NORTHEASTERN UNIVERSITY
GRADUATE SCHOOL OF COMPUTER SCIENCE
Ph.D. THESIS APPROVAL FORM

THESIS TITLE: Relevance Assessment (Un-)Reliability in Information Retrieval: Minimizing Negative Impact

AUTHOR: Pavel Metrikov

Ph.D. Thesis Approved to complete all degree requirements for the Ph.D. Degree in Computer Science.

Thesis Advisor Date: 1/8/16

Thesis Reader Date: 01/04/2016

Thesis Reader Date: 01/04/2016

Thesis Reader Date: 01/04/2016

GRADUATE SCHOOL APPROVAL:

Director, Graduate School Date: 1/11/16

COPY RECEIVED IN GRADUATE SCHOOL OFFICE:

Recipient's Signature Date: 1/11/2016

Distribution: Once completed, this form should be scanned and attached to the front of the electronic dissertation document (page 1). An electronic version of the document can then be uploaded to the Northeastern University-UMI website.
Relevance Assessment (Un-)Reliability in Information Retrieval: Minimizing Negative Impact

A dissertation presented
by

Pavel Metrikov

to the Faculty of the Graduate School
of the College of Computer and Information Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Northeastern University
Boston, Massachusetts
January, 2016
Dedication

To my advisors, present and past, Jay Aslam and Valeri Gitis (IITP RAS), and to my senior colleagues, Evangelos Kanoulas, Igor Kuralenok, and Virgil Pavlu—for leading me to become a scientist.

To my dear friends at Northeastern: Keshi Dai, Dimitris Kanoulas, Cheng Li, Alper Ockan, Bhumi Parmar, Rauno Peets, Bahar Qarabaqi, Shahzad Rajput, and Stefan Savev—for all the fun we have had together.

To my father, Andrey, to my mother, Valentina, and to my sister, Elena—for their love and faith in me.
Abstract

Collecting relevance assessments is a very important procedure in Information Retrieval. It is conducted to (1) evaluate the performance of an existing search engine, or (2) build and train a new one. While most of the popular performance evaluation measures and search engine training algorithms assume the relevance assessments are accurate and reliable, in practice this assumption is often violated. Whether intentionally or not, assessors may provide noisy and inconsistent relevance judgments potentially leading to (1) wrong conclusions about performance of a search engine, or (2) inefficient or suboptimal training of a search engine.

Addressing the problem above, we first (a) demonstrate how one can quantify the negative effect of assessor disagreement (including intra-assessor disagreement as a special case) on the ranking performance of a search engine. Beside this theoretical result, we also propose practical recipes for (b) tuning existing evaluation measures with the goal of making them more robust to the label noise, (c) improving the reliability of relevance estimates by collecting and aggregating multiple assessments (potentially through crowdsourcing), and (d) incorporating noise reduction component into learning-to-rank algorithms.
## Contents

Abstract i

List of Tables v

List of Figures vii

1 Introduction 1

1.1 Toolkit 3

1.1.1 nDCG Evaluation Measure 3

1.1.2 Kendall rank correlation coefficient 4

1.1.3 Learning-to-Rank 5

2 Impact of Assessor Disagreement on Ranking Performance 9

2.1 Assessor Disagreement Model 9

2.2 Obtaining nDCG Upper Bound via Simulation 11

2.2.1 Estimating nDCG upper bound with a Confusion Matrix alone 11

2.2.2 Estimating nDCG upper bound for large-scale Learning-to-Rank collections 13

2.3 Closed Form Approximation of nDCG Upper Bound 14

3 Minimizing Effect of Label Inconsistency by Tuning Evaluation Measures 17

3.1 Label Inconsistency and IR Evaluation Measures 17

3.2 Measuring Agreement on Order of Retrieval Systems 19

3.3 Experiments 20

3.3.1 Inferring Optimal Gains 21

3.3.2 Optimal gains used with nDCG objective of Learning-to-Rank 22

3.3.3 Simulating Label Noise 22

4 Better Ordinal Label Aggregation: a Latent Trait Model 25

4.1 Related work 27
4.2 Experiment setup, data, crowd assessments ................................ 29
4.3 Assessor and document models ................................................. 30
4.4 Aggregation of crowdsourced assessments ................................. 32
  4.4.1 Likelihood on documents: “GDA” method ............................... 32
  4.4.2 Likelihood on document pairs: “GDA-PW” ............................... 33
  4.4.3 Baseline comparison: EM algorithm ...................................... 34
  4.4.4 Baseline comparison: Polytomous Rasch ................................. 35
  4.4.5 Label Aggregation Results ................................................... 36
  4.4.6 Aggregation quality as function of number of assessments ...... 37
  4.4.7 Analysis: Assessor Informativeness ...................................... 39
  4.4.8 Analysis: Document Difficulty ............................................. 42

5 Learning-to-Rank in the Presence of Label Noise .......................... 45
  5.1 EM + LambdaMART ............................................................. 46
    5.1.1 Experiment Setup, EM Processing .................................... 47
    5.1.2 Training on Probabilistic Pairs ...................................... 49
    5.1.3 Experiments ............................................................... 51
  5.2 Learning-to-Rank with GDA-PW-MART ................................... 51
    5.2.1 Learning-to-rank by extending GDA-PW ............................ 52
    5.2.2 Results ................................................................. 54

6 Tighter Integration of Label Aggregation with Learning-to-Rank ........ 57
  6.1 Related work: learning in the presence of label noise ................. 60
  6.2 Experiment setup, data, assessments ..................................... 61
  6.3 Aggregation of Human & Virtual Assessments: GDA-PW-VA .......... 63
    6.3.1 Label Aggregation Results ............................................ 64
  6.4 Learning with Human & Virtual Assessments: GDA-PW-VA-MART ..... 66
    6.4.1 Learning-to-Rank Results ............................................ 66
  6.5 Analysis of Virtual Assessor ................................................ 67
    6.5.1 Aggregation quality as function of number of assessments ..... 67
    6.5.2 Analysis: Virtual Assessor Informativeness ....................... 69

7 Conclusion .................................................................................. 71

Bibliography ................................................................................. 73
# List of Tables

1.1 Standard Exponential Gains ........................................ 4  
2.1 Probability of confusing assessor label with true label on Yandex dataset .................................................. 13  
2.2 nDCG upper bounds derived from disagreement models C applied to popular learning-to-rank data sets ......................... 15  
3.1 Optimal nDCG gains (shared across all the queries) derived for three groups of assessors. ........................................ 21  
3.2 Optimal nDCG gains derived for three groups of assessors for each query individually ............................................ 22  
3.3 Optimal nDCG gains for "Bronze" assessors (mixing up labels "1" and "2") .............................................................. 24  
3.4 Optimal nDCG gains for "Bronze" assessors (mixing up labels "0" and "1") .............................................................. 24  
4.1 Datasets for label aggregation and learning-to-rank .................. 29  
4.2 Performance of various methods for aggregation of crowdsourced assessments, evaluated on 4 different datasets .................... 37  
4.3 Performance of various methods for aggregation of crowdsourced assessments + NIST + random assessments .................... 38  
6.1 Datasets for label aggregation and learning-to-rank (with virtual assessor) .......................................................... 62  
6.2 Performance of methods for label aggregation (including virtual assessor) .......................................................... 65
## List of Figures

1.1 Search Engine life cycle ........................................ 1  
1.2 Classification and Regression for Learning-to-Rank ................ 6  
1.3 A regression tree ........................................... 6  
1.4 LambdaMART .................................................. 7  

2.1 Two assessors disagreeing on the relevance of documents in a ranked list .................................................. 10  
2.2 Real vs. Simulated nDCG bounds due to inter-assessor disagreement 12  
2.3 Performance breakdown on Yandex learning-to-rank collection ... 13  

3.1 Label noise transforming into system score noise .................... 18  
3.2 Criterion for optimality of nDCG gains. .......................... 19  
3.3 Agreement between pairs of groups of assessors on the order of systems as function of $g_1$. ......................... 21  
3.4 LambdaMART performance for the models trained with nDCG gains optimized for inter-assessor agreement ...................... 23  

4.1 Refining relevance assessments with EM .......................... 28  
4.2 Refining relevance assessments with GDA ........................ 31  
4.3 Assessor grade model (conceptual illustration for “GDA” method) . 30  
4.4 Aggregation quality as function of number of assessments ......... 38  
4.5 User model parameters inferred from crowdsourced judgments ... 40  
4.6 Distributions of crowdworkers and assessments by inferred informativeness ......................................................... 41  
4.7 Document difficulty: consistency and relation to relevance ...... 42  
4.8 Assessor informativeness vs. $K_{\tau}$ between the assessor and an expert 44  

5.1 Graded assessment interface with topic keywords and description. 47  
5.2 EM + LambdaMART ............................................ 47  
5.3 Convolution of relevance grades distribution for documents A and B 49
5.4 Performance of LambdaMART trained on probabilistic pairwise preferences from probability distributions returned by EM ............. 52
5.5 Performance of GDA-PW-MART compared to other ranking algorithms ................................................................. 53
5.6 GDA-PW-MART .................................................................................................................................................. 54

6.1 The virtual assessor concept for learning-to-rank .................. 58
6.2 GDA-PW-VA-MART ................................................................. 60
6.3 Performance of GDA-PW-VA-MART on TREC 2010 RF dataset .... 66
6.4 Performance of GDA-PW-VA-MART on Yandex dataset ........ 68
6.5 Assessor informativeness vs. independent qualification score .... 68
6.6 Aggregation quality as function of number of assessments (with virtual assessor) ....................................................... 68
6.7 Assessor informativeness (including virtual assessor) vs. $K_\tau$ between the assessor and an expert ...................... 69
Chapter 1

Introduction

An information retrieval system (IR system), or search engine, aims to return a ranked list of documents from a document collection, ordered by their relevance to the user’s request (see Figure 1.1).

Figure 1.1: Search Engine life cycle: a search engine reports a list of potentially relevant documents for a query issued by a user; relevance of the top few results is assessed in order to (1) evaluate the quality of the search engine, and (2) to train or improve the search engine via machine learning (learning-to-rank).
In order to evaluate how well an information retrieval system performs, many different measures have been proposed in the past. Most of them rely on the process called relevance assessment, when each retrieved document is assigned a relevance level, or relevance label, by an expert. The relevance labels can be either binary (relevant/non-relevant), or graded (when documents are partially relevant to a query). The collected assessments are then aggregated into a single summary value of the overall quality of the retrieved ranked list, the exact transformation procedure being formalized in the definition of a particular IR evaluation measure.

IR measures, along with underlying relevance assessments, are typically needed to (1) compare different search engines, and (2) train a search engine via learning-to-rank procedure (see Section 1.1.3). It is intuitively clear that the reliability of IR metrics, a very desirable property to effectively conduct aforementioned tasks, is highly dependent on the reliability of the underlying relevance assessments. The latter may become a major concern for various reasons. First of all, relevance is one of the subjective notions which are hard to grasp and difficult to define [40]. As a result, it is well known that humans (experts) do disagree [2], interpret query in different ways (users), have different standards (judges), are uncertain about their assessments (e.g., should a document be labeled "highly relevant" or just "relevant"?), or simply do not understand the query (students). Furthermore, as crowdsourcing is becoming a popular and much cheaper alternative to expert knowledge in general, there is no surprise it has recently attracted much attention from the IR community, and, in particular, has been used to collect relevance judgments [21, 48]. Since in crowdsourcing the tasks are broadcast to a large unknown group of workers (“Mechanical Turkers”) in the form of an open call for solutions, it is a matter of fact that a good fraction of workers turn out to be lazy or greedy or adversarial or just do not have necessary qualifications. In spite of all these reasons, most of existing learning-to-rank algorithms assume that assessors are specially trained and their judgments are accurate, therefore no special treatment of potential label unreliability is typically taken.

In this dissertation we present our approach to confront the problems mentioned above. In Chapter 2 we first study the effect of assessor disagreement on the maximum measured performance of a search engine. We argue that no matter how good a ranking algorithm we use or how informative the features we have, a 100% performance is never achievable when evaluated with standard IR measures based on human assessments. In Chapter 3, we propose a method for tuning Information Retrieval measures by minimizing the negative impact of relevance label inconsistency. The tuned nDCG metric turns out to be also a better learning-
to-rank objective. In Chapter 4, we target the problem of utilizing the noisy relevance assessments collected through crowdsourcing. There we propose a new label aggregation method that deals with noisy input by modeling and inferring each assessor’s skills and biases and accounting for a priori unknown document difficulty. In Chapter 5 we present a two-stage approach of training a ranker in the noisy environment, along with two alternative implementations: either based on (a) the variation of popular Expectation-Maximization (EM) or (b) our method GDA-PW introduced earlier in Chapter 4. The first stage can be seen as a filter that rectifies the noisy relevance assessments through label aggregation (via either EM or GDA-PW) and passes the clean output to learning-to-rank (either the modification of LambdaMART or extension of GDA-PW). Later, in Chapter 6, we propose to integrate these two components (label aggregation and learning-to-rank) more tightly, as we believe that having a feedback from learning-to-rank can help label aggregation perform better, and perform well even when only a single relevance assessment per document is available. A better label aggregation, in its turn, can help improving the performance of subsequent learning-to-rank. Overall, our major contribution is a unified framework with alternating procedures of label aggregation and learning-to-rank, in order to better account for the presence of non-homogenous noise in the training dataset and maximize the use of information about relevance available in both human assessments and machine-generated query-document features.

1.1 Toolkit

1.1.1 nDCG Evaluation Measure

Now we are going to describe a popular IR measure called Normalized Discounted Cumulative Gain (nDCG) [22], which we will be actively using throughout the Dissertation. While most of results we present here are not specific to the particular selection of the IR metric and should hold for many other related (and correlated) metrics (e.g., Area Under Curve, Mean Average Precision [44], Expected Reciprocal Rank [12]), it has some nice properties that we will exploit in Chapter 2 (easiness of mathematical manipulation leading to a closed-form solution) and Chapter 3 (it is parameterizable and we show how to choose a particular parametrization which is most robust to the relevance label noise). Moreover, despite its relative simplicity, the nDCG metric has a good discriminatory power (i.e., it is good in discriminating good retrieval systems from bad systems, see [47]) and ability to work with multi-grade relevance assessments, which adds a lot to its popularity. As most of the IR measures, nDCG favors relevant documents being close to the top of the ranked list.
and penalizes them for being far from the top. More specifically, the contribution of each document to the overall Discounted Cumulative Gain (DCG) of the ranked list is a product of gain, a measure of document’s utility, and discount, a penalty for a relevant document appearing lower in the search result. Commonly used gains are exponentially proportional to the relevance grade of a document (see Table 1.1), discounts being inversely proportional to logarithm of the position of the result. Thus, DCG accumulated at a particular rank position \( k \) is defined as:

\[
DCG@k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i + 1)},
\]

where \( rel_i \) is the graded relevance of the result at position \( i \).

To make it comparable across the queries of different lengths and proportions of relevant documents, a normalized version of DCG is used:

\[
nDCG@k = \frac{DCG@k}{IDCG@k},
\]

where \( IDCG@k \) is \( DCG@k \) of an ideally ranked list (i.e. of a result list sorted by document relevance). It is easy to see, that nDCG takes on values between 0.0 and 1.0, the latter corresponding to an ideally sorted result list.

1.1.2 Kendall rank correlation coefficient

Often we need to compare different rankings of the same objects, and quantify how much different they are. It could be useful, for instance, when comparing two search engines (retrieval systems) on how similarly they rank the documents by estimated relevance with respect to some query. Another application that we will be dealing with is the comparison of two different evaluation measures on how similarly they rank a set of retrieval systems by their measured performance.

One of the popular measures of similarity of the orderings (rankings) is Kendall
rank correlation coefficient, or Kendall $\tau$ [27]. If we have two rankings of $n$ objects, for each pair of objects we can tell whether this pair is ranked in the same order in both rankings (a concordant pair), or in the opposite order (a discordant pair). In a more general case of partial orderings, multiple objects can share the same position in a ranked list (i.e., no object can be preferred to any other of the same rank). In this case, a pair of objects is assumed neither concordant nor discordant if they appear in the same position in either ranked list.

The Kendall $\tau$ coefficient is defined as:

$$K_\tau = \frac{\text{(number of concordant pairs)} - \text{(number of discordant pairs)}}{\text{(total number of pairs)}}$$

Kendall $\tau$ takes on values between -1.0 and 1.0 (a perfect negative correlation and a perfect positive correlation, respectively). Value of 0.0 corresponds to the absence of any observable correlation.

1.1.3 Learning-to-Rank

Learning-to-rank [32] is a common Machine Learning technique for Information Retrieval that utilizes labeled documents over a set of training queries in order to obtain a model that can be used to rank documents for new testing queries. To make it work, each document-query datapoint has to be represented by a consistent set of features correlated with relevance (e.g., based on query term frequencies [45], web-graph properties [6], click-through counters [23], etc.). A learning-to-rank algorithm learns the optimal way of combining these features into a score, a numerical value that is assigned to each document from the collection, and is used by the search engine to rank all of them with the goal of maximizing the evaluation measure of interest.

In its simplest form, a learning-to-rank algorithm can be formulated as a regression or classification problem (see Figure 1.2): given a single query-document pair, predict its relevance label. This approach is called pointwise, and numerous standard off-the-shelf Machine Learning algorithms are readily available to solve the problem: Linear and Logistic Regressions, Artificial Neural Networks, AdaBoost, Support Vector Machine, and Gradient Boosting [16] among others.

It turns out that more efficient are pairwise learning-to-rank algorithms. They approximate the ranking problem by a binary classification problem: it boils down to training a binary classifier that can tell which document is better in a given pair

---

1 It is worth noting that there exist other related measures of rank correlation (e.g., Pearson correlation coefficient), and our findings should generalize to them.
Figure 1.2: Classification and regression as pointwise Learning-to-Rank methods. (Left) a classification problem: given a query-document pair predict its relevance label (i.e., separate relevant documents from non-relevant in the feature space); (right) a regression problem: given a query-document pair predict its graded relevance label.

of documents. The goal is to minimize the number of pairwise ordering mistakes in a ranking. Many pairwise learning-to-rank algorithms have been proposed recently, including efficient adaptations of aforementioned pointwise ones: RankNet [7], RankBoost [15], and SVMRank [24].

Finally, the most recent and effective are the ones belonging to the family of listwise learning-to-rank algorithms. They try to directly optimize for one of the IR evaluation measures, which is not a trivial problem as most evaluation measures are not continuous functions with respect to ranking model’s parameters. Here we consider a state-of-the-art listwise learning-to-rank algorithm called LambdaMART [7]. In the recent Yahoo! learning-to-rank challenge [11], the winning team used an ensemble of LambdaMART models.

LambdaMART uses gradient boosting [16] to produce an ensemble of

Figure 1.3: An example of a regression tree: starting with the root node, the decision is made at each split on whether to proceed to the left or the right child, based on some feature’s value; once a leaf is reached, a real-valued number associated with that leaf is returned.
Figure 1.4: LambdaMART: (left) a ranker is built as an ensemble of regression trees, $F_m(x) = F_{m-1}(x) + h_m(x)$; (right) an IR metric (e.g., nDCG@10) on the training set is being improved as more trees (m) are added to the ensemble.

weak models (regression trees, see Figure 1.3) that together form a strong model. The ensemble $F(x)$, mapping an input feature vector $x \in \mathbb{R}^n$ to a real-valued score $s$ (such that for each document $d$: $s_d = F(x_d)$), is built in a greedy stage-wise manner by performing gradient descent in functional space (using the first and second derivatives of $C$ with respect to the score of each document $d$: $\frac{\partial C}{\partial s_d}$ and $\frac{\partial^2 C}{\partial s_d^2}$, in order to compute a Newton step). At each iteration $m$, the ensemble $F_{m-1}(x)$ from previous iteration is augmented with a new regression tree $h_m(x)$:

$$F_m(x) = F_{m-1}(x) + h_m(x),$$

where $h_m(x)$ is picked greedily so as to minimize a loss function, or cost function, $C(F_m(x))$ which assures better ranking performance on the training set, in terms of some particular IR metric (see Figure 1.4). We will cover LambdaMART in more details in Section 5.1.2.

Further in the Thesis we will be dealing mainly with LambdaMART as a baseline ranking algorithm.
Chapter 2

Impact of Assessor Disagreement on Ranking Performance

We consider the impact of the assessor disagreement on the maximum measured performance that a ranker can hope to achieve. We demonstrate that even if a ranker were to achieve perfect performance with respect to a given assessor, when evaluated with respect to a different assessor, the measured performance of the ranker decreases significantly. This decrease in performance may largely account for observed limits on the performance of learning-to-rank algorithms. Further in this chapter we will estimate the upper bound on nDCG performance of any ranking algorithm as a function of assessor disagreement summarized in terms of a confusion matrix.

2.1 Assessor Disagreement Model

In both Machine Learning and Information Retrieval, it is well known that limitations in the performance of ranking algorithms can result from several sources, such as insufficient training data, inherent limitations of the learning/ranking algorithm, poor instance features, and label errors. In this chapter we focus on performance limitations due solely to label “errors” which arise due to inter-assessor disagreement.

Consider a training assessor A that provides labels for training data and a testing assessor B that provides labels for testing data. Even if a ranker can produce a perfect list as judged by A, its performance will be suboptimal with respect to B, given inevitable inter-assessor disagreement (see Figure 2.1). In effect, no ranking algorithm can simultaneously satisfy two or more disagreeing assessors (or users).

\[1\] This chapter is based on a previously published work [36].
Thus, there are inherent limitations in the performance of ranking algorithms, independent of the quality of the learning/ranking algorithm, the availability of sufficient training data, the quality of extracted instance features, and so on.

We model inter-assessor disagreement with a confusion matrix $C$, where $c_{ij}$ corresponds to the (conditional) probability that a document labeled $i$ by testing assessor $B$ will be labeled $j$ by training assessor $A$, for some given set of label grades such as \{0, 1, 2, 3, 4\}.

Given such a model of inter-assessor disagreement, we ask the question, “What is the expected performance of a ranked list optimized for training assessor $A$ but evaluated with respect to testing assessor $B$?” We approach this question in two ways, via simulation and closed form approximation. In the former case, we use the confusion matrix $C$ to probabilistically generate training labels $A$ from testing labels $B$, optimally rank documents according to $A$, and evaluate with respect to $B$. In the latter case, we analytically derive a closed-form approximation to this limiting performance, as measured by nDCG. The former case is a more general approach as it works for any other IR measure as well, while the latter case is nDCG-specific as we exploit the simplicity of mathematical formulation of nDCG to derive a closed form (approximate) solution.

Often, to train and evaluate a search engine, a very large set (~millions) of relevance assessments is collected which comes from a pool of many assessors (~thousands), each assessor judging a subset of all query-document pairs. Under this scenario, one can still approximate the disagreement between multiple assessors with a single confusion matrix $C$, such that each entry $c_{ij}$ corresponds to the (conditional) probability that a document labeled $i$ by some testing assessor picked at random will be labeled $j$ by some training assessor also picked at random. In a particular example described later (see Section 2.2.2) the training assessors could
be regular ones, while the testing assessors could have a higher qualification (so called meta-assessors or super-assessors).

Given a confusion matrix $C$ modeling inter-assessor disagreement, one can apply our results to any Learning-to-Rank dataset. The limiting nDCG values obtained correspond to reasonable upper bounds on the nDCG performance of any Learning-to-Rank algorithm, even one given unlimited training data and perfect features (see Section 2.2.1). Considering the performance of existing algorithms on these datasets, and comparing with the upper bounds we derive, one can argue that Learning-to-Rank is approaching reasonable limits on achievable performance (see Section 2.2.2).

2.2 Obtaining nDCG Upper Bound via Simulation

Much research in the IR community has focused on addressing the problem of system evaluation in the context of missing, incomplete, or incorrect document judgments [2]. Soboroff and Carterette [10] provide an in-depth analysis of the effect of assessor disagreement on the Million Query Track evaluation techniques. Both assessors and users often disagree on the degree of document relevance to a given query, and we model such disagreement with a confusion matrix $C$ as described above. On data sets with multiple assessments per query-document pair, such as the TREC$^2$ Enterprise Track [3], these confusion matrices can be directly estimated from data, and they can be obtained from user studies as well [9, 20].

For any ranked list returned by a system, the expected limiting nDCG due to assessor disagreement can be formulated as a function of (1) the disagreement model in the form of a confusion matrix $C$ and (2) the number of assessed documents and their distribution over the label classes. One way to compute this expected nDCG is numerical simulation: For every document $d$ having testing label $i_d$ in the ranked list, we randomly draw an alternative label $j_d$ with the probability $c_{ij}$; we then sort the ranked list in decreasing order of $\{j_d\}$ and evaluate its nDCG performance with respect to labels $\{i_d\}$. This simulation is repeated multiple times and the results averaged to obtain an accurate estimate of the expected limiting nDCG.

2.2.1 Estimating nDCG upper bound with a Confusion Matrix alone

In our first experiment, we test whether the inter-assessor confusion matrix $C$ alone can be used to estimate the limiting nDCG value. We do so by considering data sets

---

2The Text REtrieval Conference (TREC, http://trec.nist.gov) is an annual information retrieval conference and competition, the purpose of which is to support and encourage research within the information retrieval community. The TREC Conference series is co-sponsored by the National Institute of Standards and Technology (NIST).
that have multiple judgments per query-document pair, such as were collected in the TREC 2007 Enterprise Track [2] where each topic was judged by three assessors: a "Gold-standard" assessor G (expert on task and topic), a "Silver-standard" assessor S (expert at task but not at topic), and a "Bronze-standard" assessor B (expert at neither). Documents were classified by each judge as "highly likely" (label "2") to be included in list of relevant documents; "possibly" included or useful (label "1"); or "not" useful (label "0"). Each G, S, and B set of assessments can take on the role of training or testing assessor, as described above, giving rise to six possible combinations: GS, GB, SG, SB, BG, BS. For each such combination, such as GS, the optimal ranked list can be computed with respect to G and evaluated with respect to S, resulting in a real suboptimal nDCG. The GS confusion matrix can also be computed from the data given and the simulation described above performed, yielding an estimated limiting nDCG. These actual and estimated limiting nDCG values can then be compared.

Using the TREC Enterprise data, Figure 2.2 compares the estimated limiting nDCG obtained through simulation with a confusion matrix (x-axis) with the real suboptimal nDCG (y-axis) obtained from different assessors. The left plot uses a confusion matrix $C_{ENT}$ obtained from the TREC Enterprise data itself, as described above, while the right plot uses a confusion matrix $C_{MSR}$ obtained from a user study conducted by Microsoft Research [9]. Note that the more accurate confusion matrix yields better simulated results, as expected, and that the confusion matrix

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure2.png}
\caption{Applying C_ENT model (left) and C_MSR model (right) to TREC enterprise data. X-axis is the simulated nDCG upper bound, while Y-axis is the actual nDCG assessor disagreement measured between 2 TREC assessors; pairs of assessor type ("Gold-Silver" as GS) are indicated by colors.}
\end{figure}
Table 2.1: Probability of confusing assessor label (columns) with true label (rows) on Yandex dataset (Russian search). Labels \{Bad, Poor, Good, Excellent, Perfect\} correspond to numerical values \{0, 1, 2, 3, 4\}.

<table>
<thead>
<tr>
<th></th>
<th>Bad</th>
<th>Poor</th>
<th>Good</th>
<th>Exc.</th>
<th>Perf.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bad</td>
<td>0.75</td>
<td>0.34</td>
<td>0.07</td>
<td>0.04</td>
<td>0.03</td>
</tr>
<tr>
<td>Poor</td>
<td>0.22</td>
<td>0.54</td>
<td>0.13</td>
<td>0.04</td>
<td>0.02</td>
</tr>
<tr>
<td>Good</td>
<td>0.02</td>
<td>0.11</td>
<td>0.73</td>
<td>0.52</td>
<td>0.05</td>
</tr>
<tr>
<td>Excellent</td>
<td>0.00</td>
<td>0.01</td>
<td>0.06</td>
<td>0.32</td>
<td>0.08</td>
</tr>
<tr>
<td>Perfect</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.08</td>
<td>0.83</td>
</tr>
</tbody>
</table>

alone can be used to accurately estimate limiting nDCG values in most cases.

### 2.2.2 Estimating nDCG upper bound for large-scale Learning-to-Rank collections

Given that a confusion matrix alone can be used, via simulation, to estimate limiting nDCG performance, we next consider other data sets and their associated real or estimated confusion matrices. Yandex\(^3\) [20] conducted user studies to obtain confusion matrices specific to Russian search \(C_{YanR}\), see Table 2.1) and to Ukrainian search \(C_{YanU}\), and these were shown to improve Learning-to-Rank performance if the learner was given such models as input.

Figure 2.3 presents the estimated limiting nDCG value when applying \(C_{YanR}\) to the corresponding learning-to-rank data set of Yandex.\(^4\) For comparison, the figure also presents the actual performance of LambdaMART learning algorithm on this data set, as well as the performance of the random ranking. Consider the difference between the estimated limiting nDCG bound and the rank-

---

\(^3\)Yandex is the largest search engine in Russia with about 60% market share in that country. Yandex ranked as the 4th largest search engine worldwide, with more than 150 million searches per day as of April 2012, and more than 50.5 million visitors (all company’s services) daily as of February 2013 (see https://en.wikipedia.org/wiki/Yandex).

\(^4\)This experiment was performed during an internship at Yandex.
ing performance: even though the assessor disagreement impairs the ranking performance substantially, the learning algorithms still have some room for improvement, by means of designing better ranking functions along with discovering better query-document feature representation.

2.3 Closed Form Approximation of nDCG Upper Bound

Let $\mathcal{L} = \{0, 1, 2, 3, 4\}$ be the set of relevance grades, $n_k$ the number of documents with reference label $k \in \mathcal{L}$, $n$ the total number of documents in the ranked list, and $P_{rank}(i, r)$ the probability that the rank of a given document with reference (testing) label $i$ is $r$, as ordered by the alternative (training) labels $j$. One can then show that the expected nDCG as measured by reference labels is

$$E[nDCG] = \frac{1}{IdealDCG} \sum_{i \in \mathcal{L}} \left[ n_i \cdot gain(i) \cdot \sum_{r=1}^{n} \frac{P_{rank}(i, r)}{discount(r)} \right]$$

where

$$P_{rank}(i, r) = \sum_{j \in \mathcal{L}} \left[ c_{ij} \cdot \left( \frac{\sum_{h=0}^{r-1} \sum_{s=r-1-h}^{n-1} \Psi_{ij}(h, s)}{s+1} \right) \right]$$

and $\Psi_{ij}(h, s)$ is the probability that other $s$ documents have the same alternative label $j$, and other $h$ documents have alternative label higher than $j$, given that a particular document with reference label $i$ has alternative label $j$. Computing $\Psi_{ij}(h, s)$ straightforwardly is inefficient for even moderately long ranked lists, with a running time of $O(n^2|\mathcal{L}|)$.

We instead employ a closed form approximation (CFA) based on approximating $\Psi$, a sum-product of binomial conditional distributions, with a Gaussian joint distribution of two variables $(h + s, s)$. This approximation becomes more accurate as ranked lists get longer. For a fixed $i$ and $j$ we have

$$\binom{h + s}{s} \sim N_{ij}^r(\mu, \Sigma),$$

$$\mu = \begin{pmatrix} \mu_{h+s} \\ \mu_s \end{pmatrix}, \Sigma = \begin{pmatrix} \sigma_{h+s}^2 & \text{cov}_{h+s,s} \\ \text{cov}_{s,h+s} & \sigma_s^2 \end{pmatrix}$$
<table>
<thead>
<tr>
<th>Collection</th>
<th>C_MSR</th>
<th>C_YanR</th>
<th>C_YanU</th>
<th>Learning-to-Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSLR30K (SIM)</td>
<td>0.780</td>
<td>0.867</td>
<td>0.900</td>
<td>0.741</td>
</tr>
<tr>
<td>MSLR30K (CFA)</td>
<td>0.794</td>
<td>0.869</td>
<td>0.898</td>
<td></td>
</tr>
<tr>
<td>Yahoo (SIM)</td>
<td>0.861</td>
<td>0.920</td>
<td>0.944</td>
<td>0.801</td>
</tr>
<tr>
<td>Yahoo (CFA)</td>
<td>0.887</td>
<td>0.919</td>
<td>0.938</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.2: nDCG upper bounds derived from disagreement models $C$ applied to popular learning-to-rank data sets. SIM rows are simulated values; the CFA rows are closed form approximation. Last column is best known learning-to-rank performance.

where

\[
\begin{align*}
a_{ij} &= \sum_{k \geq j} c_{ik} \\
\mu_{h+s} &= -a_{ij} + \sum_{k \in \mathcal{L}} n_k \cdot a_{kj} \\
\mu_s &= -c_{ij} + \sum_{k \in \mathcal{L}} n_k \cdot c_{kj} \\
\sigma^2_{h+s} &= -a_{ij} \cdot (1 - a_{ij}) + \sum_{k \in \mathcal{L}} n_k \cdot a_{kj} \cdot (1 - a_{kj}) \\
\sigma^2_s &= -c_{ij} \cdot (1 - c_{ij}) + \sum_{k \in \mathcal{L}} n_k \cdot c_{kj} \cdot (1 - c_{kj}) \\
\text{cov}_{s,h+s} &= -c_{ij} \cdot (1 - a_{ij}) + \sum_{k \in \mathcal{L}} n_k \cdot c_{kj} \cdot (1 - a_{kj}).
\end{align*}
\]

We can approximate $E[nDCG]$ in $O(n^2)$ time given that the “spread” of the Gaussian grows as $O(\sqrt{n})$ per component.

Table 2.2 presents the estimated limiting nDCG values when applying three different confusion matrices to two Learning-to-Rank data sets. The CFA rows of table show closed form approximations for comparison with simulated nDCG upper bounds (SIM rows). For comparison, the last column in the table presents the actual best known performance of a learning algorithm on these data sets.
Chapter 3

Minimizing Effect of Label Inconsistency by Tuning Evaluation Measures

Now we focus on nDCG choice of gains, and in particular on the fracture between large differences in exponential gains of high relevance labels and the not-so-small confusion, or inconsistency, between these labels in data\(^1\). We show that better gains can be derived from data by measuring the label inconsistency, to the point that virtually indistinguishable labels correspond to equal gains. Our derived optimal gains make a better nDCG objective for training learning-to-rank algorithms: a ranker tends to perform better if it was trained with nDCG gains optimal in terms of label inconsistency.

3.1 Label Inconsistency and IR Evaluation Measures

The popular IR measures nDCG (see Section 1.1.1) and ERR [12] translate relevance labels \(\{0:\text{not relevant}, 1:\text{related}, 2:\text{relevant}, \text{etc}\}\) into “gains” that better reflect the utility of each particular document with respect to the given query, (irrespective to the retrieval rank/position, typically modeled by a separate discount function). Most commonly used are exponential gains \(\{g_0 = 0, g_1 = 1, g_2 = 3, ..., g_i = 2^i - 1\}\) for both nDCG and ERR. Besides its obvious heuristic nature “three related docs are as good as a relevant one” (as \(3g_1 = g_2\)), the exponential gain function is too sensitive to the inevitable inconsistency in relevance labels due to human assessments.

Further, it feels intuitively correct that the harder it is for the assessor to decide between labels \((i, i + 1)\) on the average document, the closer the associated

\(^1\)This chapter is based on a previously published work [37].
gains \((g_i, g_{i+1})\) should be. That is, evaluation can be very unreliable when the gains 
\(g_3 = 7\) and \(g_4 = 15\) used in nDCG formulae are radically different, while assessors 
are not making a clear delimitation between labels \(i = 3\) and \(i + 1 = 4\): the 
resulting high variability of the nDCG scores is due to the inconsistency in assessing 
the relevance, and not to the actual retrieval quality as perceived by the assessor 
(see Figure 3.1). Conversely, if the data exhibits clear distinction between labels, 
then the associated gains should be significantly different. In this chapter, we are 
studying nDCG gains derived from label inconsistency.

Unfortunately, very often we lack explicit measurements of uncertainty associated 
with assessor labels, for a good reason: such measurement is extremely difficult. However, we may try to approximate it by asking several assessors to judge 
the same set of query-document pairs. Consider a pair of relevance grades \((i, i+1)\) 
and the pair of documents \((d_1, d_2)\) labeled \((i, i + 1)\) by assessor \(A\). If the other as-

Figure 3.1: Label noise transforming into system score noise: an assessor judges the relevance of a set of documents returned by a retrieval systems; the noisy document relevance assessments are transformed into the noisy performance score via nDCG formula parameterized by sets of gains

assessors agree on most such pairs \((d_1, d_2)\) that \(d_1\) is less preferable than \(d_2\), we can conclude that \(A\) is able to distinguish grades \(i\) and \(i + 1\) well. As we rely on assessor consensus, we note that the observed label inconsistency between different assessors is a conflation of many effects mentioned in the Introduction, and generally it cannot be decomposed into specific factors such as disagreement vs uncertainty vs laziness using the information typically provided by Collection Creators. For such analysis, a deep logging of assessment process, with user studies detailing behavior beyond grades, would be necessary.

Now we formulate our criterion for optimality of gain values (see Figure 3.2). Suppose we have a set of query-document pairs labeled by several assessors, along
Figure 3.2: Criterion for optimality of nDCG gains: two assessors evaluate the performance of a set of systems; the noisy performance scores are produced by transforming the noisy document relevance assessments via nDCG formula parameterized by sets of gains (per assessor); the optimal nDCG gains minimize the disagreement on the order of systems ranked by the noisy performance scores.

with a set of systems that can be evaluated with these labels in terms of nDCG scores. By varying nDCG gains per label per query per assessor, we observe the change in system ranking as the fraction of system pairs of which judges agree upon ordering (Section 3.2). We select the optimal sets of gains to reduce the amount of order disagreement, in as much as it is due to label inconsistency. Finally, we provide empirical evidence of the superiority of the derived optimal gains by using them within the nDCG training objective for learning-to-rank algorithms (Section 3.3).

Related work comprises studies of learning nDCG or ERR parameters from other perspectives: from user preferences over rankings [60], by optimizing the efficiency or stability of nDCG [26], by correlation with online click metrics [33]. Additionally, the problem could potentially be addressed by maximizing the discriminatory power of nDCG metric as a function of gains [47].

3.2 Measuring Agreement on Order of Retrieval Systems

To measure the inter-assessor agreement on the order of systems we employ the Kendall rank correlation coefficient (Kendall $\tau$, see Section 1.1.2) as it is simple and
relies on fewer assumptions compared to other potential alternative measures including: (a) Pearson correlation coefficient, or (b) fraction of retrieval systems on order of which the assessors significantly agree/disagree across a set of queries. As the former approach (a) deals with numerical values by which elements are ordered in the lists, it could be sensitive to varying discriminatory power of the evaluation measures within different ranges of values. The latter approach (b) assumes an universal ordering of systems by their performance irrespective of any query, which is however not always the case in reality: some retrieval systems may work better for one set of queries and worse for another.

Thus, to measure the agreement between assessors \( A \) and \( B \) on some particular query \( q \), for both assessors \( A \) and \( B \) we order the set of available retrieval systems by their nDCG scores evaluated using labels from corresponding assessor, and then measure the correlation of these two orderings in terms of Kendall \( \tau \). If we change the label-to-gain transformations for either or both assessors, the resulting change in nDCG scores will trigger the change in system rankings which will be captured by change of measured agreement between the two assessors, i.e. Kendall \( \tau \) (see Figure 3.2). Eventually, the goal is to find the gains \( gi \) (per assessor per query) that maximize this agreement. In practice, for multiple values of \( i \), we propose to explore \( gi \) one-by-one in several rounds, until we converge to the (locally) optimal set of \( gi \).

If for each individual query, the number of documents judged by multiple assessors is not that high, it may lead to unreliable estimates of the gains \( gi \) per assessor per query. In this case one may want to reduce the number of free parameters by considering sets of the gains \( gi \) per assessor but \textit{shared} across all available queries.

### 3.3 Experiments

In our experiments we evaluate 59 systems submitted by participants of TREC Enterprise 2007 track on the basis of judgments provided by 3 groups of judges ("Gold-standard", "Silver-standard" and "Bronze-standard") for 50 topics (refer to Section 2.2 for details). In addition to these 59 submitted systems, we add 40 "random" systems that rank in a random order any given set of documents. These random systems improve the balance between good and bad systems, and help us in identifying IR measures that better separate good systems from bad. For all judged query-document pairs we have extracted 96 typical features to be later used in learning-to-rank.
Figure 3.3: Agreement between pairs of groups of assessors on the order of systems as function of $g_1$. Points maximizing this pairwise agreement are marked with "+". The further away from optimal points, the less agreement is observed. Gold and Silver assessors achieve more agreement with each other than with Bronze assessors.

![Kendall τ](image)

Table 3.1: Optimal nDCG gains (shared across all the queries) derived for three groups of assessors. The optimal $g_1$ is higher than standard exponential $g_1$, for any group of assessors.

<table>
<thead>
<tr>
<th></th>
<th>Gold</th>
<th>Silver</th>
<th>Bronze</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_0$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$g_1$</td>
<td>1.4</td>
<td>1.8</td>
<td>1.3</td>
</tr>
<tr>
<td>$g_2$</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>

3.3.1 Inferring Optimal Gains

Following the methodology in Section 3.2, we investigate how nDCG gains affect assessor disagreement on order of system pairs. First, the case when assessors within each group share the sets of gains across all the queries. By fixing $g_0 = 0.0$ and $g_2 = 3.0$ to be standard exponential gains of the grades 0 and 2, we are sweeping gains $g_1$ between 0.0 and 4.0 individually for each group of assessors.

We measure the agreement on the order of systems for assessors from each pair of groups (see Figure 3.3), as well as the average agreement. The outcome of the procedure maximizing average agreement is shown in Table 3.1, suggesting that the optimal $g_1$, for any group of assessors, is higher (and sufficiently higher for Silver assessors) than the typically used standard exponential gain of 1.0. It is also instructive to note that "Gold" and "Silver" judges agree with each other more than they do with their "Bronze" counterparts, which is expected given their levels of expertise.

Finally, we compute optimal sets of gains for each group of assessors for each query, by fixing $g_0$ and $g_2$ to be standard exponential gains and constraining $g_1$ to be between $g_0$ and $g_2$. The results for three queries are shown in Table 3.2. Notice
that the optimal gain $g_1$ for the middle grade of 1 (subjectively defined “potentially relevant”) varies substantially from query to query and from assessor to assessor: it is for certain queries more like “non relevant”, for other queries more like “relevant”, or somewhere in between. Therefore it could be useful to infer the optimal sets of gains per assessor per query, and we confirm this fact empirically in the next section.

### 3.3.2 Optimal gains used with nDCG objective of Learning-to-Rank

Once we have computed optimal gains, we demonstrate their superiority from the learning-to-rank perspective, where the state of the art algorithms like LambdaMART [57] directly optimize for an IR measure like nDCG. We train three LambdaMART models using "Bronze" labels, by optimizing for nDCG: (1) with standard exponential gains, (2) with optimal gains from Table 3.1, and (3) with individual sets of gains per query per assessor (Table 3.2). The "Bronze" labels are chosen as we have about 10 times more "Bronze" judgments than "Gold" or "Silver". When a ranker is trained, the algorithm generally focuses more on document pairs for which the difference of gains is big. Thus, when using optimal gains, we minimize the difference of gains for indistinguishable relevance grades, which makes the algorithm focus on most reliable pairs of documents (i.e. distinguishable); eventually it amounts to building a ranking model of higher quality.

We evaluate the models on testing set (in terms of nDCG with standard exponential gains) and perform 11-fold cross-validation to reduce the variability. The results (Figure 3.4) show that the model trained with optimal shared gains performs slightly better than the one trained with traditional exponential gains, while the model trained with optimal sets of gains per query per assessor performing the best. We conclude that deriving optimal gains that minimize inter-assessor inconsistency is beneficial when used with learning-to-rank algorithms.

### 3.3.3 Simulating Label Noise

Now we would like to empirically support our intuition (see Sec. 3.1) that the more an assessor confuses the adjacent relevance grades $(i, i + 1)$, the closer the corre-

<table>
<thead>
<tr>
<th></th>
<th>G</th>
<th>S</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_0$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$g_1$</td>
<td>2.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>$g_2$</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>G</th>
<th>S</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_0$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$g_1$</td>
<td>2.5</td>
<td>2.9</td>
<td>2.4</td>
</tr>
<tr>
<td>$g_2$</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>G</th>
<th>S</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_0$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$g_1$</td>
<td>1.0</td>
<td>1.0</td>
<td>0.1</td>
</tr>
<tr>
<td>$g_2$</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>

**Table 3.2:** Optimal nDCG gains derived for three groups of assessors for each query individually. Three queries displayed, left to right: "q=24", "q=29" and "q=31".
Figure 3.4: LambdaMART performance on the testing set. Compared are the models trained with "Bronze" labels optimized for nDCG with (1) exponential gains, with (2) optimal gains shared across all the queries, and (3) optimal gains per query. The results are evaluated by nDCG with standard exponential gains.

sponding gains \((g_i, g_{i+1})\) should be. To this end, we perform a controlled experiment where we inject a certain amount of noise in the existing "Bronze-standard" labels of the same TREC Enterprise 2007 collection that we worked with in previous sections. We simulate two scenarios: (1) labels "1" and "2" are made less distinguishable by flipping some fraction of them (i.e. let’s say p% of all labels "1" chosen at random become "2" and similarly p% of all labels "2" become "1"), and (2) labels "0" and "1" are made less distinguishable. As our intuition suggests, the more the level \(p\) of injected noise, the closer the optimal gain for the middle grade "1", \(g_1\), should approach \(g_2\) in the former case (1), or \(g_0\) in the latter case (2). To explore the effect of the level of label noise, we change it gradually from 0% (no additional artificial noise; only natural confusion exhibited in the original data) to 50% (the two grades are made fully indistinguishable). We re-run experiments from Sec. 3.3.1 on the noise-corrupted datasets. Results (see Tables 3.3, 3.4) go in line with our intuition.
### Table 3.3

Optimal nDCG gains derived for "Bronze" assessors (shared across all the queries) under varying level of label noise by mixing up labels "1" and "2". The more the label noise, the less the gap between optimal $g_1$ and $g_2$ as these two grades become less distinguishable.

<table>
<thead>
<tr>
<th>Injected Label Noise Level</th>
<th>0%</th>
<th>5%</th>
<th>10%</th>
<th>15%</th>
<th>20%</th>
<th>25%</th>
<th>30%</th>
<th>35%</th>
<th>40%</th>
<th>45%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_0$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$g_1$</td>
<td>1.3</td>
<td>1.5</td>
<td>1.9</td>
<td>2.0</td>
<td>2.1</td>
<td>2.4</td>
<td>2.5</td>
<td>2.7</td>
<td>2.9</td>
<td>2.9</td>
<td>3.0</td>
</tr>
<tr>
<td>$g_2$</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>

### Table 3.4

Optimal nDCG gains derived for "Bronze" assessors (shared across all the queries) under varying level of label noise by mixing up labels "0" and "1". The more the label noise, the less the gap between optimal $g_0$ and $g_1$ as these two grades become less distinguishable.

<table>
<thead>
<tr>
<th>Injected Label Noise Level</th>
<th>0%</th>
<th>5%</th>
<th>10%</th>
<th>15%</th>
<th>20%</th>
<th>25%</th>
<th>30%</th>
<th>35%</th>
<th>40%</th>
<th>45%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_0$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$g_1$</td>
<td>1.3</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.8</td>
<td>0.7</td>
<td>0.6</td>
<td>0.5</td>
<td>0.3</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$g_2$</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>
Chapter 4

Better Ordinal Label Aggregation: a Latent Trait Model

To assemble training and testing collections for information retrieval, ground-truth relevance assessments were traditionally obtained from expensive expert assessors, as in numerous TREC efforts\(^1\). The advent of Amazon Mechanical Turk\(^2\) and other crowdsourcing services has given rise to an alternative and far less expensive source of relevance assessments. However, while crowdsourced relevance assessments are far cheaper, they are also less reliable and more difficult to accurately obtain for graded relevance assessments, where a common grade\(^3\) scale is \(g \in \{0, 1, 2, 3, 4\}\).

Here we propose a new and simpler framework for the inference of true document relevance from crowdsourced assessments which achieves a performance exceeding state-of-the-art techniques, sometimes markedly so. In our proposed framework, both assessor and document parameters are estimated. Assessor quality is measured as the usefulness for inference, precisely computed as the reduction in uncertainty for relevance inference; thus, assessor quality is closely integrated with how much the assessor’s grade counts. Assessor grades are modeled as Gaussian distributed conditionals for each available relevance grade. Each assessor is modeled individually, accounting for: (1) the notion that higher assessor grades, for a good quality worker, correspond on average with higher document true relevance (random and adversarial assessors are modeled within our framework); (2) realistic decomposition into Gaussians of conditional densities; (3) suitability

---

\(^1\)This chapter is based on our paper [35].
\(^2\)http://www.mturk.com/
\(^3\)Pairwise preference judgments conducted on pairs of documents, being an alternative to ordinal-grade relevance judgments, were shown to yield a higher inter-assessor agreement [9], however it is challenging to collect them efficiently [4], and they are also likely to be less informative as they lack information about the preference magnitude.
for crowdsourced grades in terms of amount of data collected; (4) the ability to represent grade variance as a function of the assessor, the actual grade, and the document.

For each document, we model its true relevance and difficulty as continuous variables. The true relevance scale (as opposed to an imposed grade scale) allows us to understand and model the perception of each assessor for each grade separately. Naturally, the continuous scale allows for aggregation of crowd-answers on inconsistent scales, for example a NIST assessor with two grades and Mechanical Turkers with five grades. Finally, modeling document difficulty allows for variance in assessor grade due to the documents themselves, as opposed to limiting this variance to be a function of only the assessor and the nominal grade, (e.g., through the use of true-relevance/assessed-grade confusion matrices).

We describe two methods, one based on data likelihood over individual documents and the other based on likelihood over pairs of documents. We estimate all the parameters of our model from crowdsourced data, demonstrating the inference of assessor parameters and quality as well as document difficulty and relevance. Both document-likelihood and document-pair-likelihood optimizations use information directly from the input data and show better performance than existing approaches such as simple averages or majority vote, variants of expectation-maximization aggregation [14, 21, 48], and Polytomous Rasch aggregation [1, 34].

In summary, our contributions in this Chapter are as follows: (1) A new model for assessors that includes a novel measure of quality for each assessor, is easier to learn, and is more suitable for crowdworkers (Section 4.3). (2) A new model for documents that incorporates a continuous relevance scale and a document difficulty parameter (Section 4.3). (3) A new method for the aggregation of crowdsourced assessments based on likelihood of documents, including the inference of document relevance and difficulty as well as assessor quality. (Section 4.4.1). (4) A new method for aggregation of crowdsourced assessments based on likelihood of pairs of documents (section 4.4.2). (5) A new measure of assessor quality (“informativeness”) based on a sound information-theoretic measure of reduction in uncertainty due to the assessor (Section 4.4.7). (6) A novel analysis of aggregation quality as function of number of assessments, by means of confidence intervals for inferred document relevance (Section 4.4.6). (7) An analysis of document difficulty vs. crowd disagreement (Section 4.4.8).

We demonstrate significant, at times marked, improvements for all methods proposed on four TREC datasets. Our crowdsource aggregation results are presented in Section 4.4.5.
4.1 Related work

In the literature, there are two main directions of research related to conducting inference about unknown quantities of interest, given human observations about them, that may potentially be biased and inaccurate. The first direction deals with so called latent class models, where the unknown quantity is discrete. One of the most popular examples is EM algorithm [14, 21], where a human assessor quality is modeled as a latent confusion matrix with off-diagonal elements representing the probability of mislabeling each item they provide an assessment for. The true labels and the confusion matrices are estimated through an iterative procedure that maximizes the likelihood of observing the assessments given the parameters of the model (see Figure 4.1). As a result, the estimate of the true label is determined mostly by accurate workers, while “bad” workers are weighted out. There are further extensions of this method within Bayesian inference framework aimed to regularize the problem by assuming various prior distributions over confusion matrices and also to improve the quality of inference [30, 31, 43, 52]. An extension modeling item difficulties and solving the problem via minimax entropy principle was proposed in [59], and later further extended to the case of ordinal labels by regularizing the problem with additional ordinality constraints [58]. Another extension of EM by modeling item difficulties was presented in [56], for the case of 2x2 confusion matrices.

For our domain of application (i.e., inferring document relevances), however, the second direction of research dealing with latent trait models [51] seems more promising. In these models, each item (document in our case) is assumed to have a continuous latent trait (true unknown document relevance), while each assessor is assumed to have a set of individual thresholds roughly defining the boundaries, on the scale of latent traits, between adjacent classes (relevance classes in our case). For each item and each assessor, the observed labels are assumed to be generated by a probabilistic model that depends on both item latent trait and assessor thresholds, and additionally may account for variable measurement errors for both assessor (his quality) and item (its difficulty). Polytomous Rasch model [1, 34] can be seen as a special case of a latent trait model when all the assessors have equal measurement errors. Originally, all the items were modeled to have similar measurement errors as well. However, there is known to exist an extension [39] of the basic Polytomous Rasch model for the case of variable difficulties of items (the qualities of workers are, however, modeled to be similar). The methods that we investigate in this chapter resemble the one described in [51] which accounts for variable difficulties of assessors and has been shown to work well on medical datasets, the
Figure 4.1: Refining relevance assessments with Expectation-Maximization (EM); unreliable relevance assessments are collected (e.g., via crowdsourcing), each document being labeled by several workers; EM algorithm is applied to estimate (1) worker qualities in a form of Confusion Matrices and (2) distributions over "true" relevance grade for each document. Two extreme examples of worker quality are displayed: a diligent worker ($w_1$) and a greedy uninformative worker ($w_m$).

The difference is that we choose a more flexible family of functions to model class probability curves, and we also accommodate for item difficulty. Other applications of latent trait models to label aggregation that consider both variability in labeler accuracies and item difficulties [25, 46] are likely limited in scalability due to MCMC simulation they have to rely on to conduct inference. In another recent application [28] a more computationally efficient method (via Approximate Inference) is proposed, however it may be limited by the assumption of same (fixed) thresholds being shared among all the assessors (also authors show no improvement over EM [14] in terms of nDCG).

Other related work includes methods for preference aggregation [13, 53] and ranked lists aggregation [41]. While the problem of ordinal assessment aggregation could potentially be reformulated within either aforementioned framework, it is clear that this transformation would be lossy as the information about degree of confidence with respect to pairwise preferences (encoded with assessor-specific distance between relevance grades) is discarded.

A method for measuring assessor performance in units of information (bits)
<table>
<thead>
<tr>
<th></th>
<th>Trec8</th>
<th>Ent07</th>
<th>CW12</th>
<th>RF10</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CROWD AGGREG</strong> (crowd assessments)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Queries</td>
<td>10</td>
<td>33</td>
<td>9</td>
<td>100</td>
</tr>
<tr>
<td>Rel grades, expert</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Rel grades, crowd</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Documents total</td>
<td>18.2K</td>
<td>3.0K</td>
<td>3.2K</td>
<td>20.0K</td>
</tr>
<tr>
<td>Doc per query mean</td>
<td>1826</td>
<td>91</td>
<td>359</td>
<td>200</td>
</tr>
<tr>
<td>Doc per query med</td>
<td>1798</td>
<td>80</td>
<td>368</td>
<td>199</td>
</tr>
<tr>
<td>Crowdworkers</td>
<td>661</td>
<td>132</td>
<td>328</td>
<td>764</td>
</tr>
<tr>
<td>Assessments, total</td>
<td>134K</td>
<td>18.5K</td>
<td>56.3K</td>
<td>98.3K</td>
</tr>
<tr>
<td>Assess/worker mean</td>
<td>203</td>
<td>140</td>
<td>172</td>
<td>129</td>
</tr>
<tr>
<td>Assess/wkr med</td>
<td>39</td>
<td>40</td>
<td>30</td>
<td>18</td>
</tr>
<tr>
<td>Assess/wkr/query mean</td>
<td>106</td>
<td>39</td>
<td>47</td>
<td>16</td>
</tr>
<tr>
<td>Assess/doc mean</td>
<td>7.3</td>
<td>6.2</td>
<td>17.4</td>
<td>4.9</td>
</tr>
<tr>
<td>Assess/doc med</td>
<td>6.5</td>
<td>5</td>
<td>15</td>
<td>5</td>
</tr>
<tr>
<td>Queries/worker mean</td>
<td>1.9</td>
<td>3.6</td>
<td>3.6</td>
<td>7.9</td>
</tr>
<tr>
<td>Queries/worker med</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td><strong>TESTING LTR MODEL</strong> (expert assessments)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Queries</td>
<td>40</td>
<td>17</td>
<td>41</td>
<td>584</td>
</tr>
<tr>
<td>Documents, total</td>
<td>68.8K</td>
<td>11.3K</td>
<td>11.9K</td>
<td>15.5K</td>
</tr>
</tbody>
</table>

Table 4.1: Top: datasets for label aggregation and training rankers. Bottom: datasets for testing learning-to-rank models (to be used later in Chapter 5).

based on information-theoretic approach was proposed in [55]. Since it was defined within the framework of latent class models, we adapted it to the continuous case, and devised a suitable measure of the informativeness.

4.2 Experiment setup, data, crowd assessments

We run experiments over four datasets with slightly different characteristics: TREC 8, TREC Enterprise 2007, ClueWeb 2012 (subset of it that was used in TREC 2013 Crowdsourcing Track [49]) and ClueWeb 2009 (subset of it that was used in TREC 2010 Relevance Feedback track + Million Query 2009). We perform the label aggregation task and train learning-to-rank algorithms on a subset of the queries (see Table 4.1, top for statistics), while using the rest of the queries to test the performance of trained rankers (see Table 4.1, bottom). In each dataset, the “gold” labels are TREC assessments obtained from expert annotators, present over both training and testing sets. For the former three collections, for each document from the training set we have collected at least 5 additional assessments from crowdworker
annotators (Amazon Mechanical Turkers); for the latter collection we reused existing crowdsourced judgments collected previously in [8]. The overall goal is both to (1) aggregate these non-expert labels in a meaningful way to estimate the actual relevance, and also (2) train a ranking algorithm with these crowdsourced assessments. For learning-to-rank, we extract a typical set of features (language models and web-graph properties) for each query-document, on all four collections.

4.3 Assessor and document models

Each assessment observation\(^4\) consists of a document \(d\), an assessor \(a\), and the grade observed \(g\), in short \((g,d,a)\). We are assuming a uni-dimensional relevance scale (typical for IR relevance assessments), but continuous as opposed to discrete. Each document \(d\) is assumed to have “true relevance” \(r_d\) on this scale, and a “difficulty” \(\epsilon_d > 0\) centered around 1. A fixed relevance prior is used on documents, centering document relevance to a common reference value (arbitrarily chosen to be 0 in our experiments), and having a unit variance. The actual document model \(\theta_d = (r_d, \epsilon_d)\) is to be inferred from assessments, starting from this prior and using crowdworkers assessments to optimize the likelihood (see Figure 4.2).

We are modeling each assessor \(a\) with a set of sigmoid-like density functions \(p(\text{grade}|r_d)\), one for each relevance grade (Figure 4.3). Intuitively, we want the densities to reflect probability of an assessor grade conditional to the true relevance, but we also want them to be decomposable into a set of underlying parametrizable simple functions. The model assumption is that, for any document \(d\), there is an underlying set of grade-gaussians, such that each density corresponds to a normalized gaussian component. The mean of each gaussian is invariant in \(d\) (fixed per assessor-grade), while the variance of each gaussian component depends both on assessor variance \(w_g\) for the grade \(g\), and on document difficulty \(\epsilon_d\) :

\[
\sigma^2 = w_g \epsilon_d.
\]

The difficult documents increase the variance already modeled by

---

\(^4\)Here we assume all assessments belong to one query. The generalization to multiple queries case is straightforward.
Figure 4.2: Refining relevance assessments with Grade-Document-Assessor method (GDA): unreliable relevance assessments are collected, each document being labeled by several workers; GDA algorithm is applied to estimate (1) worker qualities in a form of Grade Probability curves and (2) distributions over continuous "true" relevance for each document. Two extreme examples of worker quality are displayed: a diligent worker (\(w_1\)) and a greedy uninformative worker (\(w_m\)).

31
order labels) and we want to use the “reversed” information out of their assessments. If ordinality constraints are enforced, the algorithm will automatically disregard the adversarial judgments – like in the case of random, or uninformative, judgments.

4.4 Aggregation of crowdsourced assessments

Given a set of assessments $OBS = \{(g, d, a)\}$, a likelihood objective is defined, followed by an optimization procedure which finds the assessor and document parameters $\theta_a, \theta_d$ that achieve the maximum likelihood. Generally, the optimization is based on Newton-Raphson procedure [42], often far more efficient than Gradient Descent. We present below two likelihood objectives: first a likelihood objective defined on individual documents; second a likelihood defined on pairs of documents.

4.4.1 Likelihood on documents: “GDA" method

To define a likelihood for observations based on individual documents, the probability of observation (grade $g$, document $d$, assessor $a$) is given by a density of the grade conditional on the assessor and document models$^6$

$$p(g|\theta_d, \theta_a) = \frac{1}{Z_{ad}} \pi_a \mathcal{N}(r_d | \mu_a, w_a \epsilon_d)$$ (4.1)

where $Z_{ad} = \sum_g \pi_a \mathcal{N}(r_d | \mu_a, w_a \epsilon_d)$ is a normalizing sum of gaussian components. Note that variance of each gaussian component is the product of assessor variance and document difficulty: $w_a \epsilon_d$. The log likelihood is

$$\log L = \log \prod_{(g,d,a) \in OBS} p(g|\theta_d, \theta_a)$$ (4.2)

Maximizing likelihood with numerical optimization (using analytically derived gradients), we obtain $\theta_a^* = (\pi_a^*, \mu_a^*, w_a^*)$ for each assessor and $\theta_d^* = (r_d^*, \epsilon_d^*)$ for each document.

Knowing assessor model $\theta_a$ we can reason about the quality of the assessor $a$ in terms of informativeness of his judgments. We can compute how much information about the document relevance we gain by obtaining a judgment $g$ from him. To this end, we take the difference between entropy of the prior distribution over relevance

$^6$The name “GDA" is due to resemblance of the main expression (eq. 4.1) to Gaussian Discriminant Analysis.
and the entropy of posterior distribution given the judgment:

\[ I_g(\theta_a) = H[p_0(r)] - H[p(r|g, \theta_a)] \]

\[ = H[p_0(r)] - H\left[ \frac{p_0(r) \cdot p(g|r, \theta_a)}{p_a(g)} \right], \]  (4.3)

where \( p_0(r) \) is the prior distribution over relevance for any unjudged document that we choose to be a Gaussian with zero mean and unit variance; \( p(g|r, \theta_a) \) is given by our model (eq. 4.1) assuming the document is of average difficulty \((\epsilon_d = 1); p_a(g) \) is the normalizing constant with the meaning of overall frequency of the grade \( g \) by assessor \( a \):

\[ p_a(g) = \int_{-\infty}^{\infty} p_0(r) \cdot p(g|r, \theta_a) dr \]  (4.5)

Since \( g \) is not known in advance, we should take expectation of \( I_g(\theta_a) \) over all possible \( g \), with (eq. 4.4) and (eq. 4.5):

\[ I(\theta_a) = \sum_g p_a(g) \cdot I_g(\theta_a) \]  (4.6)

\( I(\theta_a) \) is the expected informativeness of a single judgment of the assessor \( a \), it is measured in nats (natural units for information entropy, \( 1 \text{ nat} \approx 1.44 \text{ bits} \)) and depends only on the parameters of the user model \( \theta_a \).

### 4.4.2 Likelihood on document pairs: “GDA-PW”

For a likelihood of observations on pairs of documents, the probability of two observations \((g, d, a)\) and \((g', d', a')\) is given by density of the grade conditionals of assessors \( a \) and \( a' \) and the two documents models:

\[ p(g|g', \theta_d, \theta_{d'}, \theta_a, \theta_{a'}) = \frac{1}{Z_{aa'd'd'}} \pi_g^a \mathcal{N}(r_d - r_{d'}|\mu^a_g - \mu^{a'}_g, \sqrt{\omega_g^a w_g^{a'} \epsilon_d \epsilon_{d'}}) \]  (4.7)

where \( Z_{aa'd'd'} \) is the normalization constant. Note that gaussian means are in this model the difference between assessor grade means, and variance of each gaussian component is composed of assessor variance parameters (for both grades) and document difficulty (for both documents): \( \sqrt{\omega_g^a w_g^{a'} \epsilon_d \epsilon_{d'}} \).

Denoting by \( K \) the overall number of judgments, we define the log likelihood\(^7\):

\[ \log L = \sum_{(g,d,a),(g',d',a') \in \text{OBS}} (K)^{-1} \log p(g|g', \theta_d, \theta_{d'}, \theta_a, \theta_{a'}) \]  (4.8)

\(^7\)Alternatively, one could consider in (eq. 4.8) only pairs judged by the same assessor \( a = a' \).
For the pairwise model, we can define informativeness of the assessor as well: it becomes the expected reduction of uncertainty of the relative relevance for a pair of documents $\Delta r = r_d - r_{d'}$, given a pair of judgments $g$ and $g'$:

$$I_{gg'}(\theta_a) = H[p_0(\Delta r)] - H\left[\frac{p_0(\Delta r) \cdot p(g, g'|\Delta r, \theta_a)}{p_a(g, g')}\right],$$

(4.9)

where $p_0(\Delta r)$ is the prior distribution over difference of relevance for any unjudged pair of documents (we choose it to be a Gaussian with zero mean and variance of 2); we model $p(g, g'|\Delta r, \theta_a) \propto \sqrt{p(g|g', \Delta r, \theta_a) \cdot p(g'|g, \Delta r, \theta_a, \theta_a)}$ (follows from eq. 4.8) and compute it using (eq. 4.7) (again, we assume $\epsilon_d = \epsilon_{d'} = 1$); $p_a(g, g')$ is the normalizing constant:

$$p_a(g, g') = \int_{-\infty}^{\infty} p_0(\Delta r) \cdot p(g, g'|\Delta r, \theta_a)d\Delta r$$

(4.10)

Like before we take the expectation of $I_{gg'}(\theta_a)$ over all possible $g$ and $g'$. Also, we divide the result by 2, to report the informativeness per judgment, not pair of judgments. With (eq. 4.9) and (eq. 4.10):

$$I(\theta_a) = \frac{1}{2} \sum_{g, g'} p_a(g, g') \cdot I_{gg'}(\theta_a)$$

(4.11)

### 4.4.3 Baseline comparison: EM algorithm

Expectation Maximization (EM) is an iterative method that finds the maximum likelihood estimator of a parameter $\theta$ of a parametric probability distribution, where this distribution depends on unobserved latent variables.

It has two steps: expectation (E) uses current estimate for the parameters to evaluate the expectation (over the latent variables distribution) of the likelihood function, and maximization (M) recomputes the next estimate for the parameters by maximizing the expected likelihood generated in the E step.

In our model, the observed judgments from crowdsource workers are the input; worker qualities expressed in the form of confusion matrices (one per worker) are unknown model parameters to be estimated (since we don’t know the workers background, motivation, native ability, familiarity with English, etc.). The document "true" grades are the unknown latent variables. We take a similar approach as in [21]. On the output of EM algorithm we get: (1) the maximum likelihood estimates of each worker quality, and (2) the distribution of the "true" relevance grades for each document (see Figure 4.1).
Assuming $G$ is the number of relevance grades and $M$ is the number of workers, we assign each worker a latent confusion matrix $C$ of the size $G \times G$. Each element $c_{jl}$ is the probability that worker $j$ labels the document $l$, given ground truth $g$. Let $n_{il}^j$ be an indicator of whether the judge $j$ assigns a label $l$ to the document $i$ (i.e., $n_{il}^j = 1$) or not (i.e., $n_{il}^j = 0$). After computing confusion matrices for all workers, for each document we estimate a relevance grade based on all workers’ judgments and matrices. The probability of document $i$ having true relevance grade $r$ given each worker $j$’s accuracy confusion matrix, and judgments is:

$$Pr(R_{ir}) = \frac{p_r \cdot \prod_{j=1}^{M} \prod_{l=1}^{G} (c_{il}^j)^{n_{il}^j}}{\sum_{k=1}^{G} p_k \cdot \prod_{j=1}^{M} \prod_{l=1}^{G} (c_{il}^j)^{n_{il}^j}}$$

We repeat estimation of worker confusion matrices (M step) and calculation of distributions over document relevance grades (E step) until the results converge. We consider this process to have converged when, for each document $i$, the difference between $Pr(R_{ir}) \forall r$ at iteration $t - 1$ and at iteration $t$ is less than or equal to 0.01 for all $r$.

While shown to work well on some datasets, in general this method is not very robust to sparse input: it requires many judgments per worker to reliably estimate worker parameters (typically, proportional to $G^2$ or $GT$ when number of “true” classes $T$ is chosen different from $G$). Also, this EM method was not specifically designed for problems dealing with continuous quantities (like relevance), but rather with discrete classes. So it is not quite clear how many discrete latent classes for relevance we have to pick (with more classes we expect more accurate approximation of continuous relevance, but also increased number of free parameters making the model even more prone to overfitting).

Another shortcoming of EM in the current setup is the lack of guarantee for ordinality: it is not the case that if the latent class $t + 1$ corresponds to a higher degree of relevance than the class $t$, then class $t + 2$ should always be expected more relevant than class $t + 1$.

### 4.4.4 Baseline comparison: Polytomous Rasch

In Polytomous Rasch model, the probability of assessor $a$ assigning a label $g$ to the document $d$ is modeled as follows:

$$p(g|r_d, \tau^a, \epsilon_d) = \frac{\exp \left( \sum_{k=0}^{G} \epsilon_d (r_d - \tau^a_k) \right)}{\sum_{j=0}^{G-1} \exp \left( \sum_{k=0}^{G} \epsilon_d (r_d - \tau^a_k) \right)}$$
where \( r_d \) is a scalar latent relevance associated with the document \( d \), and \( \tau^a \) is a vector of latent thresholds associated with assessor \( a \). When \( r_d = \tau^a_g \) for some \( g \), the worker is equally likely to assign labels \( g - 1 \) and \( g \) (other labels have some likelihood as well). The parameter \( \epsilon_d > 0 \), supposed to be equal to 1 originally, extends the basic model and represents document difficulty [39]. This model has very few free parameters, as compared to EM, and thus the inference is more robust when the input is sparse. On the downside, it works well only when the assessors are reliable. Notice that an assessor providing random responses (whether uniformly across the grades or not) cannot be accurately modeled within this framework. Therefore, this method (in the current form) is not very appropriate for crowdsourcing, as we expect a massive number of unreliable judgments.

In our experiments, we use the extended version of Polytomous Rasch model (accounting for document difficulty), and estimate all the parameters by maximizing their joint likelihood in a very similar way we perform optimization of the models we propose (GDA and GDA-PW).

### 4.4.5 Label Aggregation Results

The task in this experiment was to aggregate crowdsourced labels for datasets from Table 4.1, rank the documents for each query by estimated relevance based on aggregated labels, and then evaluate these rankings by using expert labels in terms of common IR measures: MAP, nDCG and Kendall \( \tau \). We compare the performance of proposed label aggregation methods, namely GDA (eq. 4.2) and GDA-PW (eq. 4.8), with performance of several baselines: (a) randomly ordered documents which doesn’t rely on training labels at all, (b) majority vote, by ordering documents by the most frequent label (or one of them if there are several), (c) ordering documents by straight average of crowdsourced labels, (d) expectation-maximization method, see Section 4.4.3, and (e) Polytomous Rasch model, see Section 4.4.4. To measure the effect of modeling document difficulty, we run GDA-PW in two modes: with and without it (fixed \( \epsilon = 1 \)). The results are presented in Table 4.2. First, we observe that our methods, GDA and GDA-PW, outperform the baselines on all datasets by all measures, in many cases by a wide margin\(^8\). Between these two, the pairwise method (GDA-PW) performs better on all datasets. Modeling document difficulty seems to help, especially in case of TREC 8 and ClueWeb 2012. It should be noted that EM method is not performing very well: it is comparable to Straight Average on two datasets, similar on third and even worse on the fourth, ClueWeb 2012. Perhaps that can be explained by analyzing the dataset statistics (see Table 4.1):

\(^8\)We test for significance of improvement using Fisher’s two-sided paired randomization test [50].
Table 4.2: Performance of various methods for aggregation of crowdsourced assessments, evaluated on 4 different datasets. The highest value in each column is highlighted. Values significantly lower (p < 0.05) than those of GDA-PW are marked with *.

to reliably estimate assessor models (confusion matrices in case of EM) we have too few assessments per worker per query (particularly, on the worst performing dataset, ClueWeb 2012). To partly mitigate this problem, we minimized the number of unknown parameters by assuming each assessor has the same model across all queries (which may not be very realistic), and also by reducing the number of discrete "true" classes of relevance (3 true classes worked best in almost all cases); we report the performance of best working configuration, but nevertheless the algorithm suffers from data sparsity. Polytomous Rasch model is less prone to overfitting on sparse datasets as it has fewer free parameters to optimize, so it tends to perform better than EM.

As an additional experiment, we decided to inject labels of expert assessors along with randomly generated labels of a virtual assessor, into the set of labels to be aggregated, without giving the algorithms a hint about the special properties of these two additional sources. We observe (see Table 4.3) that the previous conclusions are generally still valid. Our methods do a good job at recognizing the reliable sources, picking up the expert labels to further improve the performance over the rest of the methods.

### 4.4.6 Aggregation quality as function of number of assessments

When collecting crowdsourced relevance judgments for a document, we not only can estimate its relevance (assuming we already have some estimates of the qualities of assessors who contributed judgments), but more generally, we also can compute the posterior probability distribution of the true relevance. Having the posterior distribution can be very useful, for instance, to compute confidence intervals for the estimated relevance. If confidence intervals are too wide, then we may want to collect additional assessments for that document in order to obtain

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP</th>
<th>nDCG</th>
<th>K_τ</th>
<th>MAP</th>
<th>nDCG</th>
<th>K_τ</th>
<th>MAP</th>
<th>nDCG</th>
<th>K_τ</th>
<th>MAP</th>
<th>nDCG</th>
<th>K_τ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.040*</td>
<td>0.432*</td>
<td>0.000*</td>
<td>0.553*</td>
<td>0.669*</td>
<td>0.000*</td>
<td>0.761*</td>
<td>0.815*</td>
<td>0.000*</td>
<td>0.420*</td>
<td>0.618*</td>
<td>0.000*</td>
</tr>
<tr>
<td>MV</td>
<td>0.309*</td>
<td>0.724*</td>
<td>0.085*</td>
<td>0.290*</td>
<td>0.751*</td>
<td>0.295*</td>
<td>0.088*</td>
<td>0.902*</td>
<td>0.387*</td>
<td>0.306*</td>
<td>0.660*</td>
<td>0.151*</td>
</tr>
<tr>
<td>Average</td>
<td>0.368*</td>
<td>0.753*</td>
<td>0.714*</td>
<td>0.767*</td>
<td>0.810*</td>
<td>0.446*</td>
<td>0.916*</td>
<td>0.920*</td>
<td>0.475*</td>
<td>0.609*</td>
<td>0.697*</td>
<td>0.372*</td>
</tr>
<tr>
<td>EM</td>
<td>0.373*</td>
<td>0.763*</td>
<td>0.713*</td>
<td>0.762*</td>
<td>0.809*</td>
<td>0.449*</td>
<td>0.920*</td>
<td>0.928*</td>
<td>0.501*</td>
<td>0.618*</td>
<td>0.714*</td>
<td>0.420*</td>
</tr>
<tr>
<td>Rasch</td>
<td>0.448*</td>
<td>0.795*</td>
<td>0.748*</td>
<td>0.770*</td>
<td>0.815*</td>
<td>0.467*</td>
<td>0.928*</td>
<td>0.932*</td>
<td>0.521*</td>
<td>0.652*</td>
<td>0.741*</td>
<td>0.480*</td>
</tr>
<tr>
<td>GDA</td>
<td>0.447*</td>
<td>0.791*</td>
<td>0.753*</td>
<td>0.772*</td>
<td>0.819*</td>
<td>0.484*</td>
<td>0.931*</td>
<td>0.932*</td>
<td>0.527*</td>
<td>0.660*</td>
<td>0.747*</td>
<td>0.479*</td>
</tr>
<tr>
<td>GDA-PW(ε = 1)</td>
<td>0.463*</td>
<td>0.802*</td>
<td>0.756*</td>
<td>0.776*</td>
<td>0.821*</td>
<td>0.484*</td>
<td>0.932*</td>
<td>0.936*</td>
<td>0.527*</td>
<td>0.664*</td>
<td>0.769*</td>
<td>0.484*</td>
</tr>
</tbody>
</table>
more reliable estimates. For our first method (eq. 4.2) the posterior has a simple

expression (assuming we have $K$ independent judgments for that document):

$$p(r_d | \{g_1, g_2, \ldots, g_K\}, \theta) \propto p_0(r_d) \prod_{k=1}^{K} p(g_k | r_d, \epsilon, \theta_{\alpha_k})$$  \hspace{1cm} (4.12)

At the same time, the posterior for the second method (eq. 4.8) doesn’t seem

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP</th>
<th>nDCG</th>
<th>$K_r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MV</td>
<td>0.219</td>
<td>0.647</td>
<td>0.519</td>
</tr>
<tr>
<td>Average</td>
<td>0.371</td>
<td>0.771</td>
<td>0.755</td>
</tr>
<tr>
<td>EM</td>
<td>0.503</td>
<td>0.814</td>
<td>0.697</td>
</tr>
<tr>
<td>Rasch</td>
<td>0.415</td>
<td>0.785</td>
<td>0.780</td>
</tr>
<tr>
<td>GDA</td>
<td>0.655</td>
<td>0.896</td>
<td>0.941</td>
</tr>
<tr>
<td>GDA-PW($\epsilon = 1$)</td>
<td>0.674</td>
<td>0.895</td>
<td>0.955</td>
</tr>
<tr>
<td>GDA-PW</td>
<td>0.685</td>
<td>0.897</td>
<td>0.958</td>
</tr>
</tbody>
</table>

Table 4.3: Performance of various methods for aggregation of crowdsourced assessments. NIST assessments and random assessments from a virtual assessor are added to the training set. The large jump in performance is no doubt due to NIST assessors (same assessments also used for evaluation), see the red dots in Figure 4.8.

Figure 4.4: Simulation of online experiment for 4 documents of ClueWeb 2012 collection. Relevance judgments are collected one by one, each time improving the knowledge about true relevance. For each document, the true relevance estimates are plotted along with 50% confidence intervals.
to have a closed form expression. As an alternative, we can readily apply the
Laplace approximation for the posterior (refer to [5]), given that we anyway have
to compute the second derivatives of likelihood function with respect to the model
parameters while performing Newton-Raphson optimization procedure. A more
general and potentially more accurate approach would be to employ the methods
of approximate inference [5] to approximate joint posterior distribution for all the
parameters of the model.

In Figure 4.4 we plot the results of simulating online experiment when we col-
lect assessments for a few documents with different relevance (as per expert as-
sessor), one by one, using (eq. 4.12). One can see how our knowledge about the
initially unknown relevance improves over time. The more judgments we collect,
the tighter confidence intervals become. We can reason about assessor reliability
indirectly, by looking at the reduction of uncertainty caused by each assessment:
as a consequence of bayesian inference, high quality assessors will reduce uncer-
tainty more than low quality workers. Abrupt changes in the expected value will
also correspond to high-quality assessments.

4.4.7 Analysis: Assessor Informativeness

In Figure 4.5 we can observe the user models inferred from the crowdsourced data
using document likelihood (eq. 4.2). These models correspond to several very dif-
ferent assessors. To verify the validity of inference, we analyze the results of exper-
iment when labels of expert TREC assessors, along with randomly generated labels
of a virtual “random” assessor, were included into the set of crowdsourced labels.
Thus, we can analyze the model for two assessors for which we have a priori knowl-
edge about their skills. The upper row on the plots corresponds to an expert, then
we show models for three real crowdworkers in decreasing order of their inferred
quality, while the bottom row is taken by a random virtual assessor. We start with
the left column where we plot the inferred raw Gaussian for each assessor and each
grade. These Gaussians are used as building blocks to be combined into category
probability curves which we observe in the middle column: for instance, if we pick
some assessor, the black curve gives us the probability of that assessor assigning
label “0” given some particular document “true” relevance; all the curves sum up
to 1. On the right plot we can see the joint probability distributions for true rele-
vance and observed grades. The plot should be interpreted as follows: pick some
assessor and look at, let’s say, the black curve, corresponding to documents labeled
“0”. It shows how all the documents labeled “0” by that assessor are expected to
be distributed over the true relevance, while the area under that curve is propor-
**Figure 4.5:** User model parameters inferred from crowdsourced judgments for TREC 8 dataset query #420 (top three plots) and ClueWeb 2012 dataset query #246 (bottom three plots): components of Gaussian mixture (left); grade probability curves (middle); joint probability distributions for relevance and grades (right). From top to bottom on each plot: (a) NIST assessors combined; (b) a good quality crowdworker; (c) an average quality crowdworker; (d) a poor quality crowdworker, (e) simulation of an assessor who provides random judgments (not necessarily uniform across grades). The more separated joint probability curves correspond to higher quality assessors (that can better distinguish between the grades).
tional to the overall likelihood of this assessor assigning this grade. Intuitively, the better separated the curves corresponding to different grades, the better is the assessor able to distinguish between these grades, meaning the more informative. The reported results match our intuition: we have well separated curves for expert assessors (top), while we have almost indistinguishable curves (up to normalization constant) for the random one (bottom). Besides, we see that the separability of these curves for the actual crowdworkers vary in a wide range, suggesting we are indeed dealing with assessors of very different quality. We conclude that the family of functions we employ in our aggregation framework is flexible enough to model assessors with a wide range of qualities and biases.

The reported results match our intuition: we have well separated curves for expert assessors (top), while we have almost indistinguishable curves (up to normalization constant) for the random one (bottom). Besides, we see that the separability of these curves for the actual crowdworkers vary in a wide range, suggesting we are indeed dealing with assessors of very different quality. We conclude that the family of functions we employ in our aggregation framework is flexible enough to model assessors with a wide range of qualities and biases.

Next, we measure the informativeness of each assessor not just visually, but quantitatively, computing it using (eq. 4.11). We demonstrate that the inferred informativeness of each assessor makes sense, by plotting it against the measure of how consistent the judgments of that assessor are with the ones of the expert, in terms of Kendall $\tau$ coefficient for rank correlation. By looking at Figure 4.8, we observe that there is indeed high correlation between two plotted quantities (it is particularly true for the cases when we have many judgments made by an assessor). Moreover, we observe almost zero inferred informativeness for random virtual workers and a considerably high informativeness for expert assessors.

Another interesting observation: if we rank expert assessors for different collections by their informativeness, the order would be (from lower to higher): TREC

Figure 4.6: Top plots: the distribution of crowdworkers by their inferred informativeness, for each dataset. Bottom plots: the distribution of collected assessments by their expected informativeness. Overall, we observe too many poor quality workers providing too many poor quality judgments (particularly for ClueWeb 2012 dataset).
8, TREC 2010 RF, TREC Enterprise 2007 and finally ClueWeb 2012. That can be explained by the granularity of the judgments provided by these experts: binary judgments, 3, 3 and 4 possible relevance grades for these collections, respectively. While perhaps all these assessors are good in ordering relevant documents vs non-relevant, the judgments on 4 grades scale have, intuitively, much more discriminatory power than binary relevance judgments (moreover, most of the judgments are “0” and only small fraction are “1” for the latter case). Also, note that informativeness of crowdworkers for TREC 2010 RF, on average, is lowest among all the collections: it may be because they were asked to report relevance on 3-grade scale, the least discriminative one.

![Difficulty vs Relevance](image1)
![Consistency of Document Difficulty](image2)

**Figure 4.7:** Document difficulties for TREC 8. Left: document difficulties plotted against estimated relevances. Right: document difficulties estimated from labels provided by one group of crowdworkers being consistent with those of another group.

We conclude the analysis of informativeness by showing macro-level statistics for each dataset (see Figure 4.6). The main observation is that we have too many poor quality workers bringing in too many poor quality judgments (particularly for ClueWeb 2012 dataset; for TREC 8 and TREC Enterprise 2007 we had a stricter rejection mechanism based on redundant trap questions). This is a good illustration of why measuring the informativeness of the assessors is so important in detecting unconscientious workers as early as possible, and perhaps to incentivize workers by paying for the amount of useful information they bring in to the system [48,55].

### 4.4.8 Analysis: Document Difficulty

The label aggregation models that we propose in this chapter include a per-document parameter $\epsilon_d$ which captures the degree of inter-assessor disagreement (adjusted
for assessor reliability). Previously, we demonstrated empirically that this parameter may help improving the overall quality of aggregation. Now we explore its properties in more details, on the example of aggregating judgments of crowdworkers (not including expert labels) for TREC 8 collection. First, in Figure 4.7 (left) we explore the pattern of interaction between document difficulty and document relevance, both inferred from judgments (for convenience, the points are colored according to independent NIST labels). One can notice that relevant documents (as per NIST) tend to have a higher spread of difficulty (and several of them being the most difficult); the easy documents tend to be either certainly relevant or certainly non-relevant ones; the in-between documents are more difficult than average (as they cause more disagreement). Secondly, we show that the document difficulty we infer indeed represents an intrinsic property of the document, not just an assessing effect: we split the crowdworkers into two separate groups, so that most of the documents have judgments from both groups. Then we train two separate models, one for each group, each model having independent estimates for each document difficulty. Few of them are plotted on Figure 4.7 (right): there is a positive correlation between two estimates of difficulty.

Separately, we report a weak correlation between inferred document difficulty and the time it took a crowdworker to complete the judgment (we recorded duration of each judgment while collecting crowdsourced labels).
Figure 4.8: Measured informativeness of assessor vs. Kendall $\tau$ coefficient for rank correlation between the assessor and an expert. The two quantities are well correlated, hence the informativeness is a good measure of assessor quality. Random workers have a very low informativeness, unlike experts (NIST) whose informativeness is reasonably high. Each point corresponds to some assessor and some query. Circle size is proportional to the number of judgments made by the assessor for that query.
Chapter 5

Learning-to-Rank in the Presence of Label Noise

We next consider the use of noisy crowdsourced assessments for training learning-to-rank algorithms (see Section 1.1.3).

Traditionally, for example within TREC efforts [54], the ground truth was obtained through very expensive expert humans looking through retrieved documents and determining relevance to given queries. Since the cost of this manual effort is prohibitive in many ways, and definitely not scalable with web search, many researchers are using alternative assessments from crowdsourcing services, like Amazon Mechanical Turk. This effort is a lot cheaper, but less reliable; it is also far more suitable to trivial yes/no questions like “Does this picture contain a tiger?” than to intellectually complex questions like “what motivates racial profiling cases in the U.S.?”

Since crowdsourcing is quickly becoming very popular due to its low cost, clearly research is needed to understand how learning-to-rank algorithms can make the most out of the noisy crowd workers assessments. Researchers have shown in the past that the noise (and sometimes the adversarial behavior) of the cheap crowd workers can be countered with variations of the Expectation Maximization algorithm [21, 48].

Most of popular ranking algorithms (e.g. RankNet/LambdaRank/LambdaMART [7], RankBoost [15], SVMRank [24]) are designed to train on a collection of documents with reliable, high quality relevance labels. These algorithms internally learn from pairwise preferences between documents, usually these preferences being deterministically derived from nominal (ordinal) labels. That is, suppose the document $d$ has nominal label (relevance grade) $g$, and the document $d'$ has nominal label $g'$; if
We can imagine scenarios when we would like to use such powerful ranking algorithms, but it is not possible to certainly prefer one document to another. This is the case with document assessments in general, when there are multiple judgments for each document available, and these judgments can exhibit some degree of contradiction. It is particularly the case when the judgments are obtained from unreliable sources and therefore noise in the labels have to be processed prior to running the learning algorithm. In Section 5.1 we first show how one can easily run the state-of-the-art ranking algorithm LambdaMART using EM-weighted average crowdsourced assessments. Then we adapt EM to infer probabilistic preferences over pair of documents and accommodate LambdaMART to using them, directly as input to the ranking algorithm.

Later, in Section 5.2 we present a new learning-to-rank method based on extension of our label aggregation method, GDA-PW, introduced earlier in Chapter 4. In the same way GDA-PW being a more powerful label aggregation method than EM, we demonstrate that our new ranking algorithm GDA-PW-MART outperforms the EM + LambdaMART combination.

### 5.1 EM + LambdaMART

Now we are going to talk about the integration of Expectation-Maximization with LambdaMART for crowdsourced assessments. In particular, we are exploring pairwise-preference assessments, where each label presented takes the form of a preference between two documents. Since the actual assessments collected are graded judgments, we use EM to obtain preference assessments as input for LambdaMART (see interface for collecting relevance assessment pictured in Figure 5.1).

Such preference assessment between two documents pairs are ideal for LambdaMART, because the internal cost function is designed on pairs (but generally used on nominal judgments because of their availability).

The trick is to run EM (see Section 4.4.3) to aggregate the assessments and compute the posterior probability distribution over true grades for each document \(d\); then, having these probability distributions separately for documents \(d\) and \(d'\), one can compute the convolution probability that \(d\) is less/more relevant than \(d'\):

\[ g < g', d \text{ will be considered as being certainly less relevant than } d'. \]
FT 30 JUN 93 / Preparing China for life without Deng: Moves to preserve the legacy and the past of the ageing 'emperor'

By TONY WALKER

FT was right, but it was a remnant nevertheless of one man's mortality and the great efforts to preserve his legacy as the moment of reckoning approaches. The official China Daily newspaper, in an article lacking the 'human' contributions to the revolution made by Mr Deng Xiaoping, referred to the ageing and ageing emperor in the past tense. Thus, Mr Deng was an outstanding representative of the industrial and its以后, referring to its continuing efforts to restore the historical record of the ancient knowledge that challenges he faced. In the process of defending Mr Deng's theories, Mr Deng has become a great man of the party and country. It is perhaps the saddest news in the history of socialists with Chinese characteristics. Ever since Mr Deng's clear plans designed to avoid conflict with social autonomy, we reviewed how the Communist party's constitution at its Ninth Congress in October 1997. However, efforts to preserve Mr Deng's theories have been increasing to try to ensure that the Deng legacy will endure and his best advice continues in a difficult place. The Communist party continues its famous challenges, and perhaps its first to be very evident, increasing Chinese attention to a smaller economy. In the process central control is being reduced in almost every way. In the past few weeks efforts to promote Mr Deng's theories have been感受 through a climate with a large flow of Western information in Shanghai, accompanied by Western media coverage. Final results are also well anticipated of theSelected Works of Deng Xiaoping 1997). The selected works revised 1997 was published in 1998. All this suggests that not only did Mr Deng and his supporters emphasize their version of history as the party and country while the earlier revision of his history, it may also indicate that his health is indeed deteriorating more quickly than has been admitted, although...

---

**Figure 5.1:** Graded assessment interface with topic keywords and description.

**Figure 5.2:** EM + LambdaMART, a two-stage approach of training a ranker in the noisy environment: (1) Label Aggregation with EM as a filter that refines noisy assessments and passes clean output to (2) LambdaMART which is trained on pairs of documents using convolution of probability distributions over true grades returned by EM.

\[ p(d < d'), p(d' < d) \]

and use them as target probabilities \( \hat{P}_{dd'} \) and \( \hat{P}_{d'd} \) for LambdaMART (see Figure 5.2).

### 5.1.1 Experiment Setup, EM Processing

We run experiments over two datasets. In each of them, the “gold” labels are TREC assessments obtained from expert annotators, present over both training and test-
ing sets. For the training sets we have an alternative set of labels, from non-expert annotators. The overall goal is to train the learning-to-rank algorithm with these non-expert labels.

The adhoc TREC 8 track has 50 queries and a QREL of about 1800 binary labeled documents. Out of these, for 10 queries crowdsourced assessments were collected: a total of 18260 documents were each assessed 5 times. These 10 queries form our training set; the other 40 queries form the testing set. For the Enterprise 2007 track, 33 queries have a total of about 3K documents with “Gold” expert assessments. For each document we have collected about 5 additional crowdsourced assessments, on average. Other 17 queries with expert judgments constitute the testing set.

**EM over raw MTurk assessments**

Because crowdsourced assessments are very likely to be noisy, we employ the EM algorithm for a better estimation of the true relevance grade. One of the noted advantages [21] of EM in this setup is that if worker quality is modeled as a hidden random variable, the overall relevance estimated is a weighted average of the worker assessments were “bad” workers are weighted out. In our model, the observed judgments from crowdsource workers are the input; worker qualities are unknown model parameters to be estimated, the document “true” grades are the latent variables.

We use EM on multigrade judgments assuming each document has 5 natural relevance grades ranging from 0 to 4, least relevant to most relevant, and each labeler is associated with a 5 x 5 confusion matrix.

**Pair probabilities as EM output**

Abusing notation for simplicity, given documents A and B, we will also call A and B the random variables corresponding to their true relevance grades as given by EM output. Recall that EM provides us with the probability distribution over relevance grades for each document. Having these probability distributions separately for documents A and B, one can easily compute the probability that A is less relevant than B (Prob[A < B]), as well as the convolution probability that A is more relevant than B (Prob[B < A]); they don’t necessarily or usually sum to one, so there is some probability for equality(Prob[B = A]).

---

We would like to thank Jie Wu who ran EM algorithm for the experiments in this section.
Figure 5.3: Convolution of relevance grades distribution for documents A and B. \( P[B < A] \) is computed as the integral of this function.

We can compute the convolution probability \( P(B < A) \) as

\[
Pr[B < A] = \sum_x Pr[A = x] \cdot Pr[B < x | A = x]
\]

\[
= \sum_x Pr[A = x] \cdot Pr[B < x] = E_A[cdf_B]
\]

\[
= \sum_x Pr[A = x] \left( \sum_{y < x} Pr[B = y] \right)
\]

Figure 5.3 shows two examples of pointwise convolution computation, which integrated give \( P[B < A] \) larger on the left, smaller on the right. In our case \( x \) iterates through the possible set of labels \( \{0, 1, 2, 3, 4\} \), thus the integral becomes a sum.

5.1.2 Training on Probabilistic Pairs

Once we get probabilistic pairwise preferences for all document pairs (using EM convolution \( P[B < A] \) as labels), we use them for training a slightly modified LambdaMART ranker. In the next section we will go over the necessary alterations of the algorithm to handle such soft preferences\(^4\). Finally, we incorporate these changes into existing implementation of LambdaMART algorithm within an open-source package JForests [18].

\(^4\)It was mentioned in [7] that this kind of generalization can be done in principle, but no exact formulas were provided.
Modified LambdaMART

As described in [7], for a given query and for each pair of documents \((d, d')\), LambdaMART models the probability \(P_{dd'}\) that the document \(d\) should be ranked higher than document \(d'\) via a sigmoid function applied on the difference of the relevance scores, \(s_d - s_{d'}\), returned by the model:

\[
P_{dd'} \equiv \frac{1}{1 + e^{-(s_d - s_{d'})}}, P_{d'd} = 1 - P_{dd'}
\]

Along with these modeled probabilities of pairwise preferences \(P_{dd'}\), we have the desired or target ones: \(\bar{P}_{dd'}\) and \(\bar{P}_{d'd}\). LambdaMART assumes [7] a hard assignment of the target probabilities, based on typical integer document labels, i.e. \(\bar{P}_{dd'}\) is either 0 or 1, and \(\bar{P}_{d'd} = 1 - \bar{P}_{dd'}\), which also implies no possibility for the documents to be equally relevant. Here we show how to generalize equations of LambdaMART algorithm for the case of arbitrary assignment of \(\bar{P}_{dd'}\) and \(\bar{P}_{d'd}\). Also, the target probability that documents \((d, d')\) are equally relevant will be \(1 - \bar{P}_{d'd} - \bar{P}_{dd'}\).

Consider the cost function of LambdaMART:

\[
C = - \sum_{d < d'} \left| \Delta Z_{dd'} \right| \left( \bar{P}_{dd'} \log P_{dd'} + \bar{P}_{d'd} \log P_{d'd} \right) \tag{5.1}
\]

where \(\Delta Z_{dd'}\) is the change in the IR measure of interest (e.g. nDCG) given by swapping the rank positions of documents \(d\) and \(d'\) in the ranked list sorted by current model scores. In other words, we are interested in minimizing the average cross-entropy between target and modeled probability distributions, weighted by a list-wise component \(|\Delta Z_{dd'}|\), which directs the ranking algorithm to focus more on the pairs of documents that are more influential with respect to a particular IR metric.

The scoring function \(s_d\) for each document \(d\) is constructed in a gradient-boosted fashion, as an ensemble of regression trees, so as to minimize the overall cost function \(C\). To this end, a Newton-Raphson optimization procedure [42] is applied, where one has to compute a Newton step size, for each document, in the space of scores, using first and second derivatives of the cost function.

The derivative of the cost function with respect to the score of \(d\)th document (also called lambda-gradient):

\[
\lambda_d \equiv \frac{\partial C}{\partial s_d} = - \sum_{d': d' \neq d} \left| \Delta Z_{dd'} \right| \left( \rho_{dd'} \left( \bar{P}_{dd'} - \bar{P}_{d'd} \right) \right) \tag{5.2}
\]
where
\[
\rho_{dd'} \equiv \frac{1}{1 + e^{s_d - s_{d'}}}
\]
The second derivative of the cost function with respect to \( s_d \):
\[
\frac{\partial^2 C}{\partial s_d^2} = \sum_{d',d' \neq d} |\Delta Z_{dd'}| \rho_{dd'} (1 - \rho_{dd'}) \left( \bar{P}_{dd'} + \bar{P}_{d'd} \right)
\]  
(5.3)

The Newton step size for the \( k \)th leaf of the \( m \)th tree, as defined in [7]:
\[
\gamma_{km} = \frac{\sum_{x_d \in R_{km}} \frac{\partial C}{\partial s_d}}{\sum_{x_d \in R_{km}} \frac{\partial^2 C}{\partial s_d^2}}
\]  
(5.4)

Thus, using (eq. 5.2 and 5.3) the Newton step size should be modified as follows:
\[
\gamma_{km} = \frac{-\sum_{x_d \in R_{km}} \sum_{d',d' \neq d} |\Delta Z_{dd'}| \rho_{dd'} (\bar{P}_{dd'} + \bar{P}_{d'd} - \bar{P}_{d'd})}{\sum_{x_d \in R_{km}} \sum_{d',d' \neq d} |\Delta Z_{dd'}| \rho_{dd'} (1 - \rho_{dd'}) \left( \bar{P}_{dd'} + \bar{P}_{d'd} \right)}
\]  
(5.5)

5.1.3 Experiments

In our experiments we used two collections: (1) TREC 8, and (2) TREC Enterprise 2007. For each collection we train different LambdaMART models on the same training set but with differently picked labels, and evaluate the models on the same testing set with the same expert judgments, in terms of nDCG. As one can see from Figure 5.5, our method (e) performs very well compared to the other label aggregation methods.

5.2 Learning-to-Rank with GDA-PW-MART

In this section we extend the pairwise aggregation model, GDA-PW (see Chapter 4), into a relevance scoring model as a function of document features\(^5\). The training input consists of document features together with crowdsourced assessments (“judgments”, “grades” or “labels”). The likelihood is defined on pairs of documents, including all crowdsourced assessments and estimated parameters of user models and document difficulty. A likelihood optimization procedure gives the relevance scoring function, in a form of gradient-boosted ensemble of regression trees. This algorithm is a better performing alternative to the combination of EM + LambdaMART discussed in the previous section.

\(^5\)This section is based on our paper [35].
Figure 5.4: LambdaMART performance on the testing set of TREC 8 collection (left) and TREC Enterprise 2007 collection (right): (a) trained on the judgments of expert assessors; (b) trained on single crowdsourced judgment for each document; (c) trained on multiple crowdsourced judgments, averaged and rounded to the closest integer; (d) trained on the most likely grade returned by EM run over multiple crowdsourced judgements; (e) modification of LambdaMART trained on probabilistic pairwise preferences from probability distributions returned by EM.

5.2.1 Learning-to-rank by extending GDA-PW

Here we present how to adapt our label aggregation method GDA-PW (eq. 4.8) for learning-to-rank purposes by embedding it into gradient boosting framework [16]. Actually, the new method GDA-PW-MART (see Figure 5.6) is very similar to previously described LambdaMART with the difference it uses a cost function similar to GDA-PW:

\[
C = -\sum_{(g,d,a),(g',d',a')\in OBS} \frac{1}{|\Delta Z_{dd'}|} \log p(g' | g, s_d, s_{d'}, \epsilon_d^*, \epsilon_{d'}^*, \theta_a^*, \theta_{a'}^*)
\]  

(5.6)

where \(\epsilon_d^*, \epsilon_{d'}^*, \theta_a^*, \theta_{a'}^*\) are the estimates of document difficulty and assessor model parameters obtained as output of GDA-PW (eq. 4.8). The components of the cost function:

\[
p(g' | g, s_d, s_{d'}, \epsilon_d, \epsilon_{d'}, \theta_a, \theta_{a'}) = \frac{1}{Z_{aa'd'd'}} \pi_g^a N(s_d - s_{d'} | \mu_g^a - \mu_g'^{a'}, \sqrt{w_g^a w_g'^{a'} \epsilon_d \epsilon_{d'}})
\]  

(5.7)

where \(Z_{aa'd'd'}\) is the normalization constant, and \(s_d = F(x_d)\) is the scoring function for document \(d\) that appears instead of modeled true relevance \(r_d\) in (eq. 4.7). Intuitively, we would like to get, for each document \(d\), a relevance score \(s_d\) as close
Figure 5.5: Comparison of various methods for learning-to-rank from crowdsourced assessments: (a) GDA-PW-MART; (b) modification of LambdaMART trained on probabilistic pairwise preferences from probability distributions returned by EM; (c) LambdaMART trained on the most likely grade returned by EM over multiple crowdsourced judgements; (d) LambdaMART trained on multiple crowdsourced judgments, averaged and rounded to closest integer; (e) LambdaMART trained on single (randomly picked) crowdsourced judgment for each document. Reported is performance (nDCG) on the testing sets of all four TREC collections. GDA-PW-MART outperforms other ranking algorithms.
as possible to the estimate of the true relevance \( r_d^* \) that we would have gotten by optimizing for (eq. 4.8). However, the exact equality is rarely achievable, as the relevance scores are constrained by the structure of the scoring function \( F(x) \) (namely, by parameters of individual regression trees \( F \) is composed of). As such, one can think of minimizing (eq. 5.6) as a constrained version of (eq. 4.8), where the flexibility of \( s_d \) (how much it can deviate from its optimal value, \( r_d^* \)) is determined by the confidence we have about \( r_d^* \), being implicitly encoded in (eq. 4.8). For implementation, we rely on the existing LambdaMART framework (we build upon open source implementation, JForests [18]) to perform optimization for the new cost function (eq. 5.6).

5.2.2 Results

In our experiments we compare the performance of our learning-to-rank method, GDA-PW-MART, with different LambdaMART models trained on the same training set but with differently aggregated crowdsourced labels. We evaluate the models on the same testing set (see Table 4.1, bottom) with the same expert judgments, in terms of nDCG. As one can easily see from Figure 5.5, our method largely outperforms the rest of LambdaMART baselines trained using various label aggregation.
methods.

The better performance of GDA-PW-MART, as compared to LambdaMART, we explain with the ability to handle uncertainties by the former algorithm. To have more intuition, consider the following example. Suppose we have four documents: \( d_1, d_2 \) for some query, and \( d_3, d_4 \) for another query, along with multiple assessments of their relevance. Suppose that after label aggregation phase, we obtain estimates of document relevance (along with standard deviations), let's say \( r_{d_1} = 0.0 \pm 1.0, r_{d_2} = 0.0 \pm 1.0, r_{d_3} = 0.5 \pm 0.1 \) and \( r_{d_4} = 0.5 \pm 0.1 \). In this example, we are much more confident about relevance of \( d_3 \) and \( d_4 \), while having a very little information (and high uncertainty) about relevance of \( d_1 \) and \( d_2 \). Since we can’t prefer \( d_1 \) to \( d_2 \), nor can we prefer \( d_3 \) to \( d_4 \), all the target probabilities for LambdaMART will be equal: \( \bar{P}_{d_1 d_2} = \bar{P}_{d_2 d_1} = \bar{P}_{d_3 d_4} = \bar{P}_{d_4 d_3} = 0.5 \). As a result, LambdaMART will “push” equally hard on all four documents with the intention to equate scores \( s_{d_1} = s_{d_2} \) and \( s_{d_3} = s_{d_4} \), despite a very little evidence of \( d_1 \) and \( d_2 \) being equally relevant in reality. Unlike LambdaMART, our algorithm GDA-PW-MART will handle this problem by focusing on equating the scores \( s_{d_3} = s_{d_4} \), while leaving \( s_{d_1} \) and \( s_{d_2} \) loose.
Chapter 6

Tighter Integration of Label Aggregation with Learning-to-Rank

Reliable estimates of document relevance with respect to a user query are crucial for the training and evaluation of search engines. Earlier we discussed methods for improving the reliability of such estimates by acquiring multiple assessments from human assessors, potentially of varying quality. These methods jointly estimate both document relevances and assessor qualities.

There often exist additional signals of relevance—typically machine generated ones—that are represented in the form of query-document features or efficient combinations of these features already pre-trained via learning-to-rank. In this chapter we show how to combine these additional signals, or virtual assessments, with real human assessments within a new label aggregation framework. As a result, we have a better estimate of document relevances (and assessor qualities) which also leads to training a better ranking model. Optionally, that better ranking model could be used to generate more accurate virtual assessments for a new round of label aggregation and subsequent learning-to-rank: in fact the whole procedure could be repeated multiple times. Our method works even in the extreme case of only one relevance assessment per document, a case inaccessible to existing relevance-aggregation frameworks because they require several assessments as input.

Further in this chapter we will extend the pairwise aggregation (from Chapter 4) and learning-to-rank (from Section 5.2) models into a relevance scoring model as a function of document features. A better estimate or relevance implies a more
reliable evaluation and a better ranking model (or less human effort needed to achieve the same accuracy). The training input consists of document features together with all assessments, both crowdsourced ("judgments") and virtual (prediction outcomes). The likelihood is defined on pairs of documents, including all crowdsourced assessments and estimated parameters of user models and document difficulty. A likelihood optimization procedure gives the relevance scoring function, in the form of gradient-boosted ensemble of regression trees.

Figure 6.1: The virtual assessor concept for learning-to-rank: learning outcome are aggregated with the human assessments. Marked in red are our contributions in this chapter.

We call learning-to-rank outcomes "virtual assessments", because they are like human assessments, only produced by an algorithm. Our main hypothesis is this: since documents have features correlated with relevance, and learning-to-rank maps the features into outcome virtual assessments which are conditionally independent with human assessments, it must be that the virtual assessments have additional information about relevance. Thus we are able to combine them within the same probabilistic framework with the goal of improving the inference of various parameters, including true relevance, assessor qualities, biases, etc. We believe it is possible to reuse a previous ranking model to generate conditionally independent virtual assessments for each document: we need to have a ranking model trained on an independent subset of queries and applied to the current query-document pair, and this could be accomplished in cross-validated manner.

Our regularization method works by assuming generalization properties of the classifier that we are training (i.e. its performance on independent dataset) and casting the learning problem into a label aggregation framework. For each training data point we aggregate corresponding label with the prediction made by another
instance of the classifier that was pre-trained on a different subset of training set not including current point. This trick ensures that the predicted score is independent of the observed label and helps in recovering the true label avoiding trivial solutions.

We propose GDA-PW-VA (see Figure 6.2), an extension to our previous label-aggregation method GDA-PW: augment probabilistic model with component that relates the observed virtual assessments and true relevances (see 6.2), governed by parameter $\omega$ which relates the difference in virtual assessments (for a pair of docs) to the difference in true relevances; $\omega$ is assumed to be global parameter and could be picked through cross-validation. All nice properties of GDA-PW stand; similar to informativeness of assessors, we add computation for informativeness of virtual assessments as well (one could now compare the informativeness of virtual assessors vs real human assessments). Previous approaches try to simultaneously infer both label noise model (i.e. assessor model) and ranking model for linear classifiers. That won’t work for Gradient Boosted Regression Trees (GBRT) with its high capacity to learn virtually any noise, so we have to deal with generalization error (measured approximately by $\omega$), not the error on training set (again, these two errors are very similar for linear classifiers, not GBRT).

An added benefit is that in this new GDA-PW-VA framework even one human assessment per document would suffice to conduct inference when it is aggregated with virtual assessments. We demonstrate this ability on the Yandex dataset; it is important because many other existing datasets come with only one human assessment per (doc, query). Similarly, we propose extension for learning-to-rank GDA-PW-MART [35], GDA-PW-VA-MART that could take VAs together with human assessments and train a learning pairwise model.

Empirically we show improvement of label aggregation for crowdsourcing dataset (Trec2010 Relevance Feedback Track) and of learning-to-rank (second iteration over first). We show that the framework is appropriate to model all assessors, including experts (Yandex assessors), implying realistic assessor qualities measured by correlation with independent Yandex qualification scores, and a better learning-to-rank performance as we iterate the process (outcomes of learning are the new virtual assessors added to the assessments pool, repeat learning for new outcomes, etc. (figure 6.1).

This chapter details the new methods which include the virtual assessments for aggregation (Section 6.3) and learning-to-rank (Section 6.4) together with main performance results. Section 6.5 explores the quality of the virtual assessments, and the ability to aggregate using virtual assessments as a function of the number
Figure 6.2: GDA-PW-VA-MART, a looped two-stage approach of training a ranker in the noisy environment: (1) Label Aggregation with GDA-PW-VA refines noisy human assessments using additional information present in the feedback from a ranker ("virtual assessments"), and (2) learning-to-rank with GDA-PW-VA-MART which is a straightforward extension of GDA-PW-VA; it produces a better performing search engine. This approach requires as few as only one relevance assessment per document.

6.1 Related work: learning in the presence of label noise

Label noise is a common problem for many machine learning applications. Most of popular machine learning algorithms don’t handle this problem directly and assume all the labels are correct: the hope is that the noise level is low and therefore the trained classifier is not going to degrade too much. However, it has been shown that it could be helpful to address this problem specifically: examples include algorithms dealing with classification [29,43], regression [43], ordinal regression [25] and pairwise learning-to-rank [19] problems. The general intuition is that during training phase, a classifier can try to identify potentially erroneous data instances and recover them by applying knowledge acquired by learning from other data instances. All these methods consider a probabilistic framework with assumption of underlying unobservable true label $r$. The overall likelihood of the data given model parameters is decomposed into two factors: (1) probability of having particular assignment of true label $r$ given feature vector $x$ and generative model param-
eters $w$: $p(r|x, w)$; and (2) probability of observable label $l$ given true label $r$ and noise model parameter $\theta$: $p(l|r, \theta)$. Then the inference is conducted to optimize for parameters $w$ and $\theta$ jointly, by maximizing overall likelihood $p(y|x, w, \theta)$ while considering and marginalizing out hidden true labels $r$. The learned generative model parametrized with $w$ (e.g. regression weights) could be applied later to predict labels on previously unseen data. Despite demonstrating improvement over baselines, however, these aforementioned implementations may be limited in terms of classifier capacity and power: all of them deal with linear models $p(r|x, w)$ and take advantage of their good generalizability. On the other hand, it would be difficult to extend these methods to work in combination with very flexible and powerful classifiers like Gradient Boosted Regression Trees [16] that deal with arbitrarily large models described by virtually unlimited number of parameters. Unless we take special measures to regularize them somehow, these classifiers will overfit to the noisy data instances and return a trivial noise model $\theta$.

Another example of noise-aware learning-to-rank [20] demonstrates the utility of accounting for assessor mistakes summarized in terms of confusion matrix: essentially each data point label is substituted with a combination of all possible labels weighted by the probability of confusing observed label with any other.

Somewhat related are measurement error models appearing in the statistics literature [17]. These are regression models that account for measurement errors in the independent variables, or regressors (usually continuous). In our case, however, the potential error is in the dependent discrete variable (target label) that we account for within our assessor-document latent trait model.

### 6.2 Experiment setup, data, assessments

We run experiments over two datasets with substantially different characteristics: (1) a combination of publicly available TREC 2010 Relevance Feedback track and Million Query 2009 sharing the same corpus of documents ClueWeb09, and (2) a proprietary large-scale learning-to-rank collection provided by Yandex search engine.

For the former collection, we perform label aggregation task and train learning to rank algorithms on queries from TREC 2010 Relevance Feedback (see Table 6.1, top for statistics), while using the subset of the rest of queries from Million Query 2009 to test the performance of trained rankers (see Table 6.1, bottom). The “gold” labels are TREC assessments obtained from expert annotators, present over both training and testing sets. For TREC 2010 Relevance Feedback collection

---

1We are grateful to Vitali Pimenov for putting together this dataset.
Table 6.1: Top table: datasets for label aggregation and training rankers. Bottom table: datasets for testing learning-to-rank trained models.

<table>
<thead>
<tr>
<th>LABEL AGGREG</th>
<th>RF10</th>
<th>Yandex</th>
</tr>
</thead>
<tbody>
<tr>
<td>(crowd or expert assessments)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Queries</td>
<td>100</td>
<td>4571</td>
</tr>
<tr>
<td>Rel grades, expert</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Rel grades, crowd</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Documents total</td>
<td>20.0K</td>
<td>79.7K</td>
</tr>
<tr>
<td>Doc per query mean</td>
<td>200</td>
<td>17</td>
</tr>
<tr>
<td>Doc per query med</td>
<td>199</td>
<td>16</td>
</tr>
<tr>
<td>Assessors</td>
<td>764</td>
<td>479</td>
</tr>
<tr>
<td>Assessments, total</td>
<td>98.3K</td>
<td>79.7K</td>
</tr>
<tr>
<td>Assess/worker mean</td>
<td>129</td>
<td>206</td>
</tr>
<tr>
<td>Assess/wkr med</td>
<td>18</td>
<td>14</td>
</tr>
<tr>
<td>Assess/wkr/query mean</td>
<td>16</td>
<td>3.2</td>
</tr>
<tr>
<td>Assess/wkr/query med</td>
<td>5</td>
<td>2.0</td>
</tr>
<tr>
<td>Assess/doc mean</td>
<td>4.9</td>
<td>1</td>
</tr>
<tr>
<td>Assess/doc med</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Queries/worker mean</td>
<td>7.9</td>
<td>63</td>
</tr>
<tr>
<td>Queries/worker med</td>
<td>3</td>
<td>7</td>
</tr>
</tbody>
</table>

| TESTING LTR MODEL      | MQ09| RF10   |
| (expert assessments)   |     | Yandex |
| Queries               | 584 | 1116   |
| Documents, total      | 15.5K| 18.8K  |

we reused existing crowdsourced judgments collected previously in [8]. Also, we extracted set of features (BM25, language models and web-graph properties) for query-document pairs to be used for learning-to-rank. Yandex dataset is a large-scale learning-to-rank collection with comprehensive set of features comparable to publicly available MSLR10K\(^2\) and Yahoo Challenge [11] collections, but unlike those we have access to IDs of assessors who annotated Yandex collection.

The overall goal is to (1) train initial ranking model with either aggregated crowdsourced non-expert labels (TREC 2010 RF) or single expert labels (Yandex) on the training set, and then (2) show that this initial ranking model could be helpful for re-aggregation of crowdsourced labels (TREC 2010 RF) and re-training learning-to-rank model (both datasets).

\(^2\)http://research.microsoft.com/en-us/projects/mslr/
6.3 Aggregation of Human & Virtual Assessments: GDA-PW-VA

We are now presenting the new framework for label aggregation (GDA-PW-VA) and learning-to-rank (GDA-PW-VA-MART) that can take not only human assessments, but also additional continuous signals, i.e. virtual assessments.

Given a pair of observations from a set of virtual assessments \((v, d), (v', d') \in VIRT\) and denoting \(\Delta_v = v - v', \Delta_r = r_d - r_{d'}\), we model the likelihood of these observations by:

\[
p(\Delta_v | \Delta_r, \omega) = p_0(\Delta_v) \frac{\mathcal{N}(\Delta_v | \Delta_r, \omega)}{Z(\Delta_r, \omega)} \tag{6.1}
\]

where \(Z(\Delta_r, \omega)\) is a normalizing factor. Parameter \(\omega\) could be thought of as a measure of correlation between \(\Delta_r\) and \(\Delta_v\): small \(\omega\) corresponds to high correlation (virtual assessments are informative and could be helpful for inferring unknown true relevance), big \(\omega\) corresponds to weak correlation (virtual assessments are not that useful). If we assume prior \(p_0(\Delta_v)\) to be Normal with zero mean and variance\(^3\) of 2.0 and using formula for computing a product of two Gaussian pdfs then (eq. 6.1) transforms into:

\[
p(\Delta_v | \Delta_r, \omega) = \mathcal{N}(\Delta_v | \frac{2\Delta_r}{\omega + 2}, \frac{2\omega}{\omega + 2}) \tag{6.2}
\]

Denoting by \(K\) the overall number of judgments and \(N\) the overall number of documents, we define the global log likelihood of GDA-PW-VA by extending (4.8):

\[
\log L = \sum_{(g,d,a),(g',d',a') \in OBS} (K)^{-1} \log p(g | g', \theta_d, \theta_{d'}, \theta_a, \theta_{a'}) + \sum_{(v,d),(v',d') \in VIRT} (N)^{-1} \log p(\Delta_v | \Delta_r, \omega) \tag{6.3}
\]

It might be useful to estimate informativeness of virtual assessments as well – so that we could compare it with performance of human assessors on the same scale (units of information). First, we write the posterior of the difference of relevances

---

\(^3\)Prior to running our algorithms we normalize virtual assessments \(v\) to have zero mean and unit variance.
for a pair of documents:

\[
p(\Delta_r|\Delta_v, \omega) = p_0(\Delta_r) \frac{\mathcal{N}(\Delta_r|\Delta_v, \omega)}{Z(\Delta_v, \omega)}
\]

\[
= p_0(\Delta_r) \frac{\mathcal{N}(\Delta_r|\Delta_v \frac{2\omega}{\omega + 2}, \frac{2\omega}{\omega + 2})}{Z(\Delta_v, \omega)}
\]

\[
= p_0(\Delta_r) \frac{\mathcal{N}(\Delta_r|\Delta_v \frac{\omega + 2}{2}, \omega)}{Z(\Delta_v, \omega)}
\]

\[
\propto \mathcal{N}(\Delta_r|0, 2) \cdot \mathcal{N}(\Delta_r|\Delta_v \frac{\omega + 2}{2}, \omega)
\]

\[
\propto \mathcal{N}(\Delta_r|\Delta_v, \frac{2\omega}{\omega + 2})
\]

where the last transition was made using formula for computing a product of two Gaussian pdfs.

Informativeness of a pair of virtual assessments becomes the expected reduction of uncertainty about \(\Delta_r\):

\[
I_{\Delta_v}(\omega) = H[p_0(\Delta_r)] - H[p(\Delta_r|\Delta_v, \omega)]
\]

\[
= H[\mathcal{N}(\Delta_r|0, 2)] - H\left[\mathcal{N}(\Delta_r|\Delta_v, \frac{2\omega}{\omega + 2})\right]
\]

\[
= \frac{1}{2} \ln(4\pi e) - \frac{1}{2} \ln\left(2\pi e \frac{2\omega}{\omega + 2}\right) = \frac{1}{2} \ln\left(\frac{\omega}{\omega + 2}\right)
\]

where we applied formula for differential entropy of a normal distribution. One can see that informativeness of a pair of virtual assessments depends only on \(\omega\). Again, we can report informativeness of a single virtual assessment as being half of that value:

\[
I(\omega) = -\frac{1}{4} \ln\left(\frac{\omega}{\omega + 2}\right) \tag{6.4}
\]

6.3.1 Label Aggregation Results

In this experiment we aggregate crowdsourced labels for TREC 2010 RF dataset from Table 4.1, then rank the documents for each query by true relevance estimated based on these aggregated labels, and then evaluate these rankings by using expert labels, in terms of three common IR measures: MAP, nDCG and Kendall \(\tau\). The task here was to compare the performance of proposed GDA-PW-VA method – which makes use of additional information from query-document features – with performance of other label aggregation baselines: (a) randomly ordered documents which doesn’t rely on training labels at all, (b) majority vote, by ordering doc-
uments by the most frequent label (or one of them if there are several), (c) ordering documents by straight average of crowdsourced labels, (d) expectation-maximization method, (e) Polytomous Rasch model, and also previously proposed (f) GDA and (g) GDA-PW (refer to Chapter 4 for more details on baselines). To produce virtual assessments we train four different GDA-PW-MART rankers using crowdsourced labels: each ranker is trained on three out four partitions of the training set (there are four different ways of choosing three partitions out of four). The idea is for each query to have an independent ranker that was trained on a subset of queries not including current one, and therefore if we apply that ranker on current query, the produced document scores would be (conditionally) independent from the labels of these documents. Then we apply GDA-PW-VA to aggregate both crowdsourced labels and ranking scores, i.e. virtual assessments, together within same framework. Evaluating the quality of aggregation using independent expert assessments clearly shows (Table 6.2) that GDA-PW-VA outperforms the baselines by all measures\(^4\) and demonstrates the capacity of making use of additional information about relevance stored in the features.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP</th>
<th>nDCG</th>
<th>K(\tau)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.553</td>
<td>0.669</td>
<td>0.000</td>
</tr>
<tr>
<td>MV</td>
<td>0.678</td>
<td>0.751</td>
<td>0.295</td>
</tr>
<tr>
<td>Average</td>
<td>0.720</td>
<td>0.777</td>
<td>0.382</td>
</tr>
<tr>
<td>EM</td>
<td>0.767</td>
<td>0.810</td>
<td>0.446</td>
</tr>
<tr>
<td>Rasch</td>
<td>0.762</td>
<td>0.809</td>
<td>0.449</td>
</tr>
<tr>
<td>GDA</td>
<td>0.770</td>
<td>0.815</td>
<td>0.467</td>
</tr>
<tr>
<td>GDA-PW((\epsilon = 1))</td>
<td>0.772</td>
<td>0.819</td>
<td>0.484</td>
</tr>
<tr>
<td>GDA-PW</td>
<td>0.776</td>
<td>0.821</td>
<td>0.484</td>
</tr>
<tr>
<td>GDA-PW-VA (1st iter)</td>
<td><strong>0.794</strong></td>
<td><strong>0.832</strong></td>
<td><strong>0.536</strong></td>
</tr>
</tbody>
</table>

Table 6.2: Performance of various methods for aggregation of crowdsourced assessments on TREC 2010 RF dataset. The highest value in each column is highlighted. All the rest values are significantly lower (p < 0.05) than those of GDA-PW-VA.

\(^4\)We test for significance of improvement using Fisher’s two-sided paired randomization test [50].
6.4 Learning with Human & Virtual Assessments: GDA-PW-VA-MART

Similarly, we generalize cost function of GDA-PW-MART (eq. 5.6) to include virtual assessments:

\[
C = -|\Delta Z_{dd'}| \left( \sum_{(g,d,a),(g',d',a') \in OBS} \frac{1}{K} \log p(g|g', s_d, s_{d'}, \epsilon_d^*, \epsilon_{d'}^*, \theta_a^*, \theta_{a'}^*) \right) + \sum_{(v,d),(v',d') \in VRT} \frac{1}{N} \log p(\Delta_v|s_d - s_{d'}, \omega) \right) \tag{6.5}
\]

\[
\sum_{(v,d),(v',d') \in VRT} \frac{1}{N} \log p(\Delta_v|s_d - s_{d'}, \omega) \right) \tag{6.6}
\]

Again, we build upon existing LambdaMART framework [18] to perform optimization for the new cost function (eq. 6.6).

6.4.1 Learning-to-Rank Results

In our experiments we compare the performance of our learning-to-rank method, GDA-PW-VA-MART, with the baselines: GDA-PW-MART (TREC 10 RF) and LambdaMART (Yandex). We train competing algorithms on the same training sets (see Table 4.1, top) and evaluate them on the same testing sets (see Table 4.1, bottom).

For TREC 10 RF dataset, we first train a GDA-PW-MART ranker on crowdsourced labels of full training set, and also four additional GDA-PW-MART rankers each on 3/4 of training set: these additional rankers are used to produce virtual assessments for the initial iteration of GDA-PW-VA-MART. As the process of re-training GDA-PW-VA-MART could be repeated as many times as we want, for each subsequent iteration of GDA-PW-VA-MART we’ll use output of GDA-PW-VA-MART rankers trained on previous iteration.

As one can see from Figure 6.3, GDA-PW-VA-MART method outperforms the GDA-PW-MART baseline. Better performance of GDA-PW-MART, as com-

Figure 6.3: Learning-to-rank performance on TREC 2010 RF dataset: comparing GDA-PW-MART with few iterations of GDA-PW-VA-MART. GDA-PW-MART model serves as virtual assessor for GDA-PW-VA-MART (iter 1), and each iteration of GDA-PW-VA-MART serves as virtual assessor for subsequent iteration of GDA-PW-VA-MART.
pared to GDA-PW-MART, we explain with a better ability of the former algorithm to withstand overfitting to noisy examples due to inherent self-regularization properties. This effect could be better seen at later rounds of boosting when performance of the baseline GDA-PW-MART starts degrading rapidly because of overfitting.

For Yandex dataset, we first train a LambdaMART ranker using expert labels of full training set, and also four additional LambdaMART rankers each on 3/4 of training set: similarly, these additional rankers are later used to produce virtual assessments for GDA-PW-VA-MART model. From Figure 6.4 it could be seen that GDA-PW-VA-MART notably outperforms standard LambdaMART algorithm. Speaking about GDA-PW-VA-MART, it is also remarkable that we can meaningfully apply an algorithm designed for crowdsourcing, in the situation when we have only single relevance assessments per document – no other crowdsourcing algorithms would allow that. And even in this extreme case, GDA-PW-VA-MART is able to efficiently determine variable qualities of expert assessors and exploit this information to train a better ranking model. To support the latter statement about accurate inference of assessor qualities, we compare inferred qualities against an independent measure of assessor performance: within Yandex efforts of collecting relevance assessments, input from each assessor was verified by a special super-assessor on several randomly chosen queries. Based on this information, each assessor was assigned an overall qualification score. In Figure 6.5 we plot that independent qualification score against assessor informativeness based on the assessor model inferred while training GDA-PW-VA-MART. One can observe high correlation between these two measures.

6.5 Analysis of Virtual Assessor

6.5.1 Aggregation quality as function of number of assessments

When collecting crowdsourced relevance judgments for a document, we not only can estimate its true relevance (assuming we already have some estimates of the qualities of assessors who contributed judgments), but, more generally, we also can compute the posterior probability distribution of the true relevance. Having posterior distribution can be very useful, for instance, to compute confidence intervals for the true relevance. If too wide confidence intervals are observed, then we may want to collect additional assessments for that document to have more reliable estimates of relevance. Here we would be interested in comparing GDA-PW with GDA-PW-VA in terms of reliability of relevance estimates they provide. For both methods the posterior doesn’t seem to have a closed form expression. Therefore we
In Figure 6.4: Learning-to-rank performance on Yandex dataset: comparing LambdaMART with GDA-PW-VA-MART (iter 1). LambdaMART model serves as virtual assessor for GDA-PW-VA-MART.

In Figure 6.5: Analyzing assessor models for Yandex dataset: comparing inferred informativeness of each assessor with their independent qualification score.

In Figure 6.6: Simulation of online experiment for 3 documents of TREC 2010 RF collection (query 20986): GDA-PW with only crowdsourced assessments (on the left), and GDA-PW-VA including virtual assessments (on the right). Relevance judgments are collected one by one, each time improving the knowledge about true relevance. For each document, the true relevance estimates are plotted along with 50% confidence intervals.

rly on the Laplace approximation for the posterior (refer to [5]), by computing the second derivatives of likelihood functions with respect to the model parameters.

In Figure 6.6 we plot the results of simulating online experiment when we collect assessments for a few documents with different relevance (as per expert assessor), one by one. While seeing how our knowledge about relevance improves over time, we notice that GDA-PW-VA starts off with already pretty good initial
estimate of relevance provided by virtual assessments based on the information encoded in features. Thus it is clear that on average we need less human-produced assessment to achieve the same level of reliability as compared to the case when we have no virtual assessor.

### 6.5.2 Analysis: Virtual Assessor Informativeness

Next, we measure the informativeness of each assessor quantitatively, computing it using (eq. 4.11) for human assessors and (eq. 6.4) for a virtual assessor. We demonstrate that the inferred informativeness of each assessor makes sense by plotting it against the measure of how well the judgments of that assessor are consistent with ones of the expert, in terms of Kendall $\tau$ coefficient for rank correlation. By looking at Figure 6.7, we observe that there is indeed high correlation between two plotted quantities (it is particularly true for the cases when we have many judgments made by an assessor). Moreover, we observe almost zero inferred informativeness for random virtual workers and a considerably high informativeness for expert assessors. Also it should be noted that virtual assessors have a comparatively high informativeness. That emphasizes once again the importance of using additional information stored in the features when we need to obtain reliable document relevance estimates while trying to minimize burden of annotation imposed on human assessors.

Figure 6.7: Measured informativeness of assessor vs. Kendall $\tau$ coefficient for rank correlation between the assessor and an expert. Each point corresponds to some assessor and some query (we show all the queries for the same collection on the same plot). Circle size is proportional to the number of judgments made by the assessor for that query.
Chapter 7

Conclusion

In this dissertation, we identified and approached several problems emerging on the edge between Information Retrieval and Machine Learning. All these problems stem from a common cause: the unreliability of human-generated relevance assessments needed to perform two very basic tasks of IR, namely (1) evaluation of existing retrieval systems, or search engines, and (2) building the new ones. Motivated by the practical importance of the problems under examination (it is hard to overestimate the ever increasing role search engines play in everyday life) and applying a wide range of scientific methods, we analyzed and quantified the adverse effect of unreliable assessments and also developed new tools to confront this unreliability. These tools can be directly applied to existing retrieval systems, from general-purpose search engines to those tailored for a specific task (e.g., medical, legal, corporate retrieval, etc.), with the ultimate goal of improving the efficiency of such systems and leading to a better overall user experience.

Along with immediate practical application of our tools as is, one can use them as a basis for building more sophisticated instruments. For example, our label aggregation framework, owing to its fully probabilistic nature and therefore potential for information-theoretic analysis, could easily be complemented with an active learning component: where the training collection is constructed in a way that its utility (informativeness) is maximized while the construction costs are minimized (also potentially employing a dynamic reward system, when assessors are paid proportionally to their usefulness). For the evaluation of retrieval systems with unreliable relevance assessments, one could translate relevance uncertainties computed within our framework for individual documents into uncertainties of IR measures (e.g., Average Precision, nDCG, ERR, etc.): by setting some tolerable level for uncertainty of a particular measure of interest, it would be clear when there is
a need for additional assessments.

Furthermore, the methods we propose in Chapters 4-6 are not limited solely to Information Retrieval, but also apply to a broader class of problems dealing with subjective ordinal assessments (e.g., movie ratings, severity of symptoms, etc.) and when learning from these assessments is needed. Besides, the idea of iterative learning with a feedback loop presented in Chapter 6 could serve as a beneficial regularization trick for a wider class of Machine Learning applications, in addition to the case of ordinal labels considered in our work.

Yet another interesting extension of our work would be in exploring *multidimensional relevance* when we assume existence of multiple factors influencing assessor decision to assign some particular grade and allowing different assessors prioritize these factors differently. Generalizing our effective method of treating subjective/biased ordinal assessments to the multidimensional case, one could be able to reveal and exploit potentially rich space of latent factors and personal traits, e.g., for recommender systems and other personalization-oriented applications.
Bibliography


77