Offline Evaluation via Human Preference Judgments: A Dueling Bandits Problem

by

Xinyi Yan

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2022

Statement of Contributions

This thesis includes the material which is presented at SIGIR 2022 [87]. Chapter 5 includes text and analysis by my supervisor Prof. Charles Clarke for publication in that paper. Chengxi Luo deployed and maintained the server infrastructure for the crowdsourcing experiments. Prof. Charles Clarke and Prof. Mark Smucker provided editing and supervisory oversight throughout the thesis. I am the sole author of the remaining chapters.

Abstract

The dramatic improvements in core information retrieval tasks engendered by neural rankers create a need for novel evaluation methods. If every ranker returns highly relevant items in the top ranks, it becomes difficult to recognize meaningful differences between them and to build reusable test collections. Several recent papers explore pairwise preference judgments as an alternative to traditional graded relevance assessments. Rather than viewing items one at a time, assessors view items side-by-side and indicate the one that provides the better response to a query, allowing fine-grained distinctions. If we employ preference judgments to identify the probably best items for each query, we can measure rankers by their ability to place these items as high as possible. I frame the problem of finding best items as a dueling bandits problem. While many papers explore dueling bandits for online ranker evaluation via interleaving, they have not been considered as a framework for offline evaluation via human preference judgments. I review the literature for possible solutions. For human preference judgments, any usable algorithm must tolerate ties since two items may appear nearly equal to assessors. It must minimize the number of judgments required for any specific pair, since each such comparison requires an independent assessor. Since the theoretical guarantees provided by most algorithms depend on assumptions that are not satisfied by human preference judgments, I simulate selected algorithms on representative test cases to provide insight into their practical utility. In contrast to the previous paper presented at SIGIR 2022 [87], I include more theoretical analysis and experimental results in this work. Based on the simulations, two algorithms stand out for their potential. I proceed with the method of Clarke et al. [20], and the simulations suggest modifications to further improve its performance. Using the modified algorithm, over 10,000 preference judgments for pools derived from submissions to the TREC 2021 Deep Learning Track are collected, confirming its suitability. We test the idea of best-item evaluation and suggest ideas for further theoretical and practical progress.

Acknowledgements

I would like to thank my supervisors Prof. Charles L.A. Clarke and Prof. Mark D. Smucker who constantly guided and supported me throughout the course of my graduate studies, with and beyond my research. I would like to thank the university and the faculties for creating a positive and supportive environment for students to pursue their academic and career success.

I would like to thank my parents for their unconditional love, support, and encouragement. Without them, I could never have come this far. I thank all the people who made a difference in my life. They made me all I am today.

Table of Contents

List of Figures

Chapter 1

Introduction

The standard offline evaluation approach for ad hoc information retrieval systems requires a benchmark document collection, a test suite of queries, and a set of relevance judgments. These judgments are usually made by human assessors based on predefined relevance levels to measure the degree to which a search result is related to a query. Collecting relevance judgments is time-consuming and expensive since it involves human assessors. Especially for large document collections, it is intractable to obtain exhaustive relevance judgments for every single query and document pair. To reduce assessment costs, the Text REtrieval Conference (TREC) project adopts the pooling [74] method which only assesses the top k documents returned by various retrieval systems. However, due to the huge improvements in information retrieval systems and the continuously increasing document corpus size, recent work [81] on the TREC Deep Learning track has found that the document collection built with pooling is not reusable, and has the problem of high-precision saturation because of the existence of too many relevant documents in the judging pool and likely many unknown relevant documents in the unjudged portion of the corpus.

Other evaluation challenges associated with pointwise graded relevance assessments also note the existence of too many relevant documents and recognize difficulties in clearly defining and consistently applying precise relevance grading schemes [14]. If every retriever returns highly relevant items in the top ranks, it becomes difficult to distinguish among retrievers. All of these challenges create a need for novel evaluation methods. In this work, I consider best item evaluation as a potential direction. If we employ preference judgments to identify the probably best items for each query, we can measure rankers by their ability to place these best items as high as possible.

As an alternative to absolute relevance assessments of individual items, pairwise pref-

msmarco_passage_50_318366271: Nicole Jaffe (I) Nicole Jaffe. Jaffe's career was mainly based on doing voice-over work. She is best remembered as the voice of the nerdy-and-intellectual Velma Dinkley in the Scooby-Doo cartoons from 1969-1973; before actress Pat Stevens picked up the role.

msmarco_passage_30_571323592: Biography. Taliesin Jaffe born as Taliesin Axelrod Jaffe is an American actor, voice actor, ADR director, and scriptwriter. Jaffe was born on January 19, 1977. He is one of the famous director and actor who was born in Los Angeles, California, United States. His nationality is American.

msmarco_passage_24_359189946: Taliesin Jaffe The Flash. Taking the helm of the legendary Scarlett Speedster is Taliesin Jaffe, who is a video game and television actor. Jaffe has appeared in numerous titles including Fire Emblem Heroes, Final Fantasy XV, Fallout 4, Pillars of Eternity, and Street Fighter IV.

Figure 1.1: Three of 92 pooled passages for the question *Who is Jaffe?* (#1103547). Even though the question is ambiguous, the first two that provide clear and complete biographies were preferred by assessors.

erence judgments have been explored in several papers [14, 20, 39, 67, 71, 86]. Instead of viewing items one at a time and assigning predefined relevance scores, human assessors consider two items side-by-side and express a relative preference for one over the other, allowing fine-grained distinctions that may be missed by pointwise judgments [8, 20]. Assessment guidelines for pairwise preference judgments can also be simpler since judgments can be expressed in relative terms, which can ease and speed up the judging process [20].

Figure 1.1 provides an example, presenting three passages intended to answer the question "Who is Jaffe?". They all provide clear and concise biographies for two different actors with the last name “Jaffe”. Distinguishing them using absolute scores will require more complex definitions of relevance and increase the judgment burden on assessors. As a comparison, pairwise preference judgments, only requiring assessors to make relative decisions, greatly simplify the labeling process. Our experiments show that the first two passages were preferred by crowdsourced assessors since they are more focused on biographical information.

The value of preference judgments for offline evaluation has been demonstrated in recent papers. Sakai and Zeng [71] propose a variety of preference-based measures and examine the measures with users' SERP preferences and traditional measures such as nDCG. Clarke et al. [18] define and validate an evaluation measure that computes the similarity between a directed multigraph of preferences and an actual ranking generated by a ranker. Arabzadeh et al. [6] employ preference judgments to isolate best items and evaluate rankers by their ability to place best items as high as possible. Others propose and explore measures based on counts of correctly ordered items - according to the preference judgments - or measures that convert preferences to gain values for use with NDCG and other standard evaluation measures [13, 71, 86, 88].

These papers give considerable attention to preference-based evaluation measures. However, they devote relatively less attention to the preference judging process itself. Many papers assume that all pairs in a judging pool will be judged once [6, 14, 71]. Sakai and Zeng [71] judge all pairs three times, requiring thousands of judgments for a judging pool of only three or four dozen items. Carterette et al. [14] suggest reducing the number of judgments by exploiting transitivity and eliminating non-relevant items in an initial binary relevance pass, but they do not explore these ideas in detail. Clarke et al. [20] build on these ideas, describing a tournament-like judging strategy targeted at crowdsourced preference judgments.

In this thesis, I focus on the judging process, with the goal of identifying the best responses to a query while minimizing the number of preference judgments. With these best responses, we can measure rankers via best item evaluation. I frame the problem of finding best items in terms of dueling bandits, in which the algorithm attempts to identify the probably best of K arms through noisy pairwise comparisons [9, 50, 54, 92, 90, 95, 96]. There are extensive studies on dueling bandits and human preference judgments in the context of information retrieval, but explicit connections between them are rare. Glowacka [32] (Chapter 5) reviews both the use of dueling bandits for online evaluation and the use of (non-dueling) bandits for traditional pointwise relevance judgments, but does not connect the two concepts.

In the remainder of this work, I first introduce bandits algorithms together with IR evaluation measures in Chapter 2, including both traditional measures and preference-based measures. In Chapter 3, I define and justify requirements for human preferences as dueling bandits, followed by a review and analysis of candidate algorithms from both the machine learning and information retrieval literature. Since the theoretical guarantees provided by these algorithms typically depend on assumptions that are not usually satisfied by human preference judgments, I simulate selected algorithms on two representative test cases to provide insight into their practical utility in Chapter 4. In Chapter 5, a modified version of the algorithm described in Clarke et al. [20] is proposed as a suitable candidate. In a joint work, we employed the modified algorithm to collect over 10,000 crowdsourced preference judgments for the TREC 2021 Deep Learning Track [22], verifying the suitability of the algorithm. Using the collected preference judgments, we tested the idea of best item evaluation and provided support for ideas proposed by Arabzadeh et al [6]. In Chapter 6, I give a summary of the research results and suggest directions for future work.

The main contributions of this thesis are three-fold:

- Proposed the novel idea of framing the problem of finding best items via crowd-sourced preference judgments in terms of dueling bandits, and set the scene for further progress in this space.
- Reviewed candidate algorithms from the dueling bandits literature, further progressed the work in Clarke et al. [20], and outlined directions for theoretical and practical progress.
- Generated costly preference judgments and validated the idea of best item evaluation. Crowdsourced preference judgments are available publicly for further research.

Chapter 2

Background

This chapter provides the reader with background information, covering from traditional evaluation measures in information retrieval to reinforcement learning and bandit algorithms. It also looks at related work in finding the max/top-k items via crowdsourced preference judgments.

2.1 Graded Relevance

Traditional offline evaluation for information retrieval tasks typically relies on graded relevance assessments of retrieved results to measure the effectiveness of IR systems. Binary assessments classify every document as either relevant or non-relevant in response to the user request, implying that all relevant documents are equally important and have the same level of information. In a ranked retrieval context, IR systems are typically evaluated over the top-k retrieved results. Common evaluation measures for binary assessments include *precision@k*, *recall@k*, *Mean Average Precision (MAP)* and *Mean Reciprocal Rank(MRR)*.

One shortcoming of binary relevance is that it does not consider the possibility of documents being relevant to a different degree, with some documents highly relevant and others less. In order to address such issues, multi-level graded relevance judgments are employed. However, there is no globally accepted standard of relevance levels. The number of relevance levels and the definition of levels can vary from task to task. For example, judgments for the document ranking task in the TREC Deep Learning track were collected on a four-point scale of *Irrelevant, Relevant, Highly Relevant* and *Perfectly Relevant* [23]. The TREC Conversational Assistance Track [24] adopted a five-point relevance scale of *Fully meets, Highly meets, Moderately meets, Slightly meets* and *Fails to meet.*

A standard and widely adopted evaluation approach for non-binary notions of relevance is normalized discounted cumulative gain (nDCG) [41]. To compute nDCG, ordinal relevance levels must be converted into a ratio scale. For instance, the TREC Deep Learning track mapped score 0 to irrelevant and scores 1, 2, 3 to relevant, highly relevant and perfectly relevant respectively [23]. The more relevant a document is, the higher score it gets assigned.

While multi-level graded relevance has seen increasing adoption in both academia and industry, it faces several problems. Defining and consistently applying multi-level relevance grades is more difficult and time-consuming than making binary decisions, requiring a substantial amount of human effort [14]. If every retriever returns highly relevant items in the top ranks, it becomes difficult to recognize meaningful differences between retrievers and to build reusable test collections.

2.2 NDCG

Normalized discounted cumulative gain (nDCG) is an evaluation metric that measures the effectiveness of IR systems for retrieving highly relevant documents. It is based on two assumptions: 1) The more relevant a document is, the more useful information it contains. 2) Relevant documents are more valuable when appearing earlier in a ranked list, since users tend to pay more attention to top items and neglect the rest [41].

NDCG is a normalized value of the Discounted Cumulative Gain (DCG), which is a weighted sum of relevance scores of retrieved items. The discount factor dominantly used in IR literature is $\frac{1}{\log 1+r}$, where r represents the ranked position of the object. Assume that for a query q there is a collection of m documents $D = \{d_1, d_2, ..., d_m\}$ whose relevance to q is given by $rel = \{rel_1, rel_2, ..., rel_m\}$. Let *F(q, d)* be a ranking function that outputs a real-valued score for every query-document pair. Then for query q and the collection of m documents, F ranks the documents according to their scores $F(q, d_1), ..., F(q, d_m)$. Let $rank_i$ denote the rank of document d_i in the resulting ranking list. DCG value is computed as follows:

$$DCG(F, D) = \Sigma_{i=1}^{m} \frac{rel_i}{\log_2(1 + rank_i)}$$

An alternative DCG, often used by commercial search engines, gives exponentially increasing weights to higher relevance grades:

$$DCG(F, D) = \Sigma_{i=1}^{m} \frac{2^{rel_i} - 1}{\log_2(1 + rank_i)}$$

NDCG normalizes DCG on a scale of 0 to 1 by the Ideal DCG (IDCG), which is the DCG of an ideal/perfect ranking: for any pair of documents d_i and d_j in the ranking list, $rel_i \geq rel_j$ indicates $rank_i \leq rank_j$. Let $F\prime$ denote a ranking function that generates an ideal ranking of the collection D. Then IDCG is $IDCG(D) = DCG(F\prime, D)$. NDCG is computed as follows:

$$NDCG(F, D) = \frac{DCG(F, D)}{IDCG(D)}$$

NDCG has proved to be discriminative, stable, and predictive [69, 82, 83], and is one of the most standard and primary evaluation measures used in recent TREC tasks [23, 24]. However, it still has limitations. If NDCG is selected as an evaluation measure, human assessors need to make judgments by assigning scores to query-document pairs based on pre-defined relevance assessing guidelines. Such guidelines specify how to decide among multiple relevance scores based on potentially many factors. There are several concerns associated with this approach. First, anticipating all potentially useful factors is intractable when creating the guidelines, and balancing between various factors is challenging. The guidelines either can be too coarse to distinguish items from each other, or can be too fine-grained and tedious. Additionally, assessors need to take lots of mental effort to maintain consistency when making judgments using such multi-level graded relevance guidelines.

2.3 Preference Judgments

As opposed to absolute judgments assessed based on the degree of relevance of individual items, preference judgments specify the relation of two items, indicating that one result should be favored over the other. Therefore, preference judgments bypass some of the difficulties of defining and deciding between multiple relevance scores. Preferences can be derived either directly from human assessments or implicitly from existing relevance scores, click data or other signals. Throughout this thesis, I consider a preference judgment as a pairwise, binary decision made by human assessors.

The idea of using preferences as an alternative to absolute relevance dates back to the early 1990s. Research conducted by Rorvig in 1990 first introduced preference judgments and showed that test collections could be built reliably from such judgments [68]. In 1991, Frei and Schauble [31] proposed a usefulness measure on preference sets and suggested that humans are more consistent at making relative relevance judgments. In 1995, Yao [89] proposed an evaluation measure to examine the agreement between user preference of documents and system-retrieved results based on the distance between the two rankings.

When compared to pointwise absolute judgments, relative judgments are more flexible and easier to obtain, allowing for finer distinctions among documents and considering factors that can be difficult to incorporate in relevance grades [14]. For instance, if two articles are equally relevant, the more recent one may be preferred for the purpose of timeliness. However, there are two primary concerns regarding pairwise comparisons: 1) the number of pairs needed to be judged may be quadratic in the number of documents. 2) There has been a lack of validated evaluation methods.

A notable effort was made in 2008 by Carterette et al. [14] to bring preference-based measures into actual use. The authors asked expert judges to make pointwise judgments and pairwise comparisons, collecting a total of $O(n)$ absolute scores and $O(n^2)$ preferences. By studying the assessor agreement and the time spent per judgment, they found that pairwise comparisons can be made faster by humans and are less likely to have inter-assessor disagreement than pointwise judgments. In addition, on average over 99% of preference triplets are transitive: given a triplet of documents (i, j, k), if i is preferred to j and j is preferred to k, i is preferred k, suggesting that the number of pairs for judging can be reduced by exploiting transitivity [14].

As for evaluation measures for preference judgments, Carterette et al. [14] developed two simple metrics: ppref (precision of preference) and wpref (weighted precision of preference). Ppref is the percentage of pairs ordered by IR systems that agree with preferences. Wpref is the sum of pairs in ppref weighted by factor $w_{ij} = \frac{1}{\log_2 j+1}$ such that item i is preferred to item j and i, j are the ranks. Later at SIGIR 2008, Carterette and Bennett [13] introduced another two metrics: rpref (recall of preferences) and average precision of preferences. Inspired by their work to establish preference-based evaluation measures, Sakai and Zeng [71] greatly expanded the family of preference-based measures and examined whether they aligned with the users' perception of search engine result pages (SERPs). More recently, Clarke et al. [18] treated a collection of preferences as a directed multi-graph, and developed an evaluation measure that computes the similarity between a preference graph and an actual ranking generated by IR systems.

Instead of basing evaluation metrics on the counts of correctly ordered items, another way is to measure IR systems by their ability to rank best-known items as high as possible. Arabzadeh et al. [6] employed crowdsourced workers to make preference judgments and isolated only the best items which were used to re-evaluate top runs on the MS MARCO passage retrieval leaderboard.

2.4 Reinforcement Learning and Bandit Algorithms

In recent decades, the theory and practice of reinforcement learning have gained increasing attention, with many exciting advancements. Reinforcement learning is an active field in machine learning concerned with learning optimal policies for sequential decision making problems from experience gained through interacting with the environment [53, 56, 75]. As opposed to supervised learning where data is labeled or unsupervised learning where the aim typically is to discover hidden structures in unlabeled data, the learner in reinforcement learning studies what decisions in an action space yield high rewards by trying them out and evaluating reward signals. Often, different decisions not only result in different immediate rewards, but also have an impact on the next scenario and therefore all future rewards. The goal is to learn an optimal policy that maximizes the expectation of long-term return.

A distinctive feature of reinforcement learning is the exploration-exploitation tradeoff. Initially, reward distributions associated with each action are fixed but unknown, and a learner can only obtain estimates by executing respective actions. Feedback received indicates how good a selected action is, but not if it is optimal. Consequently, in order to maximize rewards, a learner needs to find a balance between actively exploring new actions to discover the one with higher long-term return and exploiting the current knowledge by selecting a greedy option to maximize short-term rewards [75]. This is known as the exploration–exploitation tradeoff.

Reinforcement learning has several sub-areas, one of which is the multi-armed bandit (MAB) problem. In the standard setting, a learner repeatedly selects one action at a time from a given set of K choices and observes the corresponding feedback, with the goal of optimizing its evaluation criterion over a time period [9, 32, 90]. Assume that there are K actions $\{a_1, a_2, ..., a_K\}$ whose associated expected rewards are $\{\mu_1, \mu_2, ..., \mu_K\}$. The K actions are referred to as the K arms of a bandit, named by analogy to a slot machine. The objective is to find a strategy to select an arm at each round $t = 1, ..., T$ and earn the corresponding reward while maximizing the cumulative rewards after T time steps. Given that the maximal reward can be obtained by always selecting the best arm, the goal of a learner can be also viewed as finding the best arm that has the highest expected reward.

Depending on what a learner is concerned with, there are various ways to measure the performance. One common evaluation criterion is cumulative regret. Let $\mu\prime = \max_k \mu_k$ denote the expected reward of selecting the best arm. Then the regret of choosing an arm a_i at round t is defined as the difference between the reward of selecting the best arm and the reward of selecting a_i:

$$r_t = \max_k \mu_k - \mu_i = \mu\prime - \mu_i$$

The total regret accumulated after T time steps is defined as:

$$R_T = \sum_{t=1}^{T} r_t = \sum_{t=1}^{T} (\mu\prime \ - \mu_i(t))$$

As we can see, each action is penalized according to the quality of the selected item. Cumulative regret measures the average quality of the decisions made by a learner during a time period T. A widely adopted objective of MAB algorithms is to minimize the expected cumulative regret [9].

In addition to looking at the expected regret, another variable of interest is sample complexity, i.e. the number of steps required to identify the optimal arm. Under the PAC ("probably approximately correct") setting, the goal can be formally formulated as finding an ε-optimal arm whose expected reward is within ε from the expected reward of the best arm with probability at least $1-\delta$ [9, 61]. In contrast to regret minimization tasks that consider the average quality of selected arms, the performance of a learner in the PAC setting is measured purely based on sample complexity, which is usually expressed in terms of ε, δ and the total number of arms K. In the PAC setting, each decision is penalized equally, regardless of the quality of the selected items.

Multi-armed bandit problems enjoy a long history, extending back to the early 1930s for applications in medical trials [77]. Recent years have seen a substantial increase for applications in online content delivery, such as displaying ads on a search page and online recommendation systems. The classic multi-armed bandit problem has several variants. For instance, dueling bandits, contextual multi-armed bandits [58] and rotting bandits [49]. Dueling bandits are discussed in the next section while the latter two are left out since they are beyond the scope of the study.

2.5 Dueling Bandits

The dueling bandits problem, first introduced by Yue et al. in 2012 [90], represents an important variant in which the algorithm attempts to identify the best of K arms through noisy pairwise comparisons. In some situations, preference-based feedback indicating that one arm is better than the other is more promising and easier to collect than real-valued feedback which numerically quantifies the quality of arms. Taking online ranker evaluation tasks as an example, interleaved comparison methods, which compare rankers by interleaving their results and using click data as preference signal, have proven to be more reliable, efficient, and substantially cheaper than collecting pointwise relevance judgments from experts [38, 42]. Due to these advantages, dueling bandits together with interleaving have been widely explored and employed as a tool for ranker evaluation in the context of online learning to rank [9]. Cumulative regret is the main evaluation criterion since we would like the two rankers being compared at each time step to be as good as possible while being able to explore new possibilities. Comparing between bad rankers can negatively hurt user experience.

The K-armed dueling bandits problem considers a set of K arms $\{a_1, a_2, ..., a_K\}$. At each time step, a pair of arms (a_i, a_j) is pulled for comparison and the feedback is a relative preference suggesting which arm is preferred. Let $p_{i,j}$ denote the probability of a_i beating a_j, and $p_{j,i} = 1 - p_{i,j}$. The preference matrix is defined as: $\mathbf{P} = [p_{i,j}]$. The goal is to find the optimal arm via pairwise comparison. Since the observed feedback is stochastic, the order between two arms can only be estimated with a certain probability and needs multiple comparisons to combat against noise. The problem of identifying the optimal item is therefore reduced to estimating the preference matrix $\mathbf{P}$.

A number of dueling bandits algorithms have been proposed [50, 54, 92, 90, 95, 96]. Depending on the assumptions on underlying preference relations as well as definitions of the optimal arm, dueling bandits problems can be formulated in several ways: the Condorcet dueling bandits problem, the Copeland dueling bandits problem, and the Borda dueling bandits problem. The categories described here are not exhaustive and I only focus on the ones that are most common and regularly referred to in this thesis.

2.5.1 The Condorcet Assumption

In the Condorcet setting, dueling bandits algorithms require the existence of the Condorcet winner, which is a single arm that beats all others with a probability greater than 0.5 [78]. Without loss of generality, let a_1 denote the Condorcet winner. Then $p_{1,i} > 0.5$ holds for all $i > 1$.

A challenge that arises naturally in the regret-minimization dueling bandits problem is formulating a proper notion of regret. Given the Condorcet winner a_1 and a pair of arms $a_{i(t)}$ and $a_{j(t)}$ selected for comparison at iteration t, a commonly used definition of regret at iteration t is $r_t = \frac{\Delta_{1,i(t)}+\Delta_{1,j(t)}}{2}$, where $\Delta_{i,j} = p_{i,j} - 0.5$. Regret measures the average suboptimality of the two arms being compared against each other with respect to the Condorcet arm a_1. The cumulative regret after T iterations is $R_T = \sum_{t=1}^{T} r_t = \frac{1}{2}\sum_{t=1}^{T}(\Delta_{1,i(t)}+\Delta_{1,j(t)})$.

2.5.2 The Copeland Assumption

A key limitation of the Condorcet setting is that practically the Condorcet winner does not exist. For instance, Zoghi et al. [94] show that the probability of the existence of the Condorcet winner in online ranker evaluation decreases rapidly as the number of rankers increases. To address this issue, the authors present two algorithms for the Copeland setting. Unlike the Condorcet winner that beats all other arms with a probability greater than 0.5, Copeland winners are the arms beating the maximal number of other arms and are guaranteed to exist. The Condorcet winner can be viewed as a special case of Copeland winner.

Let $Cpld(a_i)$ denote the Colepand score of a_i, which is defined as the number of arms $a_j \in \{a_1, a_2, ..., a_K\}$ for which $p_{i,j} > 0.5$. The normalized Copeland score of a_i is $cpld(a_i) = \frac{Cpld(a_i)}{K-1}$. Copeland winners are the ones with the largest normalized Copeland score. In the Copeland setting, the regret incurred by comparing $a_{i(t)}$ and $a_{j(t)}$ at iteration t is defined with respect to a Copeland winner. Without loss of generality, let a_1 denote a Copeland winner. Then the regret of comparing $a_{i(t)}$ and $a_{j(t)}$ is commonly defined as $2cpld(a_1) - cpld(a_{i(t)}) - cpld(a_{j(t)})$ [47, 84, 94].

2.5.3 The Borda Assumption

An alternative definition of the optimal arm that has been studied in the bandits literature is Borda winner, i.e. an arm with the greatest probability of being preferred to another randomly selected arm [28, 40, 54, 78]. More formally, assuming a pool of K arms, the Borda score of an arm a_i is defined as: $b_i = \frac{1}{K-1}\sum_{j=1,j\neq i}^{K} p_{i,j}$. The Borda winner is the arm with the largest Borda score. Similarly as for the Copeland winner, the Borda winner always exists for every possible preference probability matrix [40]. The evaluation criterion for Borda settings typically is sample complexity, i.e. the number of comparisons required to find the Borda winner.

2.6 Crowdsourced Maximum and Top-k Computation

Many existing works have considered the problem of identifying max/top-k objects from a given set in crowdsourcing environments. However, they do not frame the problem as dueling bandits. Venetis and Garcia-Molina [79] consider finding the maximum in crowd-sourced settings and provide tournament algorithms which can be tuned on parameters like monetary cost, execution latency, and output quality. In each microtask, human workers are asked to select the best item from several candidates. If they disagree on the max, additional workers are requested dynamically. The winners proceed to the next round, and the process is repeated until a single item remains. The algorithm was analyzed and empirically evaluated with respect to human error models obtained from psychometrics. Guo et al. [35] break the max problem into two subproblems, the judgment problem and the next votes problem. The judgment problem aims to estimate the current best item given a set of comparison results, and the next vote problem aims to determine which future comparisons are most effective based on the current results. The authors present maximum likelihood strategies and several graph-based heuristic strategies to compute the max via pairwise comparisons. Verroios et al. [80] study the budget allocation problem for tournament-based max operations and present a strategy that minimizes the latency. Khan and GarcíaMolina [44] present a hybrid method which asks the crowd to first grade individual items, and then perform pairwise comparisons to find the maximum element.

The max problem can be generalized to the top-k problem. Davidson et al. [25] construct tournament trees and make use of pairwise comparisons and heap-sort techniques to obtain top-k elements. Busa-Feketes et al. [12] propose a preference-based racing algorithm to identify top-k lists. However, they did not focus on minimizing the monetary cost. Li et al. [52] present budget-aware and confidence-aware frameworks obtain the top-k items with pairwise judgments that require human workers to weigh the preference for a pair of items via a sliding bar. Chen et al. [16] propose a method to aggregate crowdsourced pairwise comparisons into a global ranking. In addition to finding the next pair of items to compare, they focus on finding the best worker to do the comparison by estimating the reliability of each individual worker.

There are other works that are not restricted to pairwise comparisons. Polychronopoulos et al. [5] allow human workers to compare a small subset of items at a time and merge local ranked lists via median rank aggregation [27] to select the global top-k items. Li et al. [51] propose a rating-ranking method to identify top-k objects, which asks human workers to either rate or rank several objects. The ratings and rankings of items are finally aggregated to compute the top-k results. De Alfaro et al. [26] introduce a recursive algorithm which partitions items into subsets and asks human workers to choose the top-1 of each subset.

Chapter 3

Analysis

This chapter defines and justifies the requirements that a dueling bandits algorithm for human preference judgments should satisfy, and surveys algorithms from the machine learning and information retrieval literature.

3.1 Problem Formulation

Dueling bandits algorithms have been widely explored as a tool for online ranker evaluation via interleaving methods. Each ranker is viewed as an arm and the feedback is collected via interleaving methods. When it comes to human preference judgments, the requirements differ substantially from those for online ranker selection. Assuming a pool of K items potentially satisfying a given query, each item can be viewed as an arm. In an academic research context, this pool of items might be formed from experimental runs submitted to an evaluation forum such as TREC. In a commercial context, the pool might be formed from the results returned by production vs. experimental rankers. In this thesis, I assume that each item in this pool represents a plausible response to the query, since a preference judgment between items that are unrelated to the query would be pointless. As a result, I assume that the pool has been filtered with a binary relevance pass, eliminating unrelated items.

The objective is to identify with high probability the best of K arms through noisy pairwise comparisons. A preference judgment compares two items i and j with the unknown probability of i beating j equal to $p_{i,j}$. For human preference judgments, I assume that each comparison of i and j is independent, with different people making each judgment. I assume that human judges may be crowdsourced workers or temporary contract employees with minimal or no training. Consequently, two sources of noise are recognized: 1) genuine disagreements of opinion, where the balance between competing factors might cause independent judges to reach different conclusions; and 2) erroneous judgments due to misunderstandings, distractions, or lack of training. While the use of trained and dedicated assessors might reduce the second source, nothing can be done about the first source. As a result, even with trained and dedicated assessors, $p_{i,j}$ typically will not be close to 0 or 1. As an exception, if an assessor is considered to be "authoritative" [64] then $p_{i,j}$ will be always 0 or 1. In this case, preference judgments are not dueling bandits, and traditional sorting and selection algorithms may be applied.

Bengs et al. [9] review common assumptions on pairwise preferences across the dueling bandits literature. A prevalent assumption that can be seen in the majority of the work is "No Ties". At a purely theoretical level, it might be reasonable to assume $|\Delta_{i,j}| > 0$ if $i \neq j$. As long as there is some difference for comparison between i and j and an endless supply of independent human assessors, a preference difference that cannot be explained by chance can eventually be detected. However, practically any algorithm being employed needs to gracefully handle cases where some values of $\Delta_{i,j}$ are arbitrarily close to zero. When collecting judgments, I avoid asking assessors if two items appear equal, or nearly so, since there is no universal way to define "near equality". Instead, I require assessors to seek meaningful differences, however minor.

Ambiguous queries provide an example where $\Delta_{i,j}$ can be close to zero. The first two passages in Figure 1.1 are among the top 5 selected for the ambiguous query "Who is Jaffe?". They both provide concise biographical information. Based on the relative recency and number of their roles, Taliesin, with 170 credits on IMDb, might be preferred to Nicole, who is best known as the voice of Velma on Scooby-Doo. Nonetheless, either is preferable to the third passage, which is less focused on biographical information.

Unlike the interleaving methods used for online learning to rank, human preference judgments are expensive. For the crowdsourced experiments reported in this work, a single judgment cost nearly 36 cents on average. As a result, minimizing the number of comparisons is crucial due to budget constraints. Practically, there is also a limit on the number of independent judgments that can be obtained for a given pair since each requires a different assessor.

To summarize, a dueling bandit algorithm for human preference judgments must satisfy the following requirements:

1. **Near linearity:** The number of comparisons should grow linearly with the number of arms K, up to a logarithmic term.

2. **Equality tolerance:** The algorithm must gracefully handle cases where values of $\Delta_{i,j}$ are close or equal to zero. The impact of $\Delta_{i,j} = 0$ values must be considered.

3. **Parsimony:** The algorithm must be practically usable, minimizing both the total number of assessments and the number of independent assessments of a given pair. Even with near linear growth and equality tolerance, constant factors can render algorithms too expensive for human assessment.

The goal is to find the best item from a pool of K items. The most natural definition of "best item" is the item preferred over any other, i.e. the Condorcet winner. However, given the possibility of ties, Copeland winners provide an alternative definition of "best", i.e., the arm or arms that beat the most other arms. We can also define best items in terms of a sum of expectations, where the winner is the one with the largest Borda score:

$$b_i \; = \; \frac{1}{K-1} \sum_{j=1, j\neq i}^{K} p_{i,j} \tag{3.1}$$

In the dueling bandits literature, an alternative approach to model pairwise probabilities is through utility functions. Assume that there is a latent utility function $u : \mathcal{A} \rightarrow \mathbb{R}$ mapping an arm $a_i \in \mathcal{A}$ to a real-valued utility score $u(a_i)$. Such scores can reflect the absolute preference for each arm. In a pairwise comparison of a_i and a_j, the probability of a_i beating a_j can be represented as:

$$\mathbf{P}(a_i \succ a_j) = \sigma(u(a_i) - u(a_j)), \tag{3.2}$$

where $\sigma : \mathbb{R} \rightarrow [0, 1]$ is a link function mapping differences of utilities to probabilities [9]. The link function must be monotonic and have $\sigma(0) = 1/2$ so that an arm with a higher utility has a higher probability of beating the ones with lower utilities. *Logistic link function* $\sigma(x) = 1/(1 + exp(-x))$ is a common example of link functions.

Without loss of generality, assume a pool of K items $\mathcal{A} = \{a_1, a_2, ..., a_K\}$, a utility function $u : \mathcal{A} \rightarrow \mathbb{R}$ so that $u(a_1) > u(a_2) > ... > u(a_K) > 0$, and a monotonically non-decreasing link function $\sigma : \mathbb{R} \rightarrow [0, 1]$. According to (3.1) and (3.2), the Borda score of an arm $a_i \in \mathcal{A}$ is:

$$b_i = \frac{1}{K-1} \sum_{j=1, j\neq i}^{K} p_{i,j} = \frac{1}{K-1} \sum_{j=1, j\neq i}^{K} \sigma(u(a_i) - u(a_j))$$

We now take the difference of b_i and b_{i+1} to examine whether Borda ranking aligns with utility ranking:

$$\begin{aligned}
b_i - b_{i+1} &= \sum_{j=1, j\neq i}^{K} \sigma(u(a_i) - u(a_j)) - \sum_{j=1, j\neq i+1}^{K} \sigma(u(a_{i+1}) - u(a_j)) \\
&= \sum_{j=1, j\neq i, j\neq i+1}^{K} [\sigma(u(a_i) - u(a_j)) - \sigma(u(a_{i+1}) - u(a_j))] \\
&\qquad + \sigma(u(a_i) - u(a_{i+1})) - \sigma(u(a_{i+1}) - u(a_i))
\end{aligned}$$

Since $u(a_i) > u(a_{i+1})$, we have $u(a_i) - u(a_j) > u(a_{i+1}) - u(a_j)$. Then, by the non-decreasing monotonicity of σ, we obtain $\sigma(u(a_i) - u(a_j)) > \sigma(u(a_{i+1}) - u(a_j))$. Therefore, $\sum_{j=1,j\neq i,j\neq i+1}^{K}[\sigma(u(a_i) - u(a_j)) - \sigma(u(a_{i+1}) - u(a_j))] > 0$. Similarly, $\sigma(u(a_i) - u(a_{i+1})) - \sigma(u(a_{i+1}) - u(a_i)) > 0$ since $u(a_i) - u(a_{i+1}) > u(a_{i+1}) - u(a_i)$. Thus, we get $b_i - b_{i+1} > 0$, which implies Borda ranking aligns with utility ranking.

Borda ranking and utility ranking both obey the probability ranking principle (PRP), which is one of the theoretical foundations for ranking in information retrieval. The PRP states that the overall effectiveness of the retrieval system to its user will be maximized if documents are ranked in order of probability of relevance or usefulness [17]. In utility-based dueling bandits, it is assumed that there is a utility function $u : \mathcal{A} \to \mathbb{R}$ mapping an arm $a_i \in \mathcal{A}$ to a real-valued utility score $u(a_i)$, so that arms with higher scores are more of relevance or usefulness than the ones with lower scores. Probability of relevance is essentially a special case of utility scores: the utility function u maps arms to real-valued scores in the range of 0 and 1. Therefore, Borda ranking and utility ranking obey the probability ranking principle.

3.2 Bengs et al. [9]

While there exists a substantial body of research on dueling bandits and human preference judgments in the context of information retrieval, explicit connections between them are rare. Głowacka [32] (Chapter 5) reviews both the use of dueling bandits for online evaluation and the use of (non-dueling) bandits for traditional pointwise relevance judgments [57], but does not connect the two concepts. The focus of this thesis is to investigate the use of dueling bandits for human preference judgments.

To find possible solutions, I review existing algorithms from the machine learning and information retrieval literature in light of the requirements listed in the previous section. I consider the suitability of these algorithms for identifying the best item or items in the pool through human preference judgments. In undertaking this review, I was greatly assisted by the comprehensive survey on dueling bandits by Bengs et al. [9], which served as a foundation for the review. In the survey, existing algorithms are categorized into different classes based on specific properties. I follow their framework and discuss the possibilities of these algorithms being applied to human preference judgments.

3.2.1 Coherent Preference Relations

As discussed in Section 2.5, the problem of identifying best arms or inducing a total order over arms can be naturally reduced to estimating the pairwise preference matrix $\mathbf{P} = [p_{i,j}]$. However, unlike the traditional value-based MAB setting in which the optimality of an arm is quantified on a numerical scale and a total order over arms can be easily derived based on the absolute rewards, the correlation between the preference probabilities and the order of arms is more complicated in the preference-based setting [9]. For example, given a preference matrix P, a total order over arms may not exist due to preferential cycles. Therefore, in order to allow the learner to find the target, a number of assumptions on the preference probabilities are proposed to make the preference matrix P coherent. Bengs et al. [9] categorized the approaches of learning from coherent pairwise comparisons into three classes: 1) axiomatic approaches, 2) utility functions, and 3) statistical models. I discuss the three groups of algorithms one by one.

Axiomatic Approaches. Bengs et al. [9] collect various assumptions on pairwise preferences that can be found in the literature and give a detailed analysis on the relationships between these assumptions. Some of the common assumptions that are referred to in this thesis include:

- *Total order over arms*: There exists a total order over all arms such that $a_i \succ a_j$ indicates $\Delta_{i,j} = p_{i,j} - 0.5 > 0$.
- *Strong stochastic transitivity* (SST): For any triplet of bandits $\{a_i, a_j, a_k\}$ such that $\Delta_{i,j} \geq 0$ and $\Delta_{j,k} \geq 0$, $\Delta_{i,k} \geq \max(\Delta_{i,j}, \Delta_{j,k})$.
- *γ-relaxed stochastic transitivity* (γ-RST): For any triplet of bandits $\{a_i, a_j, a_k\}$ such that $\Delta_{i,j} \geq 0$ and $\Delta_{j,k} \geq 0$, $\gamma\Delta_{i,k} \geq \max(\Delta_{i,j}, \Delta_{j,k})$ for some $\gamma \geq 1$.

- *Moderate stochastic transitivity* (MST): For any triplet of bandits $\{a_i, a_j, a_k\}$ such that $\Delta_{i,j} \geq 0$ and $\Delta_{j,k} \geq 0$, $\Delta_{i,k} \geq \min(\Delta_{i,j}, \Delta_{j,k})$.
- *Stochastic triangle inequality* (STI): For any triplet of bandits $\{a_i, a_j, a_k\}$ such that $\Delta_{i,j} \geq 0$ and $\Delta_{j,k} \geq 0$, $\Delta_{i,k} \leq \Delta_{i,j} + \Delta_{j,k}$.
- *No ties*: $\Delta_{i,j} \neq 0$ for all $i \neq j$.
- *Existence of a Condorcet winner*: There exists an arm a_i such that $\Delta_{i,j} \geq 0$ for all $j \neq i$.

Experimental evidence reported in prior work suggests that preference judgments made by dedicated assessors are very nearly transitive [14, 71], so that it is safe to assume strong stochastic transitivity, but not stochastic triangle inequality. Interleaved Filtering(IF) [90] and Beat the Mean(BTM) [92] are two of the early dueling bandits algorithms and both follow an "explore then exploit" approach to find the best item while minimizing cumulative regrets. To establish theoretical guarantees, they make restrictive assumptions about the underlying preference relations: a total order over arms, stochastic triangle inequality, strong stochastic transitivity for IF or γ-relaxed stochastic transitivity for BTM. It has been shown that the two algorithms incur the following expected regret bounds for the K-armed dueling bandit problem:

$$O\left(\frac{K \log T}{\min_{j \neq 1} \Delta_{1,j}}\right) \text{ for Interleaved Filtering, and}$$

$$O\left(\frac{\gamma^7 K \log T}{\min_{j \neq 1} \Delta_{1,j}}\right) \text{ for Beat the Mean,}$$

where a_1 denotes the best arm and T denotes the exploration horizon.

While both of the regret bounds grow linearly with the number of arms K, the constant factors in the notations are huge. Additionally, stochastic triangle inequality is often violated in practice. Hence, the theoretical performance bounds may not hold. In addition, the bounds are expressed in terms of the inverse $1/\Delta_{i,j}$, possibly resulting in an infinite value when $\Delta_{i,j}$ is close to zero. This can have an unforeseen impact and possibly violate the equality tolerance requirement.

Later dueling bandits work for the regret minimization task commonly adapt the well-known UCB algorithm and Thompson sampling to select arms for comparison. RCS [97], RUCB [96], MergeRUCB [95], and MergeDTS [50] identify the best item with high probability while only requiring the existence of the Condorcet winner. When selecting the next pair of items being compared, RCS and RUCB select the first item by choosing the one that is most likely to be the Condorcet winner. The second item is the first one's strongest competitor, i.e., the one with the highest probability of beating the first item. One major difference between the two algorithms is that RCS estimates preference probabilities by sampling, whereas RUCB estimates probabilities by confidence intervals. MergeRUCB focuses on the large-scale online ranker evaluation problem and improves upon RUCB and RCS by using a divide and conquer strategy. In subsequent work, Li et al. [50] propose to further improve MergeRUCB by making use of Thompson sampling to select arms. While the theoretical guarantees for RCS are not provided, the other three have the following high probability regret bounds:

$$O(K^2 + \sum_{j \neq 1} \frac{\log T}{\Delta_{1,j}^2}) \text{ for RUCB, and}$$

$$O(\frac{K \log T}{\min_{i,j:p_{i,j} \neq 0.5} \Delta_{i,j}^2}) \text{ for MergeRUCB, and}$$

$$O(\frac{K \log T}{\min_{i,j:p_{i,j} \neq 0.5} \Delta_{i,j}^2}) \text{ for MergeDTS.}$$

The quadratic term in RUCB's regret bound violates the near linearity requirement. MergeRUCB and MergeDTS enjoy the same upper bounds which grow linearly in the number of arms. However, since the Condorcet winner is not guaranteed to exist, the algorithms can possibly fail to produce the correct answer. Strictly speaking, any algorithm that relies on the existence of the Condorcet winner can be excluded from consideration on theoretical grounds. Another thing worth noting is that the two algorithms both make special requirements about "ties", which is covered in Section 3.3.2. I select MergeDTS for further consideration since it represents the state-of-the-art large-scale dueling bandits algorithm, and outperforms many algorithms such as RUCB [96], RMED1 [45] in most experimental setups [50].

While previous methods focus on regret minimization, sample complexity is another variable of interest. Knockout [30], Seq-Eliminate [28], and Opt-Max [29] consider the problem of finding the best arm while minimizing sample complexity under the (ϵ, δ)-PAC setting. The three algorithms assume a total order over arms. Additionally, Knockout assumes SST, STI, and γ-RST. Seq-Eliminate assumes SST. Opt-Max assumes MST, which is a relaxed version of SST. Given the assumptions, they achieve the following sample complexity theoretical guarantees:

$$O(\frac{K}{\gamma^4 \epsilon^2} \log \frac{1}{\delta}) \text{ for Knockout, and}$$

$$O(\frac{K}{\epsilon^2}\log\frac{1}{\delta}) \text{ for Seq-Eliminate, and}$$

$$O(\frac{K}{\epsilon^2}\log\frac{1}{\delta}) \text{ if } \delta \geq \min(1/K, exp(-K^{1/4})) \text{ for Opt-Max.}$$

At first glance, they seem to be suitable candidates for human preference judgments because their theoretical performance guarantees grow linearly in the number of arms and are not expressed in terms of the inverse $1/\Delta_{i,j}$. However, the three algorithms make heavy use of a subroutine called COMPARE, which keeps comparing two arms until the confidence about the winner is sufficiently large or the comparison budget for a single pair is exhausted. Given a round dependent bias ϵ and confidence δ, the comparison budget for Knockout is $\frac{1}{2\epsilon^2}\log\frac{2}{\delta}$. Different from Knockout, Seq-Eliminate and Opt-Max take lower bias ϵ_l, upper bias ϵ_u and confidence δ. Seq-Eliminate compares two elements $\frac{2}{(\epsilon_u-\epsilon_l)^2}\log\frac{2}{\delta}$ times, and Opt-Max compares two elements $\frac{8}{(\epsilon_u-\epsilon_l)^2}\log\frac{2}{\delta}$ times. In the case of ties, the number of comparisons can easily grow from hundreds to thousands, violating the equality tolerance and parsimony requirements.

In 2020, Ren et al. [66] propose the Tournament-k-Selection (T-k-S) method to find the best k arms under (ϵ,δ)-PAC setting. Assuming a total order over arms, strong stochastic transitivity and stochastic triangle inequality, T-k-S is shown to have an optimal sample complexity of $O(\frac{K}{\epsilon^2}\log\frac{k}{\delta})$. There are two concerns about T-k-S when it comes to human preference judgments. First, stochastic triangle inequality cannot be safely assumed. Second, similar to Knockout, Seq-Eliminate and Opt-Max, T-k-S makes use of a subroutine called Distribute-Item, which repeatedly compares two items at most $\frac{2}{\epsilon^2}\log\frac{4}{\delta}$ times. Again, the number can grow easily from hundreds to thousands when two items are similar. In general, algorithms that repeatedly compare two arms until a winner is found are not suitable for human preference judgments since it may require too many independent assessors and cost too much when two arms are nearly equal. Ideally, there should be a feasible comparison budget that can be spent on every individual pair.

Mohajer et al. [63] study the top-k partition and ranking problem via noisy comparisons under the assumption of a total order over items. On the special case where $k = 1$, the proposed algorithm SELECT has a sample complexity of $O(\frac{K\log\log K}{\min_{j\neq 1}\Delta^2_{1,j}})$ for identifying the top item a_1 from a group of K items. SELECT is a single-elimination tournament where items are randomly paired and compared for a fixed number of times, with the winners proceeding to the next round. Ties are broken randomly. The algorithm is further extended to find the top-k items. I consider the algorithm in Section 3.3.

Utility-based Dueling Bandits. In addition to the axiomatic approaches that rely on the regularity properties, another way to represent preferences is through utility functions. A utility function $u : \mathcal{A} \to \mathbb{R}$ maps an arm $a_i \in \mathcal{A}$ to a real-valued utility score $u(a_i)$ so that arms with higher scores are more likely to be selected than the ones with lower scores. The probability of a_i beating a_j can be represented as $\mathbf{P}(a_i \succ a_j) = \sigma(u(a_i) - u(a_j))$, where $\sigma : \mathbb{R} \to [0, 1]$ is a link function mapping differences of utilities to probabilities. In 2009, Yue and Joachims [91] introduce the utility-based dueling bandit problem and propose a gradient-descent method built on methods for online convex optimization tasks. They assume the existence of a compact, convex parameterized space of arms containing the origin. For human preference judgments, I do not have a set of featurized vectors representing the compatibility between query-passage pairs. Generating such vectorized representation is beyond the scope of this work. Hence, gradient descent algorithms requiring a compact, convex action space such as [91, 48] are excluded from consideration.

Ailon et al. [3] propose two reduction-based approaches that reduce the utility-based dueling bandits problem to the classic value-based multi-armed bandits problem, demonstrating that one of the two methods, MultiSBM, is superior to IF [90] in many scenarios. Zimmert and Seldin [93] study a more general multi-armed bandits setting where each action can be decomposed into a Cartesian product of elementary actions. The algorithm can be applied to utility-based dueling bandits, superior to many known algorithms including IF, BTM, RUCB, MultiSBM, besides RMED1 in empirical experiments. I did not consider them in Section 3.3 because the selected method MergeDTS already outperforms RMED1.

Lin and Lu [54] consider winner selection in sports tournaments as dueling bandits and assume a latent utility for each arm as well as a linear link function. They estimate the utilities of each arm and successively eliminate empirically inferior arms based on associated confidence intervals to identify the arm with the highest utility. With a modification of the stopping criterion, they show that the proposed method can return a Borda winner under (ϵ, δ)-PAC setting. After running $O(\frac{1}{\epsilon^2} \log \frac{K}{\epsilon\delta})$ times, an ϵ-Borda winner is returned with probability at least $1 - \delta$. The sample complexity takes the form: $O(\frac{K}{\epsilon^2} \log \frac{K}{\epsilon\delta})$. Since the algorithm is specifically intended to support human tasks such as best player selection in sports tournaments, I select it for further consideration.

Regularity Through Statistical Models. This group of papers, including [11, 76] assumes that the probability of observing a ranking of the arms is generated by certain statistical models. Let $\mathbb{S}_K$ denote the set of rankings(or permutations) of the K arms $[K] = \{1, ..., K\}$. Under the assumption of a statistical ranking model $\mathbf{P} : \mathbb{S}_K \to [0, 1]$, with $\mathbf{P}(\pi)$ the probability of observing a ranking $\pi \in \mathbb{S}_K$, the probability of a_i being preferred to a_j is denoted by:

$$p_{i,j} = \mathbf{P}(a_i \succ a_j) = \sum_{\pi \in \mathbb{S}_K, \pi(j) > \pi(i)} \mathbf{P}(\pi)$$

, where $\pi(i)$ and $\pi(j)$ are the ranks of a_i and a_j in the ranking π.

To find the optimal item, Busa-Fekete et al. [11] propose the MALLOWSMPI algorithm, assuming a Mallow's ϕ-distribution model[60]. Szörényi at al. [76] propose the PLPAC algorithm, assuming a Plackett-Luce (PL) model [59, 65]. Under a PL model where each arm $i \in [K]$ has a skill parameter v_i, the probability that arm i is preferred to arm j is

$$p_{i,j} = \frac{v_i}{v_i + v_j}.$$

Assuming a ranking $\mathbf{r} = (r_1, ..., r_K)$ where r_i is the rank of the ith item and an ordering $\mathbf{o} = (o_1, ..., o_K)$ where o_{r_i} = i for all $i \in [K]$, the probability of $\mathbf{r}$ is

$$\mathbf{P}(\mathbf{r}) = \prod_{i=1}^{K} \frac{v_{o_i}}{v_{o_i} + v_{o_{i+1}} + ... + v_{o_K}}$$

The PL model satisfies strong stochastic transitivity, stochastic triangle inequality, and is guaranteed to have a Condorcet winner. Under the assumption of a PL model, the PLPAC algorithm makes use of a budgeted Quicksort algorithm to recover partial rankings. At each round, a random pivot item is selected and compared with the rest of items one by one, partitioning items into two groups. The process is repeated until a PAC-item is found. Let i^* represent the optimal arm and let $\Delta_i = \frac{1}{2}\max\{\epsilon, p_{i^*,i} - \frac{1}{2}\}$ for each $i \neq i^*$. With probability at least $1 - \delta$, PLPAC outputs an ϵ-optimal arm using at most $O(K \max_{i \neq i^*} \frac{1}{\Delta_i^2} \log \frac{K}{\Delta_i \delta})$ pairwise comparisons. It has been shown that PLPAC outperforms IF, BTM, and MALLOWSMPI in experiments on synthetic data, and on real data where the assumption is violated to a certain degree. Since evaluating to what extent human preference judgments fit to a PL model is beyond the scope of the thesis, and the paper does not have enough information for me to re-implement the method, I leave this for future work.

3.2.2 Non-coherent Preference Relations

The methods presented so far assume either the existence of the Condorcet winner or an underlying ground truth ordering over arms. In real-world scenarios, such assumptions may not hold. An alternative way is to define winners based on the pairwise preference probabilities. Various notions of winners have been proposed as an alternative to the Condorcet winner, including Copeland winners and Borda winners. In this section, I review existing methods and consider their suitability for human judgments.

In 2015, Zoghi et al. [94] introduce the Copeland dueling bandits problem and propose two algorithms designed for regret minimization tasks: Copeland Confidence Bound (CCB) and Scalable Copeland Bandits (SCB). Assuming that there are no ties, the expected regret bound for CCB is $O(\frac{K^2+(C+L_C)K\log T}{\min_{i\neq j}\Delta_{i,j}^2})$, where C is the number of Copeland winners and L_C is the number of arms that win a Copeland winner. SCB addresses the scalability concern about CCB by eliminating the $O(K^2)$ term. The expected regret bound for SCB takes the form: $O(\frac{K(\log K+L_C)\log T}{\min_{i\neq j}\Delta_{i,j}^2})$. Later in 2016, Wu and Liu [84] propose the Double Thompson Sampling (DTS) algorithm that works in both Condorcet setting and Copeland setting. DTS breaks ties randomly and achieves $O(\frac{K^2\log T}{\min_{i\neq j}\Delta_{i,j}^2}+K^2)$ regret. Based on the experiment results, DTS significantly outperforms many algorithms such as BTM, RUCB, CCB, and SCB. While the $O(K^2)$ term seems to violate the near linearity requirement, I select DTS for further consideration due to its notable practical performance and the ability to handle ties.

In addition to identifying the best item, another variant is to find the top-k subset or top-k ranking by actively comparing pairs of items. Heckel et al. [36] propose an active ranking algorithm referred to as AR to partition items into sets of pre-specified sizes based on their estimated Borda scores, which includes special cases of identifying the top-k items and a total ordering. Maystre and Grossglauser [62] propose the Multisort algorithm, which invokes the QuickSort algorithm multiple times and aggregates the rankings into a final ranking by Copeland scores. Groves and Branke [34] focus on the problem of identifying top-k Borda items and propose two adapted Bayesian sequential sampling methods. However, no bound on the sample complexity is provided. The AR algorithm is selected for further consideration since it is representative of active ranking algorithms and the implementation is made available.

3.3 Candidate Algorithms

Based on the review, most algorithms from the dueling bandits literature do not provide a close match to the requirements for near linearity, equality tolerance, and parsimony. While I do not want to exclude algorithms on purely theoretical grounds, some algorithms employ basic strategies that are clearly inconsistent with human preferences. For example, strategies that repeatedly compare two arms until a winner is found may require too many independent assessors when arms are nearly equal [28, 30, 66]. The remaining algorithms could also be excluded on theoretical grounds. Some require quadratic judgments, possibly violating near linearity. Others have performance guarantees expressed in terms of $1/\Delta_{i,j}$, possibly violating equality tolerance. Others rely on the stochastic triangle inequality or the existence of the Condorcet winner. Although they could be excluded on theoretical grounds, I select several algorithms based on their potential to work in practice even if the theoretical assumptions are not satisfied. These algorithms use basic strategies that seem plausible in the context of human preferences. Sheth and Rajkumar [72] study a practical scenario where a winner is identified via active pairwise comparisons using very few comparisons. Although this work is not covered in the survey by Bengs et al. [9], it is the same as the dueling bandits problem and seems highly relevant to my research problem. In addition, I consider candidate algorithms from the literature related to human preferences for information retrieval evaluation. Clarke et al. [20] and Arabzadeh et al. [6] focus on finding the top items from a pool. I select the unnamed algorithm in Clarke et al. [20], since it shares my goal of finding top items and accommodates pools with hundreds of items. In the rest of this section, I briefly introduce the selected algorithms.

3.3.1 Double Thompson Sampling

While prior studies mostly focus on the Condorcet setting, Wu and Liu [84] propose the Double Thompson Sampling (DTS) method that works under both Condorcet setting and Copeland setting. As previously mentioned, Copeland winners, which are guaranteed to exist, generalize the concept of a Condorcet winner and are more practical.

Double Thompson Sampling makes use of the well-known Thompson sampling algorithm that dates back to 1933 [77]. A fundamental dilemma in probabilistic problems is the exploration-exploitation trade-off. Thompson sampling attempts to resolve this by taking uncertainty into consideration while sampling candidates based on how likely they are optimal. It has been widely studied in traditional multi-armed bandits and other online learning problems [2, 15, 33, 46, 85].

DTS maintains beta posterior distributions over the expected preference probabilities. At each time step, it selects two candidates for comparison by sampling from the beta distributions. When selecting the first arm, DTS first eliminates the arms that are unlikely to be potential winners based on upper confidence bounds, and then chooses the arm that beats the maximal number of others according to the sampled probabilities. Ties are broken at random. When selecting the second arm, the posterior distributions are used again to sample the probabilities of each arm being preferred to the first candidate. Among those uncertain arms whose lower confidence bound for the preference probability against the first arm is at most $1/2$, the one with the largest sampled probability of being preferred over the first candidate is chosen as the second candidate in search of higher information gain. The beta distributions are updated accordingly based on the comparison results, and the procedure is repeated for a time horizon T.

DTS is designed and evaluated in terms of a regret minimization criterion. Under the assumption that there are no ties, DTS achieves a theoretical regret bound of the form $O(K^2 \log T)$, which grows quadratically in the number of arms. While quadratic growth violates our requirements, the algorithm may still be practically suitable for small K. The experiment results reported by the authors show that DTS outperforms algorithms such as BTM, RUCB, CCB, and SCB.

3.3.2 Merge Double Thompson Sampling

Zoghi et al. [50] consider the large-scale online ranker evaluation problem under the Condorcet assumption. Although the Condorcet winner is not guaranteed to exist, it provides the most natural definition for the "best" arm and is worth investigating. To handle cases in which the number of arms is large, the authors propose Merge Double Thompson Sampling (MergeDTS) that uses a divide and conquer strategy along with Thompson sampling to identify the Condorcet winner.

MergeDTS proceeds as follows. Arms are randomly partitioned into disjoint batches of a predefined size. At each time step, the algorithm considers a single batch and selects two arms within that batch for comparison. This specific behavior is designed to avoid global pairwise comparisons between all pairs of arms, which turns out to successfully improve both computation efficiency and time efficiency. Similar to DTS, when selecting the first candidate, MergeDTS samples preference probabilities from beta posterior distributions for all pairs of arms within the current batch. The arm that beats most of the others based on the sampled probabilities is selected as the first candidate. When selecting the second arm, MergeDTS selects the first arm's worst competitor in the current batch based on the sampled preference probabilities. The motivation behind this choice is to eliminate arms as quickly as possible. Posterior distributions and confidence bounds are updated accordingly based on the comparison results. An arm a_i is eliminated if there exists another arm a_j in the same batch where a_i's upper confidence bound on the preference probability of winning a_j is below $1/2$. Once the current batch size becomes one, it is merged with another larger sized batch. The entire process iterates until there is only a single-element batch left, which contains the Condorcet winner with high probability.

To provide theoretical guarantees, MergeDTS makes the following assumptions: 1) A single Condorcet winner exists. 2) There are no ties, unless they are "uninformative" arms that cannot beat any other arms. 3) Uninformative arms represent at most one third of the full set of arms. By assuming the maximal percentage of uninformative arms, it ensures that uninformative arms can be eliminated by informative arms with high probability and the merge steps can be successfully performed [50]. Under these assumptions, MergeDTS provides a high probability bound on the cumulative regret, which is of the order

$$O\left(\frac{K \log T}{\Delta_{min}^2}\right), \text{ where } \Delta_{min} = \min_{\Delta_{i,j}>0} \Delta_{ij}.$$

The cumulative regret grows linearly with the number of arms K, which is a main advantage of MergeDTS and makes the algorithm suitable for large-scale online ranker evaluation problems. However, when Δ_{min} is small, MergeDTS may encounter a performance decline.

3.3.3 Round-Efficient Dueling Bandits

Suppose that in a large sports tournament, each player is paired with a competitor and multiple pairwise competitions are carried out simultaneously at each round. To identify the best player in the end, Lin and Lu [54] consider the problem as best arm identification for multi-armed dueling bandits. Each player is considered as an arm, and it is assumed that there is an unknown parameter μ_i for each player i measuring the underlying strength of the player. It is also assumed that there is a linear relationship between the underlying strength and preference probabilities, so that a player with higher strength is more likely to win. In this work, the probability of arm i winning arm j is defined as $\frac{1+(\mu_i-\mu_j)}{2}$. The goal is to identify the arm with the highest strength, or an ϵ-optimal arm whose strength is within ϵ of the optimal one. The algorithm tries to minimize the total number of rounds together with the total number of pairwise comparisons, since ideally a sports tournament should not last too long and extra competition can put a burden on participants. While it is hard to measure and quantify the strength of each arm on an absolute scale, relative feedback indicating which arm is better can be obtained easily via pairwise comparisons.

The algorithm proceeds in rounds and eliminates inferior arms until only a single arm remains. In the beginning, an allowed error probability δ is given, and arms are randomly partitioned into disjoint pairs. Scaled scores and confidence intervals for the latent utilities of active arms are calculated based on the observed feedback. An arm is eliminated if it loses to the empirically best arm based on the lower confidence interval of the estimated latent strength. The process stops when only a single arm remains or enough iterations have been made. Assuming that all arms can be arranged in total order, it is shown that the algorithm provides a theoretical guarantee of identifying the arm i^* with the highest strength with probability at least $1 - \delta$ via comparisons at most:

$$O\left(\Sigma_{j \neq i^*} \frac{1}{\Delta_{i^*,j}^2} \log \frac{K}{\delta \Delta_{i^*,j}}\right),$$

and has a round complexity of:

$$O\left(\frac{1}{min_{j \neq i^*} \Delta_{i^*,j}^2} \log \frac{K}{\delta min_{j \neq i^*} \Delta_{i^*,j}}\right)$$

3.3.4 Clarke et al.

While they do not frame their work in terms of dueling bandits, Clarke et al. [20] present an algorithm using crowdsourced preference judgments to find the top answers to a question from a pool of possible answers. They focused on the potential benefits of preference judgments for information retrieval evaluation, including the impact of preference judgments on the outcomes of TREC 2019 CAsT retrieval experiment. However, they did not explore the algorithm itself in any detail. Here, I consider the algorithm as a solution to the dueling bandits problem.

Figure 3.2 presents a version of the algorithm modified to return a set of best responses. I base this presentation on the paper itself, but confirm it against their implementation. The algorithm works in a series of pruning phases. During each phase, the algorithm estimates Borda scores for the remaining items in the pool and prunes those items with estimated Borda scores less than or equal to 0.5. Pruning continues until the pool size drops to a threshold m, at which point the algorithm makes a final estimate of Borda scores and returns the set of items with the greatest scores.

When the size of the pool is greater than m, the algorithm estimates Borda scores by first generating a random graph in which each item is randomly paired with at least n other items, with no pairing repeated. If the size of the pool is odd, one item will be paired with $n + 1$ others. In their implementation, Clarke et al. [20] use a brute force approach to generate this graph. Assessors judge the pairs and the algorithm estimates Borda scores from the counts of pairs won and lost. While Clarke et al. [20] provide no theoretical guarantees, their algorithm produces crowdsourcing results shown to be consistent with the judgments of dedicated assessors. Their overall approach is similar to the work of

Karnin et al. [43], which solves the equivalent problem for (non-dueling) bandits in both fixed confidence and fixed budget settings, and that paper might provide a starting point for further theoretical investigation.

To understand how the algorithm estimates Borda scores, let's suppose that there are K items in total, denoted by $[K] = \{1, ..., K\}$. The algorithm randomly pairs each item with n other items, building a graph with K vertices where each vertex has a degree of n. Without loss of generality, suppose that an item $i \in [K]$ is randomly paired with a set $\mathbb{S}_n$ of n other items. By definition, the Borda score of i is:

$$b_i = \frac{1}{K-1} \sum_{j \in [K]-\{i\}} p_{i,j}$$

By symmetry, each item of $[K] - \{i\}$ has the probability of $\frac{n}{K-1}$ to belong to $\mathbb{S}_n$. Hence, the expectation of the sum of probabilities of i being preferred to an item in $\mathbb{S}_n$ is:

$$\begin{aligned}\mathbf{E}(\sum_{j \in [K]-\{i\}} p_{i,j} \mathbf{1}_{j \in \mathbb{S}_n}) &= \sum_{j \in [K]-\{i\}} p_{i,j} \mathbf{P}(j \in \mathbb{S}_n) \\ &= \sum_{j \in [K]-\{i\}} p_{i,j} \frac{n}{K-1} \\ &= \frac{n}{K-1} \sum_{j \in [K]-\{i\}} p_{i,j}\end{aligned}$$

By taking the average, we have:

$$\frac{1}{n}\mathbf{E}(\sum_{j \in [K]-\{i\}} p_{i,j} \mathbf{1}_{j \in \mathbb{S}_n}) = \frac{1}{K-1} \sum_{j \in [K]-\{i\}} p_{i,j} = b_i,$$

implying that the method produces the estimation of Borda scores.

Next, let's examine the expected probability of a Borda winner proceeding to the next phase. Suppose item 1 is the borda winner and let b_1 denote the Borda score, i.e. the probability of item 1 winning a random arm. Let n denote the random pairings during a pruning phase. An arm proceeds to the next phase if it wins at least half of the n competitions. Hence, the expected probability of item 1 proceeding to the next phase is:

$$\mathbf{P} = \sum_{k=\lceil \frac{n}{2} \rceil}^{n} \binom{n}{k} b_1^k (1-b_1)^{n-k}$$

Figure 3.1 shows the probability of a Borda winner proceeding to the next phase with different Borda scores. As we can see, we can increase the probability of a Borda winner proceeding to the next phase by increasing the value of n.

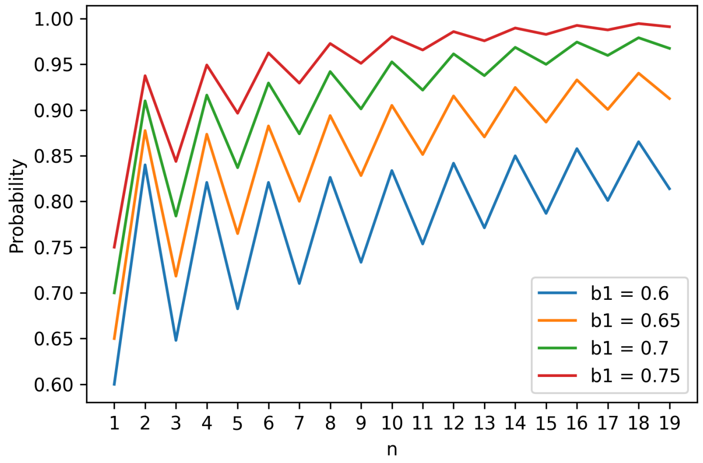

Figure 3.1: The probability of a Borda winner proceeding to the next phase with different Borda scores.

3.3.5 Mohajer et al.

Mohajer et al. [63] explore the active top-k selection and ranking problem based on noisy pairwise comparisons. To handle the special case of finding the best item, they propose a customized single-elimination tournament method referred to as SELECT. SELECT randomly pairs items and repeatedly compares them a fixed number of times, denoted by m, in order to combat against noise. One of the items in each pair can proceed to the next round based on majority voting, while the other item is eliminated. The tournament process is repeated multiple times until a single arm remains.

The SELECT algorithm returns exactly one item with any choices of m, but the algorithm performance depends on the value of m. The authors show that if m is sufficiently large, more specifically, if

$$m \geq \frac{(1+\epsilon)\ln 2}{2} \frac{\log\log K}{\min_{j\neq 1} \Delta_{1,j}^2},$$

Figure 3.2: Function **prefBest**(Pool, K, n, m) returns the probably best item from a pool of K items based on Clarke et al. [20]. The function **preference**(u, v) returns the item preferred by a human assessor or any other source of preferences.

Input

Pool	Set of items to judge
K	Pool size
n	Pairings during pruning phase
m	Threshold for completion phase

Output

Probably best item in Pool

```
def randomPairings(Pool, $K$, $n$):
    Generate and return the edge set of a random graph, where the $K$ items in the pool form the vertices
    of the graph and each vertex has outdegree $n$ (or $n + 1$).

def completePairings(Pool, $K$):
    Generate and return the edge set of a complete graph, where the $K$ items in the pool form the
    vertices of the graph (each vertex has outdegree $K - 1$).

def estimateBorda($E$):
    $\mathbf{W} \leftarrow \mathbf{0}$    #count of wins for each item
    $\mathbf{C} \leftarrow \mathbf{0}$    #count of pairings for each item
    for ($u$,$v$) in $E$:
        $w \leftarrow$ preference$(u, v)$
        $C_u \leftarrow C_u + 1$
        $C_v \leftarrow C_v + 1$
        $W_w \leftarrow W_w + 1$
    return $\mathbf{W}/\mathbf{C}$

def prune(Pool, $K$, $n$):
    $E \leftarrow$ randomPairings(Pool, $K$, $n$)
    $B \leftarrow$ estimateBorda($E$)
    Pruned $\leftarrow \emptyset$
    for $b_i$ in $B$:
        if $b_i \geq 0.5$:
            Pruned $\leftarrow$ Pruned $\cup \{i\}$
    return Pruned

def finalize(Pool, $K$):
    $E \leftarrow$ completePairings(Pool, $K$)
    $B \leftarrow$ estimateBorda($E$)
    return $\{j \mid b_j = \max_{i \in B} b_i\}$

def prefBest(Pool, $K$, $n$, $m$):
    while $K > m$:
        Pool $\leftarrow$ prune(Pool, K, $n$)
        $K \leftarrow |\text{Pool}|$
    return finalize(Pool, $K$)
```

then with high probability the SELECT algorithm can find the top-1 item using a total of

$$O(\frac{K \log \log K}{\min_{j \neq 1} \Delta_{1,j}^2})$$

comparisons.

In order to find the top-k items where $k > 1$, the authors propose a heap-based algorithm referred to as TOP. TOP makes use of SELECT and proceeds as follows: 1) split K items into k sub-groups and apply SELECT within each sub-group to find k winners. 2) Heap sort the shortlist of the k winners from sub-groups and extract the top item from the list. 3) Apply SELECT in the sub-group to which the top item belongs to identify the second best item, and add the second best item to the shortlist. Then repeat step 2 to find the next best item. Steps 2) and 3) are repeated until top-k items have been retrieved. The sample complexity of finding the top-k items is

$$O(\frac{(K + k \log k) \max\{\log k, \log \log K\}}{\min_{j \neq 1} \Delta_{1,j}^2})$$

3.3.6 Active Ranking

Heckel et al. [36] consider the problem of partitioning a set of K items into sub-groups of predefined size according to the probability that a given item beats a randomly selected item, i.e. the Borda score. The identification of the top-k items or a total order over items can be viewed as a special case. The authors propose the Active Ranking (AR) algorithm which is a successive elimination strategy. At each round, the method maintains a set of active arms and compares each arm with another randomly selected arm. The number of comparisons won is counted, and estimates of the Borda scores together with associated confidence intervals are computed based on the counts. An item is removed from the active set if the AR algorithm is confident that the item belongs to a certain pre-defined set. The AR algorithm does not make any particular structural assumptions on the underlying pairwise preference probabilities. It is shown that imposing parametric assumptions can lead to at most a logarithmic reduction in sample complexity.

3.3.7 PARWiS

Based on the review and analysis so far, while multiple algorithms enjoy linear or quasilinear theoretical guarantees in terms of the number of arms K, there are usually large constants hiding inside the order notation, resulting in a large number of total comparisons. To address this challenge, Sheth and Rajkumar [72] study the winner selection problem under a shoestring comparison budget. Given a set of K arms $[K] = \{1, ..., K\}$, the authors consider the Bradley-Terry-Luce(BTL) model for pairwise probabilities: assuming that each arm $i \in [K]$ has a real-valued score w_i, then for all i, j the probability of i being preferred to j is calculated as:

$$p_{i,j} = P(i \succ j) = \frac{w_i}{w_i + w_j}.$$

The goal is to find the item with the highest BTL score by actively performing at most B pairwise comparisons.

The proposed algorithm, PARWiS, is based on a spectral ranking procedure that constructs and maintains a Markov chain. The Markov chain is used to strategically select the next pair of items being compared, and to recover an approximate BTL score vector. PARWiS proceeds as follows. First, two randomly selected arms are compared once, with the winner proceeding to the next comparison with another random arm. The process is repeated until all K items are compared, resulting in $K - 1$ pairwise comparisons. The $K-1$ comparison results are used to initialize a Markov chain. In the next phase, PARWiS compares the current winner with the item that is most likely to disrupt the current winner according to the Markov chain and an associated objective function. In the paper, the authors experiment with several objective functions including the log-likelihood function, the least squares, and an objective function that computes the stationary distribution of the Markov chain. For more information, see [72]. After every comparison, the Markov chain is updated accordingly, until the comparison budget is exhausted. Estimated BTL scores are then computed from the Markov chain to identify the best item. When there are multiple winners, the algorithm is designed to find any one of them. The authors perform a set of experiments to show that PARWiS returns the top item effectively with shoestring budgets. However, no theoretical guarantee is provided.

Chapter 4

Simulations

In this chapter, I simulate the selected algorithms on two artificial test cases in order to understand their ability to satisfy the requirements of Section 3.1 and test their practical utility. The two test cases are deliberately designed to take into account both the common algorithm requirements in the dueling bandits literature and real-life examples of human preference judgments.

Case A assumes a total order over items with no ties. Although this assumption usually is not satisfied in human preference judgments due to the possible existence of many ties, I include this test case to examine whether the selected algorithms are practically usable in the best-case scenario. In other words, when theoretical assumptions are satisfied, I am interested to see if the number of assessments and independent judges required to identify the best item is feasible, not going over the budget. I expect that Case A could be handled easily by most dueling bandit algorithms since it satisfies the most common requirements of these algorithms. The other test case (Case B) represents something close to the opposite extreme. I assume that there are many ties and multiple "best" arms. Best arms tie with each other. None-best arms tie with each other. Best arms win non-best arms with a probability greater than 0.5. Both cases assume $K = 100$, with items $0, 1, 2, ...99$:

Case A (total order, no ties):
$i < j \implies q_{i,j} = 0.75$ and $q_{j,i} = 0.25$

Case B (two winners, many ties):
$q_{0,1} = q_{1,0} = 0.5$
$i > 1 \implies q_{0,i} = 0.75$ and $q_{i,0} = 0.25$
$i > 1 \implies q_{1,i} = 0.75$ and $q_{i,1} = 0.25$

	Case A			Case B			
	total order, no ties			two winners, many ties			
Algorithm	best found	comparisons	assessors	one found	both found	comparisons	assessors
PARWiS [72]	537	1000	17-68	686	287	1000	7-83
Active Ranking [36]	175	1000	3-11	405	545	1000	3-11
Mohajer et al. [63] (SELECT)	715	990	10	905	0	990	10
Mohajer et al. [63] (TOP-2)	-	-	-	588	277	980	5
Clarke et al. [20] (extra final phase)	510	624-781	3-6	733	81	616-795	3-6
Clarke et al. [20]	502	599-759	2-5	666	94	592-764	2-5
Round-Efficient Dueling Bandits [54]	141	1000	2-5	680	89	1000	2-5
Double Thompson Sampling [84]	124	1000	4-24	710	7	1000	4-20
MergeDTS [50]	108	1000	31-93	581	58	1000	28-81

Figure 4.1: Summary of $1,000$ simulations of selected algorithms on two artificial test cases intended to be representative of two possible extremes.

$$i > 1 \text{ and } j > 1 \implies q_{i,j} = 0.5$$

The two test cases are picked for their simplicity and as a contrast to one another. The value of $K = 100$ is close to the limit for the number of arms in the experiments of Chapter 5. 0.75 is picked as a “winning” probability since it corresponds to the level at which we tolerate test case errors in the experiments with crowdsourced assessors (Chapter 5). In the dueling bandits literature, many algorithms require a pre-defined time horizon as a parameter. Such time horizons can be viewed as a limit on the number of preference judgments required to identify the best arm. I choose $T = 1000$ as a practical limit on the number of judgments for $K = 100$ items. In any case, I treat $T = 1000$ as a fixed budget and halt simulations after $1,000$ comparisons. The simulations use the original code when authors have made it available, maintaining the default values of parameters in that code. When code is not available, I base simulations on my own implementations from pseudo-code and descriptions in the papers.

I executed each of the selected algorithms $1,000$ times on each of the two cases. The results are summarized in Figure 4.1, ordered by the number of simulations in Case B where at least one winner is found. Simulation details and a discussion for the individual algorithms are provided in the subsections that follow. Algorithms ran until their budget for comparisons was exhausted. I determined the maximum number of independent assessors according to the number of times the simulation requested a judgment for each specific pair. The maximum number of assessors also varied from simulation to simulation, and I report a range.

4.1 Double Thompson Sampling

Li et al. [50][1] provide implementations and optimal parameters for both DTS and MergeDTS. The two algorithms are originally designed for regret minimization tasks and have a large time horizon such as $T = 1e7$. The return statement in their code returns the cumulative regret. I modified the return statement to return the empirically best arm when the number of comparisons reached 1000. In the simulations, I set $\alpha = 0.8^7$, which they report as the empirically optimal parameter for DTS [50]. α is an exploration parameter that controls the trade-off between exploitation and exploration.

For Case A where there exists a total order and no ties, 959 simulations returned exactly one item, out of which 124 simulations correctly identified the optimal item 0. In addition, 922 items that were not item 0 were returned. For Case B with two winners and many ties, 710 simulations returned one of the correct answers, and 7 simulations found both winners. Across all simulations, 357 items that were neither item 0 nor item 1 were returned.

4.2 Merge Double Thompson Sampling

Similar to DTS, MergeDTS requires an exploration parameter α. Additionally, it needs a partition size M and an exploration bonus C that is added to the confidence intervals. Li et al. [50] extensively analyzed the sensitivity of parameter and suggest the practically optimal parameters for MergeDTS are $\alpha = 0.8^6$, $M = 16$ and $C = 4,000,000$.

I follow the same setting in our simulations. Among 1000 simulations for Case A, MergeDTS found the correct target item 0 108 times and returned 1503 items that were not item 0. For Case B, 581 simulations returned one of the two winners while 58 simulations found both. Across all simulations, 914 items that were neither item 0 nor item 1 were returned.

4.3 Round-Efficient Dueling Bandits

I implemented the Round-Efficient Dueling Bandits of finding the optimal arm based on the pseudo-code in Lin and Lu [54]. The implementation can be found in this repository[2]. In the simulations, I set the allowed error probability $\delta = 0.2$.

[1] github.com/chang-li/MergeDTS

[2] github.com/XinyiYan/duelingBandits

Of 1000 simulations for Case A, the algorithm returned one item 559 times, where 141 times it found the optimal arm; 1632 items that were not item 0 were returned. For Case B, 680 simulations returned one of the two optimal arms, and 89 simulations found both arms; 606 items that were neither item 0 nor item 1 were returned during the simulations for Case B.

4.4 Clarke et al.

The simulations use the implementation of Clarke et al. [20][3] for these simulations. The code is run "as is", replacing their example judging script with scripts that implement Case A and Case B. Since the algorithm is intended for precisely our scenario, no changes to the implementation were required.

For both cases, nearly half of the simulations returned two or more items tied for best — 497 for Case A and 489 for Case B. For Case A, 502 simulations of the 1000 simulations correctly returned item 0, while 323 simulations returned item 1 and 213 simulations returned item 2. In total, the simulations for Case A returned 995 items that were not item 0. For Case B, 94 simulations returned both item 0 and item 1, while 666 returned one or the other, but not both. In addition, the simulations for Case B returned 729 other items, which were neither item 0 nor item 1.

Given the large number of ties produced by the algorithm "as is", I attempted to address the problem by adding an extra final phase. This extra final phase judges all pairs in the final pool a second time so that the algorithm bases its final estimates on two comparisons of each pair. From a practical standpoint, an extra final phase requires only an extra independent assessment for each pair.

With an extra final phase, there are fewer ties and incorrect results. For Case A, where there is a single best item, 290 simulations return one or more items tied for best. Of the 1000 simulations, 510 correctly returned item 0, while 302 simulations returned item 1 and 166 simulations returned item 2. In total, the simulations for Case A returned 780 items that were not item 0. For Case B, 81 simulations returned both item 0 and item 1, while 733 returned one or the other, but not both. In addition, the simulations for Case B returned 430 other items, which were neither item 0 nor item 1.

[3] github.com/claclark/preferences

4.5 Mohajer et al.

Mohajer et al. [63][4] provide their MATLAB implementations for finding the top-1 and top-k items. I re-implemented the two methods in Python with very few modifications. For Case A where there is a single winner, I apply the top-1 selection method SELECT. Since the algorithm compares every pair m times and we want to find the best item out of 100 items within 1000 comparisons, I set m to be 10, resulting in 990 comparisons in total. Among the 1000 simulations for Case A, the SELECT algorithm found the correct target 715 times, outperforming all the other selected methods. In addition, it correctly returned one of the two winners in Case B 905 times. However, the SELECT method can not inherently find more than one winner. To find both winners in Case B, I apply the top-2 selection method which needs an extra phase of heap-sorting. In order to keep the total number of comparisons below the budget, I set m to be 5 for the top-2 selection method, resulting in 980 total comparisons. 588 simulations returned one of the two winners. 277 simulations returned both winners.

4.6 Active Ranking

Heckel et al. [36][5] implement a variant of the top-k ranking algorithm proposed in the paper. While the algorithm in the paper uses a successive elimination strategy, their implementation is based on the LUCB (lower upper confidence bound), but shares the same sample complexity guarantees as they report. I use their implementation for the simulations and restrict the total number of comparisons to 1000. I set the allowed error probability $\delta = 0.1$, following their setup. For Case A, I set $k = 1$ and 175 simulations correctly returned the optimal item. For Case B, I set $k = 2$. 405 simulations returned one of the correct answers and 545 simulations found both winners.

4.7 PARWiS

Sheth and Rajkumar [72][6] provide implementations for the PARWiS algorithm. I modify their code to take a preference probability matrix as an input instead of a BTL scores vector. For all other input parameters such as the objective function, I use their default

[4] github.com/a-elmahdy/Active-Learning-from-Noisy-Comparisons.git
[5] github.com/reinhardh/supplement_active_ranking
[6] github.com/DevSheth/active-ranking-under-shoe-string-budget

values. The shoestring budget is set to be 1000. The algorithm returns the top winner and a ranking over all items according to BTL scores.

Among the 1000 simulations for Case A, PARWiS successfully recovered the optimal item 537 times. The range of the maximum number of independent assessors is $[17, 68]$. For Case B, 686 simulations recovered one of the two winners, with the maximum number of independent assessors in the range of $[7, 83]$. Since PARWiS returns a ranking and there are two winners in Case B, I extract the item with the second highest BTL score from the ranking and compare it with the true winners. 287 simulations recovered both of the winners.

4.8 Discussion

In contrast to the previous work presented at SIGIR 2022 [87], I considered more algorithms in this thesis and further expanded the simulation results. All algorithms performed fairly reasonably on Case B, with between 63.9% and 97.3% of simulations finding at least one or two of the "winners". All algorithms performed less reasonably on Case A, which I had assumed would be the easier case, with between 10.8% and 71.5% finding the single "winner". In our previous SIGIR paper [87], I determined that only Clarke et al. [20] fully satisfied the three requirements. Based on the analysis and simulations reported in this work, the tournament method of Mohajer et al. [63] and the heuristic approach of Clarke et al. [20] both appear to be suitable for human preference judgments.

While PARWiS [72] performed reasonably well on both test cases, the large number of independent assessors required is a major obstacle. Between the two suitable algorithms, the SELECT method [63] compares each pair a fixed number of times, and only returns a single item. It successfully recovered the true winner or one of the true winners more often than the other method [20] did. However, if we want to find more than one winner, SELECT needs to be extended to a heap-based algorithm, requiring extra rounds of comparisons. In order not to run out of the comparison budget, each pair will be compared fewer times, at the expense of the algorithm performance. The approach of Clarke et al. [20] demonstrates practical value but lacks an explicit theoretical foundation beyond what I did in Section 3.3.4. I proceed with the algorithm of Clarke et al. [20] and leave the method of Mohajer et al. [63] as a potential future work. Dueling bandit algorithms are sensitive to parameterization. I made efforts to adapt the other algorithms by adjusting parameters, but without success.

Chapter 5

Human Preference Judgments

Based on the foregoing analysis, we proceed with the algorithm of Clarke et al. [20], which provides a feasible solution to the requirements of Section 3.1. We apply the algorithm to produce human preference judgments for the passage retrieval task of the TREC 2021 Deep Learning track [22]. The crowdsourced Mechanical Turk judgments reported in this section were supervised by Chengxi Luo, as reported in the joint paper [87]. We focus the experiments on further validation of the algorithm, on opportunities for improvement, and on exploration of sparse labels for information retrieval evaluation, providing support for ideas proposed by Arabzadeh et al. [6].

5.1 Method

We applied the algorithm of Clarke et al. [20] to crowdsource preference judgments for pools generated from runs submitted to the passage retrieval task of the TREC 2021 Deep Learning Track [23]. For these experiments we worked with an initial pool of 50 queries and provided by the track organizers. During their own judging process they added seven queries and dropped four, so that the overlap is 46 queries [73]. Participants in the track submitted 63 runs to the passage retrieval task. Each run comprised a ranked list of up to 100 passages intended to answer each of 477 questions. TREC assessors judged these queries on a four-point relevance scale (“perfect”, “highly relevant”, “related”, “irrelevant”) [73]. The query in Figure 1.1 (#1103547) was one of the dropped query because it has no highly relevant or perfect passages.

We based our experiments on the implementation provided by Clarke et al. [20] in the associated repository[1]. Apart from the addition of a second finalization phase, we followed the crowdsourcing procedure described in that paper as closely as possible, using the code without change, including default parameters. We implemented the second finalize phase by requesting two judgments for each pair in the final pool through the crowdsource platform, computing scores in a separate script from both the judgments in the final phase of the original implementation and our extra finalization phase. Apart from this additional finalization phase, our procedure is exactly **prefBest**(Pool, K, 7, 9) as listed in Figure 3.2, where $n = 7$ and $m = 9$ are the default values in the original implementation.

We started with depth-10 pools generated from the runs submitted for the passage retrieval task, along with their pointwise graded judgments. The implementation builds its own internal pools from the three grades of relevant passages, omitting irrelevant passages, "top down" until a minimum threshold is reached or until all relevant passages are included. For our experiments, we maintained the default value of 5 for this threshold. The implementation first adds all perfectly relevant passages to its pool, if the threshold is not reached, it then adds all highly relevant and then all relevant passages until the threshold is reached or there are no relevant passages remaining. For the 50 questions, preference judging pools ranged in size from 5 to 130, with a median size of 15 and a mean size of 31.4.

We used Amazon Mechanical Turk (MT) to crowdsource judgments. We used the judging interface described in [20] as the starting point, modifying it to simplify data management and improve performance. Each judging task comprises ten target pairs — for which we need judgments — and three test pairs. These test pairs were intended to reduce judging errors due to misunderstandings, distractions, or the absence of assessor training, as discussed in Section 3.1. Each test pair consisted of a question, a "best-known answer" passage, and an off-topic passage. These questions and best-known answers were taken from the collection developed by Clarke et al.[20], as provided in their github repository. Target pairs and test pairs were randomly assigned to tasks. Within each pair, the left and right placement of passages was determined randomly. For judging purposes, we edited some of the 50 questions to correct obvious grammatical and other errors. We included these edited questions in the data release for this paper.

We required crowdsourced workers to have a task approval rate greater than 95%, and to have more than 1,000 approved tasks. We required workers to be located in the United States, consistent with the context of these TREC experiments. We paid workers \$2.00 for completing each task, consistent with the expected time to complete a single task and our local minimum wage. In addition, we paid a platform fee of \$0.40 to Amazon MT for

[1] github.com/claclark/preferences

each task. Under our research protocol, workers could stop after partially completing a task and receive a prorated amount, but none did. While we were debugging the performance of our system a small number of workers experienced unacceptable delays and were paid compensation. Our research protocol received ethics approval from the University of Waterloo.

Data from workers who did not correctly answer at least 75% of the test pairs was excluded. This requirement means that workers completing a single task were required to correctly answer all three test pairs; workers completing two tasks were required to correctly answer at least five of the six test questions. We paid these workers for their work, but they were excluded from participating in later phases. The total cost for all judgments was $3,830.41, including interface debugging tasks, excluded data, and platform fees. This produced a total of 10,697 usable judgments at a cost of just under 36 cents each. Of these, 9,713 judgments were required by the algorithm in Section 3.3.4 and an additional 984 judgments were required for our extra finalization phase.

5.2 Exact duplicates

After running the entire experiment, including the extra finalization phase, we discovered that the test collection used for the TREC Deep Learning passage retrieval task (the MS MARCO v2 dataset) contained exact duplicates – passages that were character-for-character identical. Of the 1570 passages in our combined judging pools, 460 passages contained at least one duplicate in the same pool. If we keep just one of these duplicates, it would leave 1308 items in the pool.

If we had encountered these duplicates earlier in the experiment, it is likely that we would have merged them into equivalence classes, so that assessors would not have to choose between absolutely identical passages. However, we did not notice them until we were looking at top results to select examples for the joint paper[87] and realized that the top 5 of question #364210 included an identical copy of the top answer, which was tied for second. We did not see any identical pairs during our interface tests, perhaps because comparisons between exact duplicates are relatively rare, despite the relatively large number of them. Excluding the extra finalization phase, only 169 (1.7%) of the 9,713 judgments required for our experiments are between identical elements.

As a result, we have an unplanned test of our equality tolerance requirement. If the top result for a question has an exact duplicate, it should also appear in the top results. Comparing the top results from the main experiment (excluding the extra finalization phase), the top document for 15 questions had a duplicate in the pool. If we exclude five questions where the pool size is less or equal to $m = 9$, where there were no pruning phases, the top document for ten questions had at least one exact duplicate in the pool. The pool sizes for these questions range from 11 to 130, with an average size of 50.5.

For eight of these ten questions, the top passage had a single exact duplicate in the pool. For six of these eight questions, the duplicate also appeared in the top-5, usually ranked second or third. For one of the eight questions, the duplicate reached the upper half but did not reach the finalization phase. For the remaining question, the duplicate was eliminated during the initial pruning phase.

For one question (#629937) the top passage had two duplicates in the pool, which were ranked third and fourth. One question (#508292) had two duplicate passages tied for first. These passages had two other duplicates in their pool of 103 passages. One of these duplicates was ranked second. The other made it into the top half but was eliminated during the second pruning phase.

Overall, 9 of 12 duplicates for these ten questions (75%) reached the top-5, a result consistent with our expectations from the analysis and simulations in previous sections. This outcome might be improved by increasing the value of n to decrease the number of top results lost during the pruning phases. As future work, we hope to extend our theoretical understanding of the approach to choose the number of comparisons during the pruning phase more dynamically, based on the data itself [43].

5.3 Extra finalization phase

The simulations suggested that an extra finalization phase would reduce the number of ties and incorrect results. To test this observation, for each question we requested a second round of assessments for the pairs in the final phase. This extra finalization phase gives us three sets of best items. Set I from the first round, Set II from the second round, and a Combined Set constructed by merging the judgments from both finalization phases and computing scores from the combination.

For the 50 questions, both Set I and Set II had 75 best passages — an average of 1.5 passages per question. The Combined Set had fewer ties, with 66 best passages – an average of 1.32 passages per question. While the exact distributions of ties differ between Set I and Set II, both have (different) questions with 4 passages tied for best, while the Combined Set has none.

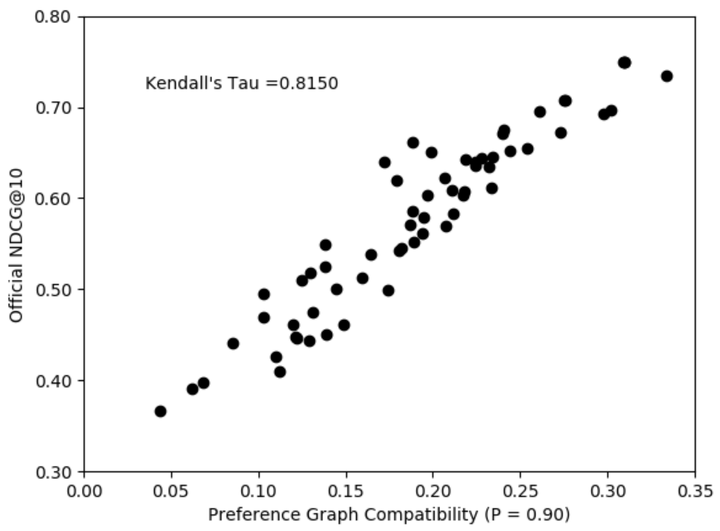

Figure 5.1: Scatter plot comparing experimental runs submitted to the passage retrieval task of the TREC 2021 Deep Learning track on pointwise graded judgments vs. pairwise preference judgments. Each point represents a single run. Official TREC runs use slightly different questions and relevance judgments than our initial pools [73].

Comparing the top passages in Set I and Set II, only 22 of the topics share at least one best answer, and the intersection of Set I and Set II contains only 25 passages. Even though the Combined Set is smaller, 40 of the topics in the Combined Set share a top answer with Set I; 39 of the topic in the Combined Set share a top answer with Set II. The Combined Set shares 44 passages with both Set I and Set II.

Although it is perhaps not surprising, the Combined Set has fewer ties and is more consistent with both Set I and Set II than they are with each other. We report the experiments in the next section on the Combined Set. As future work, we hope to discover and validate more a principled method for the finalization phase.

5.4 Best item evaluation

As discussed in Chapter 1, one of the motivations to identify best items is to evaluate IR systems by their ability to place these items as high as possible. With the algorithm of Clarke et al. [20], we crowdsourced preference judgments for the passage retrieval task of

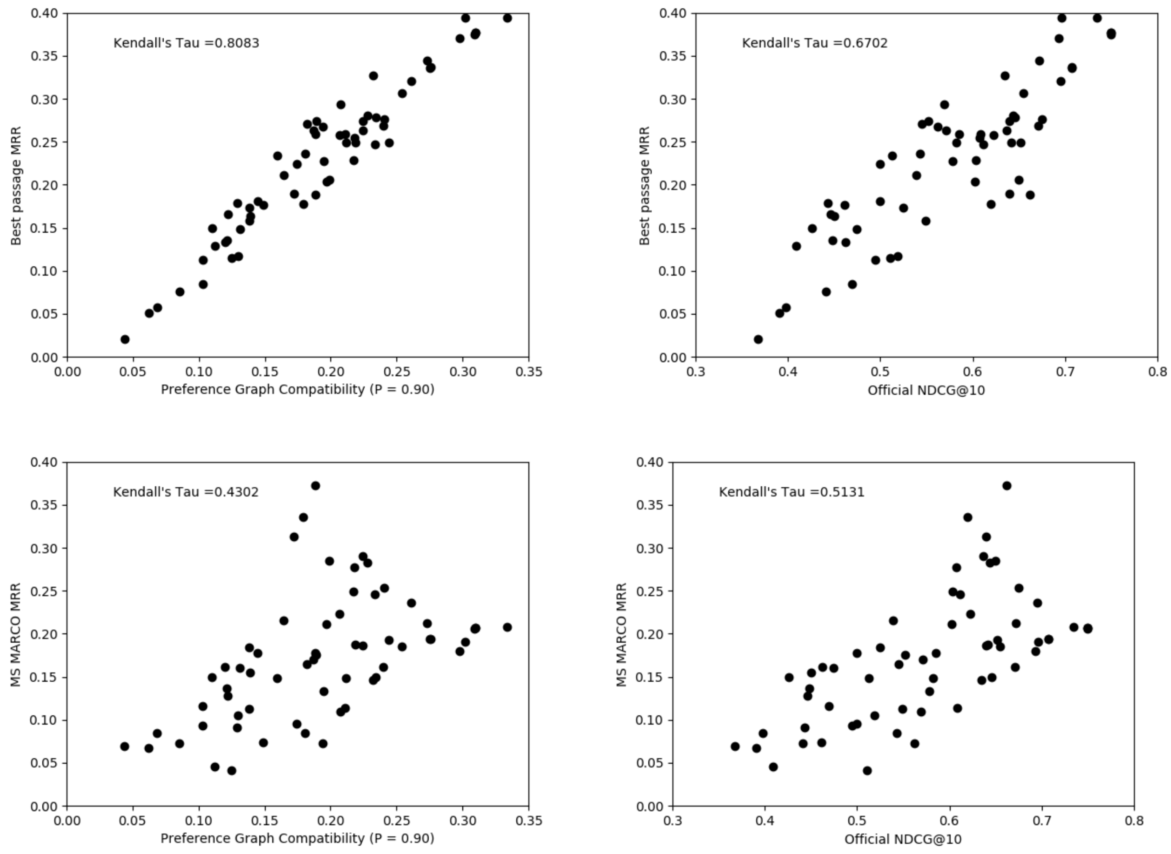

Figure 5.2: Scatter plots comparing sparse binary judgments against graded judgments and preference judgments for experiments runs submitted to the passage retrieval task of the TREC 2021 Deep Learning Track. Each point represents a single run. Official TREC runs use slightly different questions and relevance judgments than our initial pools [73].

the TREC 2021 Deep Learning Track [23], and identified best responses to the queries. In this section, we aim to further test the idea of best item evaluation using our crowdsourced preference judgments in order to create more complete stories about this topic. We compare best item evaluation with traditional graded evaluation methods and MS MARCO sparse labels to provide more insight.

Traditional evaluation approaches in information retrieval aim for judgments to be as complete as possible. Experimental runs are pooled as deeply as possible, and efforts are made to judge as many items as possible. The goal is to create a re-usable collection so that future experiments conducted on the collection provide meaningful results, even if these new experimental runs surface unjudged documents in their top ranks [10, 70]. NDCG is one of the common measures used in traditional pointwise graded evaluation. However, this traditional evaluation methodology faces several challenges. For instance, TREC has struggled with the problem of completeness for most of its history, with the overview paper for TREC 2021 providing an excellent summary [73]. Some TREC tracks, including the Deep Learning Track, use active learning methods [1] to expand the set of known relevant documents. Unfortunately, despite these efforts, "Most topics have very many relevant . . . and as a result the collection is likely not a reusable collection." [73].

The MS MARCO leaderboards[2] represent a radical departure from traditional information retrieval evaluation methodology. Over the past three years, the MS MARCO leaderboards have tracked dramatic improvements in core information retrieval tasks, including passage retrieval [21]. MS MARCO bases evaluation on sparse labels, with 94% of the questions in the development set for passage ranking having only a single known relevant passage [6]. Given this sparsity of relevance labels, mean reciprocal rank (MRR) forms the primary evaluation measure. While the sparse labels are not intended to represent the "best" passages [22], better systems will tend to place these passages higher in their rankings. This overall approach has been validated through comparisons with Deep Learning Track results in 2019 and 2020 [21].

Arabzadeh et al. [6] suggest that if the sparse labels represented the best known items, the long term validity of the approach might be further improved. We further suggest that there may be fewer concerns about the re-usability of a collection based on best known items. If a new experiment ranks an unjudged item above the best known item, the unjudged item only invalidates measures if it is better than all previously judged items, a much higher bar than mere relevance.

To further explore the validity of best answer evaluation, we compare best item evaluation with measures computed over all available judgments. For our comparison with

[2]microsoft.github.io/msmarco/

graded judgments, we use the official TREC NDCG@10 score. We also compare with Preference Graph Compatibility (PGC) [18], which is computed directly from a multi-graph of preferences. Figure 5.1 plots NDCG against PGC, computed over the 63 runs submitted to the passage retrieval task of the TREC 2021 Deep Learning Track [22]. We compute PGC using the entirety of the preference judgments collected by our experiments, including the extra finalization phase. A Kendall's τ value of 0.8150 indicates the consistency of the preference judgments with the pointwise judgments, suggesting that the pairwise judgments are of sufficient quality to form the basis for measurement.

We create the top plots in Figure 5.2 based on the Combined Set of preference labels from Section 5.3 to test the idea of best item evaluation. MRR values are computed using best known responses, and compared with PGC and NDCG. Despite the sparsity of these preference labels, the ordering of runs is correlated with PGC, with a Kendall's τ over 0.8. Both of the top plots show a generally consistent ordering of runs.

As a comparison, we create the bottom plots in Figure 5.2 using MS MARCO sparse labels. Questions for the TREC 2021 Deep Learning Track were taken from questions held out from MS MARCO data releases [22], so that MS MARCO sparse labels are available for these questions. Craswell et al. [22] report MRR values using these labels, which we use to create the bottom plots. With a Kendall's τ under 0.5, MRR on these labels is much less correlated with our preference judgments. The top run on the MS MARCO labels (`p_f10_mdt53b`) places just below the median on PGC. The group submitting this run is known for it experience and expertise on the MS MARCO collection [55], expertise which may have inadvertently engendered poorer performance on the preference-based labels.

As a final experiment, we created random sparse labels based on the official graded relevance judgments for the Deep Learning Track. For each of the 50 questions, we pooled the passages with the highest relevance grade for that question. From each pool, we picked a single random relevance passage as the sparse label for that question, and computed MRR for each run using these sparse labels. We repeated this process 1,000 times. For 95% of these trials, the Kendall's τ correlation between MRR and NDCG@10 fell between 0.485 and 0.735. Kendall's τ with the MS MARCO labels (0.5131) is closer to the lower end of this range, while Kendall's τ with the preference labels is closer to the upper end. On the 1,000 trials, the best run achieved an MRR value between 0.120 and 0.219. On the MS MARCO labels `p_f10_mdt53b` achieves an MRR of 0.3728, while on the preference labels `NLE_P_v1` achieves 0.3946, both well above the best MRR values on random labels. This outcome suggests that neither the MS MARCO labels nor the preference labels are arbitrary relevant passages. In general, these passages are preferred by the rankers themselves, where rankers heavily tuned on MS MARCO appear to prefer MS MARCO passages.

Chapter 6

Conclusion

Neural rankers push the limits of traditional information retrieval evaluation methodologies. If most items returned by a ranker are relevant in the traditional sense, how do we measure further improvements? A possible answer is to focus on the top items, the best of the best [6]. A better ranker would place these top items higher in its ranking. This thesis explores dueling bandits as a framework for finding these items via human preference judgments, and is built on top of our previous work presented at SIGIR 2022 [87] by considering additional algorithms and generating more experimental results.

After defining requirements in Section 3.1, I reviewed dueling bandits algorithms in the research literature. While many algorithms have proven their practical value in the context of online ranker selection, they have not been applied in the context of human preference judgments. Instead of excluding them at a purely theoretical level, two simple test cases in Section 4 are designed to examine the practical performance of several candidate algorithms selected based on their potentials. While I determined that only Clarke et al. [20] satisfied the requirements for human preference judgments in the previous work [87], as an outcome of the review and simulations reported in this thesis, both the heuristic approach of Clarke et al. [20] and the tournament-based method of Mohajer et al. [63] satisfy the requirements. I proceed with the algorithm of Clarke et al. [20] and leave the other method for future work.

Using the algorithm of Clarke et al. [20], we crowdsourced preference judgments for the passage retrieval task of the TREC 2021 Deep Learning Track, identifying the probably best answer for each of the 50 question in the evaluation set. These experiments suggest that the algorithm meets our requirements, particularly the requirement of ties being tolerated. We demonstrate the practical value of an extra finalization phase. These experiments also suggest that evaluation can be based on the best known items alone, supporting the views of Arabzadeh et al. [6].

As potential future work of this thesis, I hope to focus on the opportunities for experimenting with and improving existing algorithms for human preference judgments, on analysis of theoretical bounds for the algorithm of Clarke et al. [20], and on further exploration of sparse labels for information retrieval evaluation.

Code, crowdsourced preference judgments and best answers are available in an associated repository:

https://github.com/XinyiYan/duelingBandits

References

[1] Mustafa Abualsaud, Nimesh Ghelani, Haotian Zhang, Mark D Smucker, Gordon V Cormack, and Maura R Grossman. A system for efficient high-recall retrieval. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 1317–1320, 2018.

[2] Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*, pages 39–1. JMLR Workshop and Conference Proceedings, 2012.

[3] Nir Ailon, Zohar Karnin, and Thorsten Joachims. Reducing dueling bandits to cardinal bandits. In *International Conference on Machine Learning*, pages 856–864. PMLR, 2014.

[4] Azzah Al-Maskari, Mark Sanderson, and Paul Clough. The relationship between IR effectiveness measures and user satisfaction. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 773–774, 2007.

[5] Luca Alfaro, James Davis, Hector Garcia-Molina, Neoklis Polyzotis, and Vassilis Polychronopoulos. Human-powered top-k lists. In *WebDB*, pages 25–30, 2013.

[6] Negar Arabzadeh, Alexandra Vtyurina, Xinyi Yan, and Charles LA Clarke. Shallow pooling for sparse labels. *Information Retrieval Journal*, pages 1–21, 2022.

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

[8] Ashraf Bah, Praveen Chandar, and Ben Carterette. Document comprehensiveness and user preferences in novelty search tasks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 735–738, 2015.

[9] Viktor Bengs, Róbert Busa-Fekete, Adil El Mesaoudi-Paul, and Eyke Hüllermeier. Preference-based online learning with dueling bandits: A survey. *J. Mach. Learn. Res.*, 22:7–1, 2021.

[10] Chris Buckley and Ellen M Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 25–32, 2004.

[11] Róbert Busa-Fekete, Eyke Hüllermeier, and Balázs Szörényi. Preference-based rank elicitation using statistical models: The case of mallows. In *International Conference on Machine Learning*, pages 1071–1079. PMLR, 2014.

[12] Róbert Busa-Fekete, Balazs Szorenyi, Weiwei Cheng, Paul Weng, and Eyke Hüllermeier. Top-k selection based on adaptive sampling of noisy preferences. In *International Conference on Machine Learning*, pages 1094–1102. PMLR, 2013.

[13] Ben Carterette and Paul N Bennett. Evaluation measures for preference judgments. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 685–686, 2008.

[14] Ben Carterette, Paul N Bennett, David Maxwell Chickering, and Susan T Dumais. Here or there. In *European Conference on Information Retrieval*, pages 16–27. Springer, 2008.

[15] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. *Advances in Neural Information Processing Systems*, 24, 2011.

[16] Xi Chen, Paul N Bennett, Kevyn Collins-Thompson, and Eric Horvitz. Pairwise ranking aggregation in a crowdsourced setting. In *Proceedings of the sixth ACM International Conference on Web Search and Data Mining*, pages 193–202, 2013.

[17] Charles LA Clarke, Maheedhar Kolla, Gordon V Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 659–666, 2008.

[18] Charles LA Clarke, Chengxi Luo, and Mark D Smucker. Evaluation measures based on preference graphs. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1534–1543, 2021.

[19] Charles LA Clarke, Mark D Smucker, and Alexandra Vtyurina. Offline evaluation by maximum similarity to an ideal ranking. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 225–234, 2020.

[20] Charles LA Clarke, Alexandra Vtyurina, and Mark D Smucker. Assessing top-preferences. *ACM Transactions on Information Systems (TOIS)*, 39(3):1–21, 2021.

[21] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Jimmy Lin. Ms marco: Benchmarking ranking models in the large-data regime. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1566–1576, 2021.

[22] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Jimmy Lin. Overview of the TREC 2021 deep learning track. In *30th Text REtrieval Conference. Gaithersburg, Maryland*, 2021.

[23] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, Ellen M Voorhees, and Ian Soboroff. TREC deep learning track: reusable test collections in the large data regime. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2369–2375, 2021.

[24] Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. TREC CAsT 2019: The conversational assistance track overview. *arXiv preprint arXiv:2003.13624*, 2020.

[25] Susan B Davidson, Sanjeev Khanna, Tova Milo, and Sudeepa Roy. Using the crowd for top-k and group-by queries. In *Proceedings of the 16th International Conference on Database Theory*, pages 225–236, 2013.

[26] Luca De Alfaro, Vassilis Polychronopoulos, and Neoklis Polyzotis. Efficient techniques for crowdsourced top-k lists. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 4, pages 22–31, 2016.

[27] Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th International Conference on World Wide Web*, pages 613–622, 2001.

[28] Moein Falahatgar, Yi Hao, Alon Orlitsky, Venkatadheeraj Pichapati, and Vaishakh Ravindrakumar. Maxing and ranking with few assumptions. *Advances in Neural Information Processing Systems*, 30, 2017.

[29] Moein Falahatgar, Ayush Jain, Alon Orlitsky, Venkatadheeraj Pichapati, and Vaishakh Ravindrakumar. The limits of maxing, ranking, and preference learning. In *International Conference on Machine Learning*, pages 1427–1436. PMLR, 2018.

[30] Moein Falahatgar, Alon Orlitsky, Venkatadheeraj Pichapati, and Ananda Theertha Suresh. Maximum selection and ranking under noisy comparisons. In *International Conference on Machine Learning*, pages 1088–1096. PMLR, 2017.

[31] Hans-Peter Frei and Peter Schäuble. Determining the effectiveness of retrieval algorithms. *Information Processing & Management*, 27(2-3):153–164, 1991.

[32] Dorota Glowacka et al. Bandit algorithms in information retrieval. *Foundations and Trends® in Information Retrieval*, 13(4):299–424, 2019.

[33] Aditya Gopalan, Shie Mannor, and Yishay Mansour. Thompson sampling for complex online problems. In *International Conference on Machine Learning*, pages 100–108. PMLR, 2014.

[34] Matthew Groves and Juergen Branke. Top-κ selection with pairwise comparisons. *European Journal of Operational Research*, 274(2):615–626, 2019.

[35] Stephen Guo, Aditya Parameswaran, and Hector Garcia-Molina. So who won? dynamic max discovery with the crowd. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 385–396, 2012.

[36] Reinhard Heckel, Nihar B Shah, Kannan Ramchandran, and Martin J Wainwright. Active ranking from pairwise comparisons and when parametric assumptions do not help. *The Annals of Statistics*, 47(6):3099–3126, 2019.

[37] Reinhard Heckel, Max Simchowitz, Kannan Ramchandran, and Martin Wainwright. Approximate ranking from pairwise comparisons. In *International Conference on Artificial Intelligence and Statistics*, pages 1057–1066. PMLR, 2018.

[38] Katja Hofmann, Shimon Whiteson, and Maarten De Rijke. Fidelity, soundness, and efficiency of interleaved comparison methods. *ACM Transactions on Information Systems (TOIS)*, 31(4):1–43, 2013.

[39] Kai Hui and Klaus Berberich. Low-cost preference judgment via ties. In *European Conference on Information Retrieval*, pages 626–632. Springer, 2017.

[40] Kevin Jamieson, Sumeet Katariya, Atul Deshpande, and Robert Nowak. Sparse dueling bandits. In *Artificial Intelligence and Statistics*, pages 416–424. PMLR, 2015.

[41] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[42] Thorsten Joachims. Evaluating retrieval performance using clickthrough data. In *Text Mining*, pages 79–96, 2003.

[43] Zohar Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. In *International Conference on Machine Learning*, pages 1238–1246. PMLR, 2013.

[44] Asif R Khan and Hector Garcia-Molina. Hybrid strategies for finding the max with the crowd: technical report. Technical report, Stanford InfoLab, 2014.

[45] Junpei Komiyama, Junya Honda, Hisashi Kashima, and Hiroshi Nakagawa. Regret lower bound and optimal algorithm in dueling bandit problem. In *Conference on learning theory*, pages 1141–1154. PMLR, 2015.

[46] Junpei Komiyama, Junya Honda, and Hiroshi Nakagawa. Optimal regret analysis of thompson sampling in stochastic multi-armed bandit problem with multiple plays. In *International Conference on Machine Learning*, pages 1152–1161. PMLR, 2015.

[47] Junpei Komiyama, Junya Honda, and Hiroshi Nakagawa. Copeland dueling bandit problem: Regret lower bound, optimal algorithm, and computationally efficient algorithm. In *International Conference on Machine Learning*, pages 1235–1244. PMLR, 2016.

[48] Wataru Kumagai. Regret analysis for continuous dueling bandit. *Advances in Neural Information Processing Systems*, 30:1488–1497, 2017.

[49] Nir Levine, Koby Crammer, and Shie Mannor. Rotting bandits. *Advances in Neural Information Processing Systems*, 30:3074–3083, 2017.

[50] Chang Li, Ilya Markov, Maarten De Rijke, and Masrour Zoghi. MergeDTS: A method for effective large-scale online ranker evaluation. *ACM Transactions on Information Systems (TOIS)*, 38(4):1–28, 2020.

[51] Kaiyu Li, Xiaohang Zhang, and Guoliang Li. A rating-ranking method for crowd-sourced top-k computation. In *Proceedings of the 2018 International Conference on Management of Data*, pages 975–990, 2018.

[52] Yan Li, Hao Wang, Ngai Meng Kou, Zhiguo Gong, et al. Crowdsourced top-k queries by pairwise preference judgments with confidence and budget control. *The VLDB Journal*, 30(2):189–213, 2021.

[53] Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

[54] Chuang-Chieh Lin and Chi-Jen Lu. Efficient mechanisms for peer grading and dueling bandits. In *Asian Conference on Machine Learning*, pages 740–755. PMLR, 2018.

[55] Jimmy Lin, Haonen Chen, Chengcheng Hu, Sheng-Chieh Lin, Yilin Li, Xueguang Ma, Ronak Pradeep, Jheng-Hong Yang, Chuan-Ju Wang, Andrew Yates, and Xinyu Zhang. New nails for old hammers: Anserini and pyserini at TREC 2021. In *30th Text REtrieval Conference. Gaithersburg, Maryland*, 2021.

[56] Michael L Littman. Reinforcement learning improves behaviour from evaluative feedback. *Nature*, 521(7553):445–451, 2015.

[57] David E Losada, Javier Parapar, and Álvaro Barreiro. Feeling lucky? multi-armed bandits for ordering judgements in pooling-based evaluation. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 1027–1034, 2016.

[58] Tyler Lu, Dávid Pál, and Martin Pál. Contextual multi-armed bandits. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 485–492. JMLR Workshop and Conference Proceedings, 2010.

[59] R Duncan Luce. *Individual choice behavior: A theoretical analysis.* Courier Corporation, 2012.

[60] Colin L Mallows. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130, 1957.

[61] Shie Mannor and John N Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5(Jun):623–648, 2004.

[62] Lucas Maystre and Matthias Grossglauser. Just sort it! a simple and effective approach to active preference learning. In *International Conference on Machine Learning*, pages 2344–2353. PMLR, 2017.

[63] Soheil Mohajer, Changho Suh, and Adel Elmahdy. Active learning for top-k rank aggregation from noisy comparisons. In *International Conference on Machine Learning*, pages 2488–2497. PMLR, 2017.

[64] Douglas W Oard, William Webber, et al. Information retrieval for e-discovery. *Foundations and Trends® in Information Retrieval*, 7(2–3):99–237, 2013.

[65] Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2):193–202, 1975.

[66] Wenbo Ren, Jia Liu, and Ness Shroff. The sample complexity of best-k items selection from pairwise comparisons. In *International Conference on Machine Learning*, pages 8051–8072. PMLR, 2020.

[67] Kevin Roitero, Alessandro Checco, Stefano Mizzaro, and Gianluca Demartini. Preferences on a budget: Prioritizing document pairs when crowdsourcing relevance judgments. In *Proceedings of the ACM Web Conference 2022*, pages 319–327, 2022.

[68] Mark E Rorvig. The simple scalability of documents. *Journal of the American Society for Information Science*, 41(8):590–598, 1990.

[69] Tetsuya Sakai. Evaluating evaluation metrics based on the bootstrap. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 525–532, 2006.

[70] Tetsuya Sakai and Noriko Kando. On information retrieval metrics designed for evaluation with incomplete relevance assessments. *Information Retrieval*, 11(5):447–470, 2008.

[71] Tetsuya Sakai and Zhaohao Zeng. Good evaluation measures based on document preferences. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 359–368, 2020.

[72] Dev Yashpal Sheth and Arun Rajkumar. PARWiS: Winner determination from active pairwise comparisons under a shoestring budget. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 569–578. IEEE, 2021.

[73] Ian Soboroff. Overview of TREC 2021. In *30th Text REtrieval Conference. Gaithersburg, Maryland*, 2021.

[74] Karen Spark-Jones. Report on the need for and provision of an'ideal'information retrieval test collection. *Computer Laboratory*, 1975.

[75] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[76] Balázs Szörényi, Róbert Busa-Fekete, Adil Paul, and Eyke Hüllermeier. Online rank elicitation for plackett-luce: A dueling bandits approach. *Advances in Neural Information Processing Systems*, 28, 2015.

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

[78] Tanguy Urvoy, Fabrice Clerot, Raphael Féraud, and Sami Naamane. Generic exploration and k-armed voting bandits. In *International Conference on Machine Learning*, pages 91–99. PMLR, 2013.

[79] Petros Venetis and Hector Garcia-Molina. Dynamic max algorithms in crowdsourcing environments. Technical report, Stanford InfoLab, 2012.

[80] Vasilis Verroios, Peter Lofgren, and Hector Garcia-Molina. tdp: An optimal-latency budget allocation strategy for crowdsourced maximum operations. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1047–1062, 2015.

[81] Ellen M Voorhees, Nick Craswell, and Jimmy Lin. Too many relevants: Whither cranfield test collections? In *Proceedings of the 45th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2022)*, 2022.

[82] William Webber, Alistair Moffat, and Justin Zobel. The effect of pooling and evaluation depth on metric stability. In *EVIA@ NTCIR*, pages 7–15, 2010.

[83] William Webber, Alistair Moffat, Justin Zobel, and Tetsuya Sakai. Precision-at-ten considered redundant. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 695–696, 2008.

[84] Huasen Wu and Xin Liu. Double thompson sampling for dueling bandits. *Advances in Neural Information Processing Systems*, 29:649–657, 2016.

[85] Yingce Xia, Haifang Li, Tao Qin, Nenghai Yu, and Tie-Yan Liu. Thompson sampling for budgeted multi-armed bandits. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[86] Xiaohui Xie, Jiaxin Mao, Yiqun Liu, Maarten de Rijke, Haitian Chen, Min Zhang, and Shaoping Ma. Preference-based evaluation metrics for web image search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 369–378, 2020.

[87] Xinyi Yan, Chengxi Luo, Charles LA Clarke, Nick Craswell, Ellen M Voorhees, and Pablo Castells. Human preferences as dueling bandits. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 567–577, 2022.

[88] Ziying Yang, Alistair Moffat, and Andrew Turpin. Pairwise crowd judgments: Preference, absolute, and ratio. In *Proceedings of the 23rd Australasian Document Computing Symposium*, pages 1–8, 2018.

[89] Y.Y. Yao. Measuring retrieval effectiveness based on user preference of documents. *Journal of the American Society for Information science*, 46(2):133–145, 1995.

[90] Yisong Yue, Josef Broder, Robert Kleinberg, and Thorsten Joachims. The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012.

[91] Yisong Yue and Thorsten Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1201–1208, 2009.

[92] Yisong Yue and Thorsten Joachims. Beat the mean bandit. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 241–248. Citeseer, 2011.

[93] Julian Zimmert and Yevgeny Seldin. Factored bandits. *Advances in Neural Information Processing Systems*, 31:2835–2844, 2018.

[94] Masrour Zoghi, Zohar S Karnin, Shimon Whiteson, and Maarten De Rijke. Copeland dueling bandits. *Advances in Neural Information Processing Systems*, 28:307–315, 2015.

[95] Masrour Zoghi, Shimon Whiteson, and Maarten de Rijke. MergeRUCB: A method for large-scale online ranker evaluation. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 17–26, 2015.

[96] Masrour Zoghi, Shimon Whiteson, Remi Munos, and Maarten Rijke. Relative upper confidence bound for the k-armed dueling bandit problem. In *International Conference on Machine Learning*, pages 10–18. PMLR, 2014.

[97] Masrour Zoghi, Shimon A Whiteson, Maarten De Rijke, and Remi Munos. Relative confidence sampling for efficient on-line ranker evaluation. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, pages 73–82, 2014.