absolute reward feedback of an arm upon playing a single arm per round. Some of the existing results on dueling bandits line of works also focuses on PAC learning from pairwise preference feedback for best arm identification problem (Yue & Joachims, 2011; Urvoy et al., 2013; Szörényi et al., 2015; Busa-Fekete et al., 2014a), or even more general problem objectives e.g. PAC top set recovery (Busa-Fekete et al., 2013; Mohajer et al., 2017; Chen et al., 2017), or PAC-ranking of items (Busa-Fekete et al., 2014b; Falahatgar et al., 2017), even in the feedback setup of noisy comparisons (Braverman & Mossel, 2008; Caragiannis et al., 2013). There are also very few recent developments that focuses on learning for subsetwise feedback in an online setup (Sui et al., 2017; Brost et al., 2016; Saha & Gopalan, 2018a; 2019; Ren et al., 2018; Chen et al., 2018). Some of the existing work also explicitly consider the Plackett-Luce parameter estimation problem with subset-wise feedback but for offline setup only (Jang et al., 2017; Khetan & Oh, 2016). While most of the above work address the (ϵ,δ)-PAC recovery problem, i.e. finding an 'ϵ-approximation' of the desired (set of) item(s) with probability at least $(1-\delta)$, few of them also focuses of instant dependent PAC recovery guarantees where the sample complexity explicitly depends of the parameters of the underlying model, e.g. for classical multiarmed bandits (Audibert & Bubeck, 2010; Karnin et al., 2013; Kalyanakrishnan et al., 2012), or even for preference based bandits (Szörényi et al., 2015; Chen et al., 2018).

2. Problem Setup

Notation. We denote by $[n]$ the set $\{1,2,...,n\}$. For any subset $S \subseteq [n]$, let $|S|$ denote the cardinality of S. When there is no confusion about the context, we often represent (an unordered) subset S as a vector, or ordered subset, S of size $|S|$ (according to, say, a fixed global ordering of all the items $[n]$). In this case, $S(i)$ denotes the item (member) at the ith position in subset S. For any ordered set S, $S(i:j)$ denotes the set of items from position i to j, $i<j$, $\forall i,j \in [|S|]$. We denote by $\boldsymbol{\Sigma}_S = \{\sigma \mid \sigma$ is a permutation over items of $S\}$, where for any permutation $\sigma \in \Sigma_S$, $\sigma(i)$ denotes the element at the i-th position in $\sigma, i \in [|S|]$. We also denote by $\boldsymbol{\Sigma}_S^m$ the set of permutations of any m-subset of S, for any $m \in [k]$, i.e. $\Sigma_S^m := \Sigma_{S'}$ s.t. $S' \subseteq S$, $|S'| = m$. $\mathbf{1}(\varphi)$ is generically used to denote an indicator variable that takes the value 1 if the predicate φ is true, and 0 otherwise. $x \vee y$ denotes the maximum of x and y, and $Pr(A)$ is used to denote the probability of event A, in a probability space.

Definition 1 (Plackett-Luce probability model)**.** *A Plackett-Luce probability model, specified by positive parameters $(\theta_1,\ldots,\theta_n)$, is a collection of probability distributions $\{Pr(\cdot|S) : S \subset [n], S \neq \emptyset\}$, where for each non-empty subset $S \subseteq [n]$, $Pr(i|S) = \frac{\theta_i \mathbf{1}(i \in S)}{\sum_{j\in S}\theta_j}$ $\forall 1 \le i \le n$. The indices $1,\ldots,n$ are referred to as 'items' or 'arms' .*

Since the Plackett-Luce probability model is invariant to positive scaling of its parameters $\boldsymbol{\theta} \equiv (\theta_i)_{i=1}^n$, we make the standard assumption that $\max_{i\in[n]}\theta_i = 1$.

An online learning algorithm is assumed to interact with a Plackett-Luce probability model over n items (the 'environment') as follows. At each round $t = 1,2,\ldots$, the algorithm decides to either (a) terminate and return an item $I \in [n]$, or (b) play (test) a subset $S_t \subset [n]$ of k distinct items, upon which it receives stochastic feedback whose distribution is governed by the probability distribution $Pr(\cdot|S_t)$. We specifically consider the following structures for feedback received upon playing a subset S:

1. **Winner feedback:** The environment returns a single item J drawn independently from the probability distribution $Pr(\cdot|S)$ where $Pr(J=j|S) = \frac{\theta_j}{\sum_{k\in S}\theta_k}$ $\forall j \in S$.

2. **Top-m Ranking feedback** $(1 \le m \le k-1)$**:** Here, the environment returns an ordered list of m items sampled without replacement from the Plackett-Luce probability model on S. More formally, the environment returns a partial ranking $\boldsymbol{\sigma} \in \boldsymbol{\Sigma}_S^m$, drawn from the probability distribution $Pr(\boldsymbol{\sigma} = \sigma|S) = \prod_{i=1}^m \frac{\theta_{\sigma^{-1}(i)}}{\sum_{j\in S\setminus\sigma^{-1}(1:i-1)}\theta_j}$, $\sigma \in \boldsymbol{\Sigma}_S^m$. This can also be seen as picking an item $\boldsymbol{\sigma}^{-1}(1) \in S$ accord-

ing to *Winner feedback* from S, then picking $\boldsymbol{\sigma}^{-1}(2)$ from $S \setminus \{\boldsymbol{\sigma}^{-1}(1)\}$, and so on for m times. When $m = 1$, Top-m Ranking feedback is the same as Winner feedback.

Definition 2 ((ϵ, δ)-PAC or fixed-confidence algorithm)**.** *An online learning algorithm is said to be (ϵ, δ)-PAC with termination time bound Q if the following holds with probability at least $1 - \delta$ when it is run in a Plackett-Luce model: (a) it terminates within Q rounds (subset plays), (b) the returned item I is an ϵ-optimal item: $\theta_I \geq \max_{i \in [n]} \theta_i - \epsilon = 1 - \epsilon$. (Probability is over both the environment and the algorithm.)*

By the *sample complexity* of an (ϵ, δ)-PAC online learning algorithm $\mathcal{A}$ for a Plackett-Luce instance $\boldsymbol{\theta} \equiv (\theta_i)_{i=1}^n$ and playable subset size k, we mean the smallest possible termination time bound Q for the algorithm when run on $\boldsymbol{\theta}$. We use the notation $N_{\mathcal{A}}(\epsilon, \delta) \equiv N_{\mathcal{A}}(\epsilon, \delta, \boldsymbol{\theta}, n, k)$ to denote this sample complexity. We aim to design (ϵ, δ)-PAC algorithms with as small a value of sample complexity as possible, depending on the number of items n, the playable subset size k, approximation error ϵ, confidence δ, and most importantly, the Plackett-Luce model parameters $(\theta_i)_{i=1}^n$. We also assume item 1 is optimal: $\theta_1 = \max_{i \in [n]} \theta_i = 1$, and $\Delta_i = \theta_1 - \theta_i$ for any $i \in [n]$.

3. Instance-dependent regret for *Probably-Correct-Best-Item* problem

3.1. Prelude: An algorithm for $\epsilon = 0$

For clarity of exposition, we first describe the design of a $(0, \delta)$-PAC or *Probably-Correct-Best-Item* learning algorithm, i.e., an algorithm that attempts to learn the unique best item in a Plackett-Luce model when such an item exists[1]: $1 = \theta_1 > \max_{i \geq 2} \theta_i$. This is then generalised in the next section to an online learning algorithm that is (ϵ, δ)-PAC.

High-level idea behind algorithm design. The algorithm we propose (*PAC-Wrapper*) is based on using an (ϵ, δ)-PAC-algorithm known to have (expected) termination time bounded in terms of ϵ and δ (a 'worst' case termination guarantee not necessarily dependent on instance parameters) as an underlying black-box subroutine. The wrapper algorithm uses the black-box repeatedly, with successively more stringent values of ϵ and δ, to eliminate suboptimal arms in a phased manner. The termination analysis of the algorithm shows that any suboptimal arm $i \in [n] \setminus \{1\}$ survives for about $O\left(\frac{1}{\Delta_i^2} \ln \frac{k}{\delta}\right)$ rounds before being eliminated, which leads to the desired bound of $O\left(\sum_{i=2}^n \frac{1}{\Delta_i^2} \ln \frac{k}{\delta}\right)$ on algorithm's run time performance (with high probability $(1-\delta)$) (Thm. 3).

[1]When there is more than one best item the problem of finding a best item with confidence is not well-defined.

Algorithm description. The *PAC-Wrapper* algorithm we propose (Alg. 1) runs in phases indexed by $s = 1, 2, \ldots$, where each phase s comprises of the following steps.

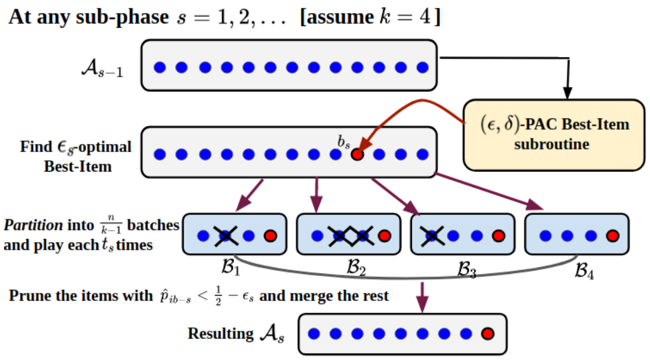
At any sub-phase $s = 1, 2, \ldots$ [assume $k = 4$]
$\mathcal{A}_{s-1}$
(ϵ, δ)-PAC Best-Item subroutine
Find ϵ_s-optimal Best-Item
b_s
Partition into $\frac{n}{k-1}$ batches and play each t_s times
$\mathcal{B}_1$
$\mathcal{B}_2$
$\mathcal{B}_3$
$\mathcal{B}_4$
Prune the items with $\hat{p}_{ib-s} < \frac{1}{2} - \epsilon_s$ and merge the rest
Resulting $\mathcal{A}_s$

Figure 1. A sample run of Alg. 1 at any sub-phase s with the set of surviving arms $\mathcal{A}_{s-1}$: **Step 1.** The algorithm finds a (ϵ_s, δ_s)-PAC item b_s, where $\epsilon_s = \frac{1}{2^{s+2}}$ and $\delta_s = \frac{\delta}{40s^3}$. **Step 2.** It partitions $\mathcal{A}_{s-1}$ into $B_s = \left\lceil \frac{\mathcal{A}_{s-1}}{k-1} \right\rceil$ groups $\mathcal{B}_1, \ldots \mathcal{B}_{B_s}$ of size k, each containing b_s, and plays each for $t_s = \frac{2k}{\epsilon_s^2} \ln \frac{k}{\delta_s}$ times. **Step 3.** Based on the received feedback of t_s plays, the algorithm updates the empirical pairwise probability $\hat{p}_{ij}$ of each item pair (i, j) within a group $\mathcal{B}$ by applying *Rank-Breaking*, and discards any item $i \in \mathcal{B}$ with $\hat{p}_{ib_s} < \frac{1}{2} - \epsilon_s$. The rest of the surviving items are then combined to $\mathcal{A}_s$, and the algorithm recurses to $s + 1$.

Step 1: Finding a good reference item. It first calls an (ϵ_s, δ_s)-PAC subroutine (described in Sec. 3.4 for completeness) with $\epsilon_s = \frac{1}{2^{s+2}}$ and $\delta_s = \frac{\delta}{120s^3}$ to obtain a 'reasonably good item' b_s—an item that is likely within an ϵ_s margin of the Best-Item with probability at least $(1 - \delta_s)$) and thus a potential Best-Item. For this we design a new sequential elimination-based algorithm (Alg. 5 in Appendix A.3), and argue that it finds such a (ϵ_s, δ_s)-PAC 'good item' with *instance-dependent* sample complexity (Thm. 6), which is crucial in the overall analysis. This is an improvement upon the instance-agnostic Algorithm 6 of (Saha & Gopalan, 2019) whose sample complexity guarantee is not strong enough to be used along with the wrapper.

Step 2: Benchmarking items against the reference item. After obtaining a candidate good item, the algorithm divides the rest of the current surviving arms into equal-sized groups of size $k - 1$, say the groups $\mathcal{B}_1, \ldots, \mathcal{B}_{B_s}$, and 'stuffs' the good 'probe' item b_s into each group, creating $B_s = \left\lceil \frac{\mathcal{A}_{s-1}}{k-1} \right\rceil$ item groups of size k (the Partition subroutine, Algorithm 2, Appendix A.2). It then plays each group $\mathcal{B}_b$, $b \in [B_s]$ for a total of $t_s = \frac{2\hat{\Theta}_S}{\epsilon_s^2} \ln \frac{k}{\delta_s}$ rounds, where $\hat{\Theta}_S$ denotes a 'near-accurate' relative score estimate of the Plackett-Luce model for the set $\mathcal{B}_b$–we use the subroutine *Score-Estimate* for estimating $\hat{\Theta}_S$ (see Alg. 3, Thm. 13 in Appendix A.2). From the winner data obtained in this process, it updates the empirical pairwise win count w_i of each item within any batch $\mathcal{B}_b$ by applying a rank-breaking idea (see Alg. 4, Appendix A.2) .

Step 3: Discarding items weaker than the reference item. Finally, from each group $\mathcal{B}_b$, the algorithm eliminates all arms whose empirical pairwise win frequency over the probe item b_s is less than $\frac{1}{2} - \epsilon_s$ (i.e. $\forall i \in \mathcal{B}_b$ for which $\hat{p}_{ib_s} < \frac{1}{2} - \epsilon_s$, $\hat{p}_{ij}$ being the empirical pairwise preference of item i over j obtained via *Rank-Breaking*). The next phase then begins, unless there is only one surviving item left, which is output as the candidate Best-Item. Pointers to the 4 subroutines used in the overall algorithm are as below.

(1). (ϵ,δ)***-PAC Best-Item* subroutine:** Given $\epsilon,\delta \in (0,1)$, this finds an (ϵ,δ)-Best-Item in $O\Big(\frac{\Theta_{[k]}}{\epsilon^2}\ln\frac{k}{\delta}\Big)$ samples, where $\Theta_{[k]} = \max\limits_{S\subseteq[n]||S|=k}\sum_{i\in S}\theta_i$ (See Alg. 5, Thm. 6 in Appendix A.3).

(2). ***Rank-Breaking* subroutine:** This is a procedure of deriving pairwise comparisons from multiwise (subsetwise) preference information (Soufiani et al., 2014; Khetan & Oh, 2016). (See Alg. 4, Appendix A.2).

(3). ***Score-Estimate* subroutine:** Given a set S and a reference item $b \in [n]$, this estimates the relative Plackett-Luce scores of the set w.r.t. b (see Alg. 3, Appendix A.2).

(4). ***Partition*:** This partitions a given set of items into equally sized batches (See Alg. 2, Appendix A.2).

Fig. 1 graphically depicts a sample run of a sub-phase s (for $k = 4$). Note that as the playable subset size is k, we need to specially treat the final few sub-phases when the number of surviving arms (i.e. $|\mathcal{A}_s|$) falls below k (Lines 22-31 in Alg. 1). The complete algorithm is given in Appendix A.1.

Theorem 3 (*PAC-Wrapper*$(0,\delta)$-PAC sample complexity bound with Winner feedback)**.** *With probability at least* $(1-\delta)$*,* $\mathcal{A}$ *as PAC-Wrapper (Algorithm 1) returns the Best-Item with sample complexity* $N_{\mathcal{A}}(0,\delta) = O\bigg(\frac{\Theta_{[k]}}{k}\sum_{i=2}^n \max\Big(1,\frac{1}{\Delta_i^2}\Big)\ln\frac{k}{\delta}\Big(\ln\frac{1}{\Delta_i}\Big)\bigg)$*, where* $\Theta_{[k]} := \max_{S\subseteq[n]||S|=k}\sum_{i\in S}\theta_i$*.*

Remark 1. As $\Theta_{[k]} \le k$, *PAC-Wrapper* takes $O(\frac{1}{\Delta_i^2})\ln\frac{1}{\delta}$ rounds to eliminate all suboptimal items with confidence δ. However, the dependence of the upper bound on $\Theta_{[k]}$ implies a $1/k$ factor gain in sample complexity when the underlying instance is 'easy'. Indeed, when $\Theta_{[k]} = O(1)$, e.g., in an instance where $\theta_1 \approx 1$ and $\theta_i \approx 0\ \forall i \ne 1$, then the algorithm just takes $O(\frac{1}{k\Delta_i^2})\ln\frac{1}{\delta}$ time to terminate. On the other hand, if $1 = \theta_1 > \theta_i \approx 1$, then $\Theta_{[k]} = \Omega(k)$ which gives the worst case orderwise complexity.

Proof sketch The proof of Thm. 3 is based on the following claims:

Claim-1: At any sub-phase $s = 1,2,\ldots$, the Best-Item a^* is likely to beat the (ϵ_s,δ_s)-PAC item b_s by sufficiently high margin with probability at least $(1-\frac{\delta}{20})$, and hence is never discarded (Lem. 19).

Claim-2: Let $[n]_r := \{i \in [n] : \frac{1}{2^r} \le \Delta_i < \frac{1}{2^{r-1}}\}$, and we denote the set of surviving arms in $[n]_r$ at s^{th} sub-phase by $\mathcal{A}_{r,s}$, i.e. $\mathcal{A}_{r,s} = [n]_r \cap \mathcal{A}_s$, for any $s = 1,2,\ldots$. Then with probability at least $(1-\frac{19\delta}{20})$, any such set $\mathcal{A}_{r,s}$ reduces at a constant rate once $s \ge r$, $r = 1,\ldots,\log_2(\Delta_{\min})$ (Lem. 20)—this ensures that all suboptimal elements get eventually discarded after they are played sufficiently often.

Claim-3: The number of occurrences of any sub-optimal item $i \in [n]\setminus\{1\}$ before it gets discarded away is proportional to $O\Big(\frac{1}{\Delta_i^2}\ln\frac{k}{\delta}\Big)$. Combining this over all arms yields the desired sample complexity. Details of the proof is given in Appendix A.4. □.

3.2. An algorithm for general $\epsilon > 0$

It is straightforward to extend the $(0,\delta)$-PAC guarantee for *PAC-Wrapper* to get a more general (ϵ,δ)-PAC algorithm for any given $\epsilon \in [0,1]$. The idea is to simply execute the algorithm as originally specified until (and if) it reaches a phase s such that ϵ_s falls below the given tolerance ϵ (i.e. $\epsilon_s \le \epsilon$), at which point the algorithm can stop right after calling the subroutine (ϵ,δ)-*PAC Best-Item* and output the item b_s returned by it. The full algorithm is given in Appendix A.5 for the sake of brevity.

Theorem 4 (*PAC-Wrapper* (ϵ,δ)-PAC sample complexity bound with Winner feedback)**.** *For any* $\epsilon \in [0,1]$*, with probability at least* $(1-\delta)$*,* $\mathcal{A}$ *as PAC-Wrapper (Algorithm 1) returns the* ϵ*-Best-Item (see Defn. 2) with sample complexity* $N_{\mathcal{A}}(\epsilon,\delta) = O\bigg(\frac{\Theta_{[k]}}{k}\sum_{i=2}^n \max\Big(1,\frac{1}{\max(\Delta_i,\epsilon)^2}\Big)\ln\frac{k}{\delta}\Big(\ln\frac{1}{\max(\Delta_i,\epsilon)}\Big)\bigg)$.

Discussion. To the best of our knowledge, this is the first (ϵ,δ)-PAC learning algorithm for the Plackett-Luce model with general multi-wise comparisons with an *item-wise* instance-dependent sample complexity bound. For $\epsilon > 0$, this is order-wise stronger than the best known worst-case (instance-independent) upper bound of $O\left(\frac{n}{\epsilon^2}\log\left(\frac{k}{\delta}\right)\right)$ (Saha & Gopalan, 2019), since $\max(\Delta_i,\epsilon)^2 \ge \epsilon^2$. Thus *PAC-Wrapper* is provably able to adapt to the hardness of the Plackett-Luce instance $\boldsymbol{\theta}$ to stop early in case the instance is 'well-separated'. Note that for dueling bandits ($k = 2$), our result strictly improves order-wise upon the $\tilde{O}\left(n \cdot \max_{i\ge 2}\frac{1}{\max(\Delta_i,\epsilon)^2}\right)$ sample complexity[2] of the best known (ϵ,δ)-PAC algorithm (PLPAC) (Szörényi et al., 2015)—which can be worse by a factor of n for many instances. For example, consider an instance having one 'strong' suboptimal item, say $j \in [n]\setminus\{1\}$ with $\Delta_j \approx 0$, but $\Omega(n)$ many extremely 'weak' items with $\Delta_i \approx 1$; our

[2]Notation $\tilde{O}(\cdot)$ hides polylogarithmic factors in $\epsilon,\delta,\Delta_i,n,k$.

sample complexity bound is just $\tilde{O}\Big(\frac{1}{2\Delta_j^2}\ln\frac{1}{\delta} + \frac{n}{2}\ln\frac{1}{\delta}\Big)$, whereas that of PLPAC is $O\Big(\frac{n}{\Delta_j^2}\ln\frac{n}{\Delta_j\delta}\Big)$.

3.3. PAC learning in the Plackett-Luce model with Top-m Ranking feedback

Main Idea. Algorithmically, the key modification to make is in the *Rank-Breaking* subroutine of *PAC-Wrapper*, which now uses a rank-ordered list of m feedback items to output all possible rank-broken comparison pairs. The essence of the $\frac{1}{m}$ factor improvement in the sample complexity over Winner feedback lies in the fact that this naturally gives rise to $O(m)$ times additional number of pairwise preferences in comparison to Winner feedback. Hence, it turns out to be sufficient to sample any batch $\mathcal{B}_b, \forall b \in [B_s]$ for only $O\big(\frac{1}{m}\big)$ times compared to the earlier case, which finally leads to the improved sample complexity of *PAC-Wrapper* for Top-m Ranking feedback. The full description of Alg. 7 is given in Appendix A.7 for the sake of brevity.

Theorem 5 (*PAC-Wrapper*: Sample Complexity for $(0,\delta)$-PAC Guarantee for Top-m Ranking feedback)**.** *With probability at least $(1-\delta)$, PAC-Wrapper (Algorithm 1) returns the Best-Item with sample complexity $N_{\mathcal{A}}(0,\delta) = O\Bigg(\frac{\Theta_{[k]}}{k}\sum_{i=2}^n \max\Big(1, \frac{1}{m\Delta_i^2}\Big)\ln\frac{k}{\delta}\Big(\ln\frac{1}{\Delta_i}\Big)\Bigg)$.*

Remark 2. Following the similar procedure as argued in Sec. 3.4, one can easily derive an (ϵ,δ)-PAC version of *PAC-Wrapper (for Top-m Ranking feedback)* as well, and a similar guarantee as that of Thm. 4 with a reduction factor $1/m$. We omit the details in the interest of space.

3.4. (ϵ,δ)-PAC subroutine (used in the main algorithm, *PAC-Wrapper*, i.e. in Alg. 1, 5 or 7)

We briefly describe here the core (ϵ,δ)-PAC subroutine used in algorithms 1 and 7 to find an ϵ Best-Item with high probability $(1-\delta)$ in an instance-dependent way (full details are available in Appendix A.3): The algorithm (ϵ,δ)*-PAC Best-Item* first divides the set of n items into batches of size k, then plays each group sufficiently long enough until a single item of that group stands out as the empirical winner in terms of its empirical pairwise advantage over the rest (again estimated though *Rank-Breaking*). It then just retains this empirical winner for every group and recurses on the set of surviving winners until only a single item is left, which is declared as the (ϵ,δ)-PAC item.

Theorem 6 ((ϵ,δ)*-PAC Best-Item*: Correctness and Sample Complexity with Top-m Ranking feedback)**.** *For any $\epsilon \in \big(0, \frac{1}{8}\big]$ and $\delta \in (0,1)$, with probability at least $(1-\delta)$, (ϵ,δ)-PAC Best-Item (Algorithm 5) returns an item $b_s \in [n]$ satisfying $p_{b_s 1} > \frac{1}{2} - \epsilon$ with sample complexity $O\Big(\frac{n\Theta_{[k]}}{k}\max\big(1, \frac{1}{m\epsilon^2}\big)\log\frac{k}{\delta}\Big)$, where $\Theta_{[k]} := \max_{S\subseteq[n],|S|=k}\sum_{i\in S}\theta_i$.*

Remark 3. The best item-finding subroutine we develop, along with the corresponding analysis, is an improvement over Alg. 6 of (Saha & Gopalan, 2019) which had k instead of ${\Theta_{[k]}}_{[k]} \le k$ here. The improvement is especially pronounced for instances where $\Theta_{[k]} = O(1)$ (e.g. where $\theta_{a^*} \to 1$ and for all $i \in [n]\setminus\{a^*\}$, $\theta_i \to 0$ etc.). Note that this is an artefact of the *adaptive nature* of our proposed algorithm (Alg. 5) which samples each batch adaptively for just sufficiently enough times before discarding out the weakest $(k-1)$ items (see Line 11), whereas (Saha & Gopalan, 2019) sample each batch for a fixed $O\Big(\frac{k}{\epsilon^2}\ln\frac{k}{\delta}\Big)$ times irrespective of the empirical outcomes, leading to a *worse, instance independent sample complexity*.

4. Instance-dependent lower bounds on sample complexity

We here derive information-theoretic lower bounds on sample complexity for *Probably-Correct-Best-Item* problem. We first show a lower bound of $\Omega\Big(\sum_{i=2}^n \frac{\theta_i\theta_1}{\Delta_i^2}\ln\big(\frac{1}{\delta} + \frac{n}{k}\ln\frac{1}{\delta}\big)\Big)$ with Winner feedback implying that the sample complexity of *PAC-Wrapper* (Thm. 3) is tight upto logarithmic factors. We then analyze the lower bound for Top-m Ranking feedback and show an $\frac{1}{m}$-factor improvement in the sample complexity lower bound, establishing the optimality (up to logarithmic factors) of our *PAC-Wrapper* algorithm for Top-m Ranking feedback (see Alg. 7 and Thm. 5).

4.1. Lower bound for Winner feedback

Theorem 7 (Sample complexity lower bound: $(0,\delta)$-PAC or *Probably-Correct-Best-Item* with Winner feedback)**.** *Given $\delta \in [0,1]$, suppose $\mathcal{A}$ is an online learning algorithm for Winner feedback which, when run on any Plackett-Luce instance, terminates in finite time almost surely, returning an item I satisfying $Pr(\theta_I = \max_i \theta_i) > 1-\delta$. Then, on any Plackett-Luce instance $\theta_1 > \max_{i\ge 2}\theta_i$, the expected number of rounds it takes to terminate is $\Omega\Big(\max\Big(\sum_{i=2}^n \frac{\theta_i\theta_1}{\Delta_i^2}\ln\frac{1}{\delta}, \frac{n}{k}\ln\frac{1}{\delta}\Big)\Big)$.*

Proof sketch. We employ the measure-change technique of Kaufmann et al (Kaufmann et al., 2016) (see Lem. 26, Appendix) for lower bounds on the PAC sample complexity for standard multiarmed bandits (MAB). The novelty of our proof lies in mapping their result to our setting: For our case each MAB instance corresponds to an instance of the BB-PL problem with the arm set containing all subsets of $[n]$ of size k: $A = \{S = (S(1), \ldots S(k)) \subseteq [n]\}$.

We now consider any general true PL$(n,\boldsymbol{\theta})$ problem instance PL$(n,\boldsymbol{\theta}^1) : \theta_1^1 > \theta_2^1 \ge \ldots \ge \theta_n^1$, and corresponding to each suboptimal item $a \in [n] \setminus \{1\}$, we define an alternative problem instance $\text{PL}(n, \boldsymbol{\theta}^a) : \theta_a^a = \theta_1^1 + \epsilon;\ \theta_i^a = \theta_i^1,\ \forall i \in [n] \setminus \{a\}$, for some $\epsilon > 0$. Then, applying Lemma 26 on every pairs of problem instances $(\boldsymbol{\theta}^1, \boldsymbol{\theta}^a)$, and suitably upper bounding the KL-divergence terms we arrive at $n-1$ constraints of the form:

$$\ln \frac{1}{2.4\delta} \le \sum_{S \in A \mid a \in S} \mathbf{E}_{\boldsymbol{\theta}^1}[N_S(\tau_A)] KL(p_S^1, p_S^a)$$

$$\le \sum_{S \in A \mid a \in S} \mathbf{E}_{\boldsymbol{\theta}^1}[N_S(\tau_A)] \frac{\Delta_a'^2}{\theta_S^1(\theta_1^1 + \epsilon)},\ \forall a \in [n] \setminus \{1\}$$

Since the total sample complexity of $\mathcal{A}$ being $\mathcal{N}(0,\delta) = \sum_{S \in A} N_S$ (here N_S is the number of plays of subset S by $\mathcal{A}$), the problem of finding the sample complexity lower bound actually reduces to solving the (primal) linear programming (LP) problem:

$$\textbf{Primal LP (P):}\ \min_{S \in A} \sum_{S \in A} \mathbf{E}_{\boldsymbol{\theta}^1}[N_S]$$

$$\text{such that} \sum_{S \in A \mid a \in S} \mathbf{E}_{\boldsymbol{\theta}^1}[N_S] \frac{\Delta_a'^2}{\theta_S^1(\theta_1^1 + \epsilon)} \ge \ln \frac{1}{2.4\delta},$$

$$\forall a \in [n] \setminus \{1\}$$

However above has $O\binom{n}{k}$ many optimization variables (precisely $\mathbf{E}_{\boldsymbol{\theta}^1}[N_S]$s), so we instead solve the *dual* LP to reach the desired bound. Lastly the $\Omega\left(\frac{n}{k} \ln \frac{1}{\delta}\right)$ term in the lower bound arises as any learning algorithm must at least test each item a constant number of times via k-wise subset plays before judging it optimality which is the bare minimum sample complexity the learner has to incur (Chen et al., 2018). The complete proof is given in Appendix B.1. $\square$.

4.2. Lower bound for Top-m Ranking feedback

Theorem 8 (Sample complexity Lower Bound: $(0,\delta)$-*Probably-Correct-Best-Item* with Top-m Ranking feedback). *Suppose $\mathcal{A}$ is an online learning algorithm for Top-m Ranking feedback which, given $\delta \in [0,1]$ and run on any Plackett-Luce instance, terminates in finite time almost surely, returning an item I satisfying $Pr(\theta_I = \max_i \theta_i) > 1 - \delta$. Then, on any Plackett-Luce instance $\theta_1 > \max_{i \ge 2} \theta_i$, the expected number of rounds it takes to terminate is* $\Omega\left(\max\left(\frac{1}{m} \sum_{i=2}^n \frac{\theta_i \theta_1}{\Delta_i^2} \ln\left(\frac{1}{\delta}\right), \frac{n}{k} \ln \frac{1}{\delta} \right) \right)$.

Proof sketch. The crucial fact used here is owning to the chain rule for KL-divergence, the KL divergence for Top-m Ranking feedback is m times larger than that of just with Winner feedback: $KL(p_S^1, p_S^a) = KL(p_S^1(\sigma_1), p_S^a(\sigma_1)) + + \sum_{i=2}^m KL(p_S^1(\sigma_i \mid \sigma(1:i-1)), p_S^a(\sigma_i \mid \sigma(1:i-1)))$, where we abbreviate $\sigma(i)$ as σ_i and $KL(P(Y \mid X), Q(Y \mid X)) := \sum_x Pr\Big(X = x\Big)\Big[KL(P(Y \mid X = x), Q(Y \mid X = x))\Big]$ denotes the conditional KL-divergence. Using this and the upper bound on the KL divergences for Winner feedback setup as derived for Thm. 7, we get that in this case $KL(p_S^1, p_S^a) \le \frac{m\Delta_a'^2}{\theta_S^1(\theta_1^1+\epsilon)}, \forall a \in [n] \setminus \{1\}$, where lies the crux of the $\frac{1}{m}$-factor improvement in the sample complexity lower bound compared to Winner feedback. The lower bound now can be derived following a similar procedure that of Thm. 7. Details are given in B.1. $\square$.

5. The *Fixed-Sample-Complexity* Learning Problem

This section studies the problem of finding the Best-Item within a maximum allowed number of queries Q, with minimum possible probability of misidentification. Note the algorithms for *Probably-Correct-Best-Item* setting cannot be used here as they do not take the total sample complexity Q as input; also, simply terminating such algorithms with a suitable δ after Q runs may not necessarily be optimal. We present results for the general Top-m Ranking feedback.

5.1. Lower Bound: *Fixed-Sample-Complexity* setting

We derive an instance-dependent lower bound on error probability in which the problem complexity depends on the complexity term $\left(\sum_{a=2}^n \frac{\theta_a}{\Delta_a^2}\right)^{-1}$, unlike the case for our first objective (*Probably-Correct-Best-Item*), which depends on the gap parameter $\frac{1}{\Delta_a^2}$, $\forall \in [n] \setminus \{1\}$. We first define a natural consistency or 'non-trivial learning' property for any best-arm algorithm given a fixed budget of Q:

Definition 9 (*Budget-Consistent* Best-Item Identification Algorithm). *An online algorithm $\mathcal{A}$, taking as input a sample complexity budget Q, terminating within Q rounds and outputting an item $I \in [n]$, is said to be Budget-Consistent if, for every Plackett-Luce instance $\boldsymbol{\theta} \equiv (\theta_i)_{i=1}^n$ with a unique best item $a^*(\boldsymbol{\theta}) := \arg\max_{i \in [n]} \theta_i$, it satisfies $Pr_{\boldsymbol{\theta}}\big(I = a^*(\boldsymbol{\theta})\big) \ge 1 - \exp(-f(\boldsymbol{\theta}) \cdot Q)$ when run on $\boldsymbol{\theta}$, where $f : [0,1]^n \mapsto \mathbb{R}_+$ is an instance-dependent function mapping every Plackett-Luce instance to a real number.*

Informally, a *Budget-Consistent* algorithm picks out the best arm in a Plackett-Luce instance with arbitrarily low error probability given enough rounds Q. We next define the notion of a *Order-Oblivious* or *label-invariant* algorithm before stating our main lower bound result.

Definition 10 (Order obliviousness or label invariance). *A Budget-Consistent algorithm $\mathcal{A}$ is said to be Order-Oblivious if its output is insensitive to the specific labelling of items, i.e., if for any PL model $(\theta_1, \ldots, \theta_n)$, bijection $\phi : [n] \to [n]$ and any item $I \in [n]$, it holds that $Pr(\mathcal{A}\ \text{outputs}\ I \mid (\theta_1, \ldots, \theta_n)) = Pr(\mathcal{A}\ \text{outputs}\ I \mid (\theta_{\phi(1)}, \ldots, \theta_{\phi(n)}))$, where $Pr(\cdot \mid (\alpha_1, \ldots, \alpha_n))$ denotes the probability distribution on the trajectory of $\mathcal{A}$ induced by the PL model*

$(\alpha_1, \ldots, \alpha_n)$.

Theorem 11 (Confidence lower bound in fixed sample complexity Q for Top-m Ranking feedback)**.** *Let $\mathcal{A}$ be a Budget-Consistent and Order-Oblivious algorithm for identifying the Best-Item under Top-m Ranking feedback. For any Plackett-Luce instance $\boldsymbol\theta$ and sample size (budget) Q, its probability of error in identifying the best arm in $\boldsymbol\theta$ satisfies $Pr_{\boldsymbol\theta}\left(I \neq \arg\max_{i\in[n]} \theta_i\right) = \Omega\left(\exp\left(-2mQ\tilde\Delta\right)\right)$, where the complexity parameter $\tilde\Delta := \left(\sum_{a=2}^n \frac{(\theta_a)^2}{\Delta_a^2}\right)^{-1}$.*

Remark 4. As expected, the error probability reduces with increasing feedback size m and budget Q. However a more interesting tradeoff lies in the instant dependent complexity term $\tilde\Delta$: for 'easy' instances where most of the suboptimal item have $\theta_a \to 0$ (i.e. $\Delta_a \to 1$), $\tilde\Delta$ shoots up, in fact attains $\tilde\Delta \to \infty$ in the limiting case where $\theta_a \to 0\, \forall i \in [n \setminus \{1\}$. On the other hand, for 'hard' instances, where there exists even one suboptimal item $a \in [n] \setminus \{1\}$ with $\theta_a \approx 1$ (i.e. $\Delta_a \approx 0$), $\tilde\Delta \to 0$ raising the minimum error probability significantly, which indicates the hardness of the learning problem.

5.2. Proposed Algorithm for *Fixed-Sample-Complexity* setup: *Uniform-Allocation*

Main Idea. Our proposed algorithm *Uniform-Allocation* solves the problem with a uniform budget allocation rule: Since we are allowed to play sets of size k only, we divide the items into k-sized batches and eliminate the bottom half of the winning items once each batch is played sufficiently. The important parameter to tune is how long to play the batches. Given a fixed budget Q, since one does not have an idea about which batch the Best-Item lies in, a good strategy is to allocate the budget uniformly across all sets formed during the entire run of the algorithm, which can shown to be precisely $O(\frac{n+k\log_2 k}{k})$ sets, so we allocate a budget of $Q' = O\left(\frac{kQ}{n+k\log_2 k}\right)$ samples per batch.

Algorithm description. The algorithm proceeds in rounds, where in each round it divides the set of surviving items into batches of size k and plays each $Q' = \frac{(n+k)kQ}{2n^2\log_2 k}$ times. Upon this it retains only the top half of the winning arms, eliminating the rest forever. The hope here is that with 'enough' observed samples, the Best-Item always stays in the top half and never gets eliminated. The next round recurses on the remaining items, and the algorithm finally returns the only single element is left as the potential Best-Item. The pseudocode is moved to Appendix C.2.

Theorem 12 (*Uniform-Allocation*: Confidence bound for Best-Item identification with fixed sample complexity Q)**.** *Given a budget of Q rounds, Uniform-Allocation returns the Best-Item of PL$(n, \boldsymbol\theta)$ with probability at least $1 - O\left(\log_2 n \exp\left(-\frac{mQ\Delta_{\min}^2}{16(2n+k\log_2 k)}\right)\right)$, where $\Delta_{\min} = \min_{i=2}^n \Delta_i$.*

Remark 5. Thm. 12 equivalently shows that with sample complexity at most $O\left(\frac{16(2n+k\log_2 k)}{m\Delta_{\min}^2}\ln\left(\frac{\log_2 n}{\delta}\right)\right)$, *Uniform-Allocation* returns the Best-Item with probability at least $(1-\delta)$. The bound is clearly optimal in terms of m and Q (comparing with Thm. 11), however it still remains an open problem to close the gap between the complexity term $\tilde\Delta = \left(\sum_{a=2}^n \frac{(\theta_a)^2}{\Delta_a^2}\right)^{-1}$ in the lower bound, vs. the $\left(\frac{n}{\Delta_{\min}^2}\right)^{-1}$ term that we obtained.

6. Experiments

This section reports numerical results of our proposed algorithm *PAC-Wrapper* (PW) on different Plackett-Luce environments. All reported performances are averaged across 50 runs. The default values of the parameters are set to be $k = 5$, $\epsilon = 0.01$, $\delta = 0.01$, $m = 1$ unless explicitly mentioned/tuned in the specific experimental setup. We compared our algorithm with the only existing benchmark algorithm *Divide-and-Battle* (DnB) (Saha & Gopalan, 2019) (even though, as described earlier, it does not apply to instance-optimal analysis, specifically for $\epsilon = 0$; this is reflected in our experimental results as well). We use 8 different PL environments (with different $\boldsymbol\theta$ parameters) for the purpose, their descriptions are moved to Appendix D.

Throughout this section, by the term *sample-complexity*, we mean the average (mean) termination time of the algorithms across multiple reruns (i.e. number of subsetwise queries performed by the algorithm before termination).

6.1. Results: *Probably-Correct-Best-Item* setting

Sample-Complexity vs Error-Margin (ϵ)**.** Our first set of experiments analyses the sample complexity ($\mathcal{N}(\epsilon, \Delta)$) of *PAC-Wrapper* with varying ϵ (keeping δ fixed at 0.1). As expected, Fig. 2 shows that the sample complexity increases with decreasing ϵ for both the algorithms. However, the interesting part is, for PW the sample complexity becomes almost constant beyond a certain threshold of ϵ (precisely when ϵ falls below $\Delta_{\min}$) in every case, whereas for DnB it keeps on scaling in $O(\frac{1}{\epsilon^2})$ irrespective of the 'hardness' of the underlying PL environment due to its non-adaptive nature—this is the region where we excel out. Also, note that the harder the dataset (i.e. the smaller its $\Delta_{\min}$), the smaller this threshold is, as follows from Thm. 4, which verifies the instance-adaptive nature of our PW algorithm as it terminates as soon as ϵ falls below $\Delta_{\min}$.

Itemwise sample complexity. This experiment reveals the survival time of the items (i.e. total number plays of an item before elimination) in *PAC-Wrapper* algorithm. The results in Fig. 3 clearly shows the inverse dependency of

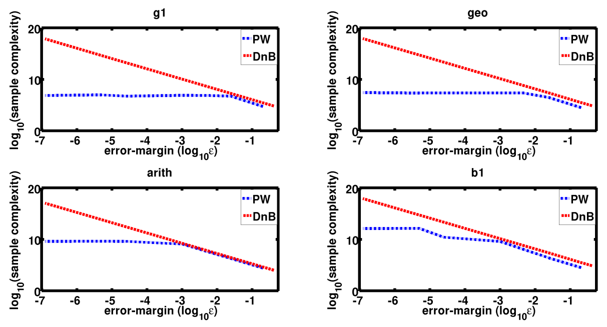

Figure 2. Sample-Complexity vs Error-Margin (ϵ) (both in $\log$ scale) of PW and DnB across 4 different problem instances.

the survival time of items w.r.t. their θ parameter, e.g. for **g4** dataset, the survival times of the items are categorized into 4 groups, highest for item 1, with items 2-6, 7-11, and 12-16 following it—justifying the $O\big(\frac{1}{\Delta_i^2}\big)$ survival times for each item i (in Thm. 3 or 5).

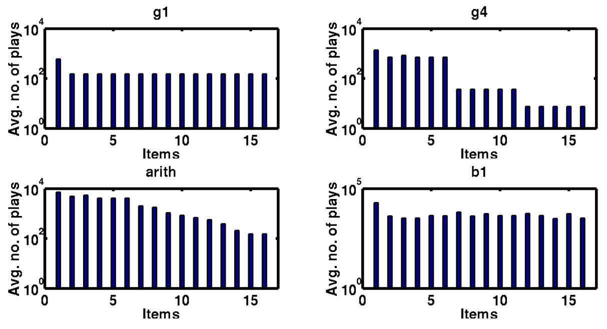

Figure 3. Survival time of different items (Itemwise sample complexity) in PW on 4 different problem instances.

Tradeoff: Sample-Complexity vs size of Top-ranking Feedback m**.** In this case we verified the flexibility of *PAC-Wrapper* for Top-m Ranking feedback (Alg. 7). We run it on different datasets with increasing size of top-ranking feedback (m). Again, justifying the claims of Thm. 5, Fig. 4 shows the sample complexity varies at a rate of $\frac{1}{m}$ (note that as m is doubled, sample complexity gets about halved), while rest of the parameters (i.e. k, δ, ϵ) are kept unchanged.

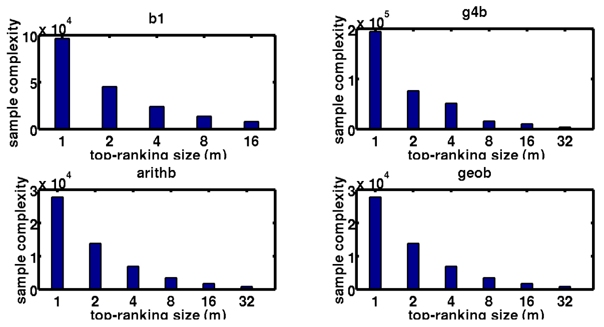

Figure 4. Sample-Complexity vs length of rank ordered feedback (m) of PW for 4 different problem instances.

6.2. Results: *Fixed-Sample-Complexity* setting

Success probability $(1-\delta)$ **vs Sample-Complexity** (Q)**.** Finally we analysed the success probability $(1-\delta)$ of algorithm *Uniform-Allocation* (UA) for varying sample complexities (Q), keeping ϵ fixed at $(\Delta_{\min})/2$. Fig. 5 shows that the algorithm identifies the Best-Item with higher confidence with increasing Q—justifying its $O(\exp(-Q)$ error confidence rate as proved in Thm. 12. Note that **g4** being the easiest instance, it reaches the maximum success rate 1 at a much smaller Q, compared to the rest. By construction, DnB is not designed to operate in *Fixed-Sample-Complexity* setup, but due to lack of any other existing baseline, we still use it for comparison force terminating it if the specified sample complexity is exceeded, and as expected, here again it performs poorly in the lower sample complexity region.

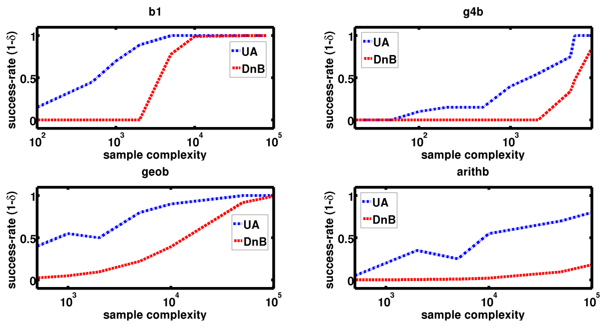

Figure 5. Comparative performances of PW and DnB in terms of Success probability $(1-\delta)$ vs Sample-Complexity (Q) across 4 different problem instances.

7. Conclusion and Future Work

Moving forward, it would be interesting to explore similar algorithmic and statistical questions in the context of other common subset choice models such as the Mallows model, Multinomial Probit, etc. It would also be of great practical interest to develop efficient algorithms for large item sets, especially when there is structure among the parameters to be exploited. One can also aim to develop instant dependent guarantees for other 'learning from relative feedback' objectives, e.g. PAC-ranking (Szörényi et al., 2015), top-set identification (Busa-Fekete et al., 2013) etc., both in fixed confidence as well as fixed budget setting.

Acknowledgements

The authors are grateful to Praneeth Netrapalli and the anonymous reviewers for valuable feedback. This work is supported by a Qualcomm Innovation Fellowship 2019-20, and the Indigenous 5G Test Bed project grant from the Dept. of Telecommunications, Government of India.

References

Audibert, J.-Y. and Bubeck, S. Best arm identification in multi-armed bandits. In *COLT-23th Conference on Learning Theory-2010*, pp. 13–p, 2010.

Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Braverman, M. and Mossel, E. Noisy sorting without resampling. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 268–276. Society for Industrial and Applied Mathematics, 2008.

Brost, B., Seldin, Y., Cox, I. J., and Lioma, C. Multi-dueling bandits and their application to online ranker evaluation. *CoRR*, abs/1608.06253, 2016.

Busa-Fekete, R., Szorenyi, B., Cheng, W., Weng, P., and Hüllermeier, E. Top-k selection based on adaptive sampling of noisy preferences. In *International Conference on Machine Learning*, pp. 1094–1102, 2013.

Busa-Fekete, R., Hüllermeier, E., and Szörényi, B. Preference-based rank elicitation using statistical models: The case of mallows. In *Proceedings of The 31st International Conference on Machine Learning*, volume 32, 2014a.

Busa-Fekete, R., Szörényi, B., and Hüllermeier, E. Pac rank elicitation through adaptive sampling of stochastic pairwise preferences. In *AAAI*, pp. 1701–1707, 2014b.

Caragiannis, I., Procaccia, A. D., and Shah, N. When do noisy votes reveal the truth? In *Proceedings of the fourteenth ACM conference on Electronic commerce*, pp. 143–160. ACM, 2013.

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

Chen, X., Gopi, S., Mao, J., and Schneider, J. Competitive analysis of the top-k ranking problem. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1245–1264. SIAM, 2017.

Chen, X., Li, Y., and Mao, J. A nearly instance optimal algorithm for top-k ranking under the multinomial logit model. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 2504–2522. SIAM, 2018.

Even-Dar, E., Mannor, S., and Mansour, Y. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of machine learning research*, 7(Jun):1079–1105, 2006.

Falahatgar, M., Hao, Y., Orlitsky, A., Pichapati, V., and Ravindrakumar, V. Maxing and ranking with few assumptions. In *Advances in Neural Information Processing Systems*, pp. 7063–7073, 2017.

Freund, Y. and Schapire, R. E. Game theory, on-line prediction and boosting. In *COLT*, volume 96, pp. 325–332. Citeseer, 1996.

Graepel, T. and Herbrich, R. Ranking and matchmaking. *Game Developer Magazine*, 25:34, 2006.

Hofmann, K. et al. Fast and reliable online learning to rank for information retrieval. In *SIGIR Forum*, volume 47, pp. 140, 2013.

Jamieson, K., Malloy, M., Nowak, R., and Bubeck, S. lil' ucb : An optimal exploration algorithm for multi-armed bandits. In Balcan, M. F., Feldman, V., and Szepesvari, C. (eds.), *Proceedings of The 27th Conference on Learning Theory*, volume 35 of *Proceedings of Machine Learning Research*, pp. 423–439. PMLR, 2014.

Jang, M., Kim, S., Suh, C., and Oh, S. Optimal sample complexity of m-wise data for top-k ranking. In *Advances in Neural Information Processing Systems*, pp. 1685–1695, 2017.

Kalyanakrishnan, S., Tewari, A., Auer, P., and Stone, P. Pac subset selection in stochastic multi-armed bandits. In *ICML*, volume 12, pp. 655–662, 2012.

Karnin, Z., Koren, T., and Somekh, O. Almost optimal exploration in multi-armed bandits. In *International Conference on Machine Learning*, pp. 1238–1246, 2013.

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

Khetan, A. and Oh, S. Data-driven rank breaking for efficient rank aggregation. *Journal of Machine Learning Research*, 17(193):1–54, 2016.

Mohajer, S., Suh, C., and Elmahdy, A. Active learning for top-k rank aggregation from noisy comparisons. In

International Conference on Machine Learning, pp. 2488–2497, 2017.

Popescu, P. G., Dragomir, S., Slusanschi, E. I., and Stanasila, O. N. Bounds for Kullback-Leibler divergence. *Electronic Journal of Differential Equations*, 2016, 2016.

Radlinski, F., Kurup, M., and Joachims, T. How does clickthrough data reflect retrieval quality? In *Proceedings of the 17th ACM conference on Information and knowledge management*, pp. 43–52. ACM, 2008.

Ren, W., Liu, J., and Shroff, N. B. Pac ranking from pairwise and listwise queries: Lower bounds and upper bounds. *arXiv preprint arXiv:1806.02970*, 2018.

Saha, A. and Gopalan, A. Battle of bandits. In *Uncertainty in Artificial Intelligence*, 2018a.

Saha, A. and Gopalan, A. Active ranking with subset-wise preferences. *arXiv preprint arXiv:1810.10321*, 2018b.

Saha, A. and Gopalan, A. PAC Battling Bandits in the Plackett-Luce Model. In *Algorithmic Learning Theory*, pp. 700–737, 2019.

Soufiani, H. A., Parkes, D. C., and Xia, L. Computing parametric ranking models via rank-breaking. In *ICML*, pp. 360–368, 2014.

Sui, Y., Zhuang, V., Burdick, J. W., and Yue, Y. Multi-dueling bandits with dependent arms. *arXiv preprint arXiv:1705.00253*, 2017.

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

Urvoy, T., Clerot, F., Féraud, R., and Naamane, S. Generic exploration and k-armed voting bandits. In *International Conference on Machine Learning*, pp. 91–99, 2013.

Yue, Y. and Joachims, T. Beat the mean bandit. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 241–248, 2011.

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