Reducing Reliance on Relevance Judgments for System Comparison by Using Expectation-Maximization

Ning Gao,[1] William Webber[2] and Douglas W. Oard[1]

[1]College of Information Studies/UMIACS, University of Maryland, College Park
{ninggao, oard}@umd.edu
[2]William Webber Consulting, william@williamwebber.com

Abstract. Relevance judgments are often the most expensive part of information retrieval evaluation, and techniques for comparing retrieval systems using fewer relevance judgments have received significant attention in recent years. This paper proposes a novel system comparison method using an expectation-maximization algorithm. In the expectation step, real-valued pseudo-judgments are estimated from a set of system results. In the maximization step, new system weights are learned from a combination of a limited number of actual human judgments and system pseudo-judgments for the other documents. The method can work without any human judgments, and is able to improve its accuracy by incrementally adding human judgments. Experiments using TREC Ad Hoc collections demonstrate strong correlations with system rankings using pooled human judgments, and comparison with existing baselines indicates that the new method achieves the same comparison reliability with fewer human judgments.

1 Introduction

Information retrieval systems are generally evaluated on the relevance of the documents they retrieve. Relevance judgments must be made by humans, and obtaining these judgments can be the most expensive part of retrieval evaluation. The expense is greatest when building a reusable test collection, since judgments are required not only for documents returned by a particular system or set of systems, but also for those that might be returned by future systems on the same collection. The traditional method of managing the judgment task for reusable collections is through pooling, where only the top-ranked documents (such as the first 100 documents) from participating systems are judged. Even pooling can lead to a heavy judgment burden, however, with depth 100 pooling on Text Retrieval Conference (TREC) Ad Hoc collections requiring thousands of document judgments (133,681 for TREC 5, for example).

In this paper, we introduce a novel Expectation-Maximization (EM) algorithm for comparing system effectiveness. The expectation step predicts the relevance of documents based on which systems return them; then the maximization step estimates system effectiveness from the predicted relevance of the documents the system returns. This process is repeated iteratively until the algorithm converges. A particular feature of our method is that it can work with no human judgments at all, but can also incrementally incorporate human judgments to improve accuracy.

We conduct experiments comparing our EM algorithm with existing state-of-the-art methods for reduced-effort evaluation, both without and with human judgments, on four TREC Ad Hoc collections. The task is to estimate a ranking of systems by retrieval effectiveness. We find that our method generally outperforms existing judgment-free methods. Our method improves still further as human relevance judgments are added, outperforming existing limited-judgment methods for most judgment set sizes.

The remainder of the paper is laid out as follows. In Section 2, we survey related work on the reduced-effort evaluation of retrieval systems. Section 3 describes our expectation-maximization method for retrieval system ranking, along with the various methods of calculating system votes for document relevance and, where human judgments are available, of selecting documents for judgment. In Section 4, we evaluate the different voting and document selection methods against each other, and against established methods of judgment-free and limited-judgment system evaluation. Finally, Section 5 presents our conclusions.

2 Related Work

The use of system pools to select documents for relevance judgment was first proposed by Spärck-Jones and van Rijsbergen [13]. Zobel examined the reliability of pools in early TREC Ad Hoc collections, finding that although they only captured half or fewer of the relevant documents in the collection, the resulting judgment sets were not significantly biased against unpooled systems [17]. Cormack et al. proposed Move-To-Front (MTF) pooling [6]. Under MTF, the documents from each system are judged in order of rank, and the systems themselves are also prioritized according to the previously judged documents. Various approaches have been proposed for evaluation with incomplete judgments. One is to use an evaluation metric that handles missing judgments. Buckley and Voorhees introduced this idea with the BPref metric, which evaluates only by judged documents [3].

A second approach to a limited judgment budget is to sample documents for judgment, then statistically estimate the evaluation measure. Aslam et al. described Hedge, a greedy algorithm for creating pools likely to contain large numbers of relevant documents [1]. Yilmaz and Aslam introduced a uniform random sample of the pooled documents and infer the pooled Average Precision (AP) metric to create infAP [14]. Aslam et al. also proposed an unequal sampling method based upon AP weights [2]. A simpler and more general stratified-sampling approach, xinfAP, is described in Yilmaz et al. [16]. Carterette and Allan suggested that the documents to (incrementally) prioritize for judgment are those which, if found relevant, would maximize the difference in scores between evaluated systems [5]. Carterette's Robust Test Collection (RTC) treats the retrieval systems as experts, and their retrieval of documents as votes for relevance [4].

All the methods in this paper are item-based, estimating pseudo-judgments of relevance for each document, and then evaluating the systems from these pseudo-judgments. There are also distribution-based methods. In Dai et al., the distribution of scores over documents is modeled by a mixture of two distributions, a Gaussian distribution for relevant documents and an exponential distribution for non-relevant documents [7].

An alternative approach to system evaluation is to do without human relevance judgments, ranking systems based solely on the system document rankings themselves. The

first to propose this approach was Soboroff et al., who used voting based on random assignment of relevance to documents in the document pool [12]. Nuray and Can's Condorcet method used system document rankings to rank documents by predicted relevance, and then assigned binary relevance to documents up to some cutoff [11]. Hauff et al. compared all the above methods across 16 different test collections (from TREC and elsewhere), finding Soboroff et al's random-voting method best on nine collections, and Nuray and Can's Condorcet method best on six [8]. Hosseini et al. used the EM framework to solve the problem of acquiring relevance judgements for Book Search tasks through crowdsourcing when no true relevance labels are available [9].

3 System Comparison by EM

In this section, we introduce our method for performing system comparisons with limited or no human judgments. We begin with a description of our fully automatic EM method, using no human judgments at all, in Section 3.1. We then describe how we adapt the technique to make use of incomplete human judgments in Section 3.2.

3.1 Zero Relevance Judgments

To illustrate the operation of our EM algorithm, we start from the simplest case, which is when no human judgments are available. In the expectation step (E-step), real-valued pseudo-judgments are estimated using the document rankings produced by a diverse range of systems. The loss between the result vector of a system and the pseudo-judgment vector is then calculated for each system. In the maximization step (M-step), the weights of systems are tuned to minimize the loss with respect to the pseudo-judgments. The E-step and M-step alternate until the values converge to a fixed point. The final real-valued pseudo-judgments of the documents are then taken as estimates of document relevance, and used to evaluate retrieval systems, for instance by binarizing relevance at a prediction threshold and then calculating an effectiveness metric.

The process can be more precisely explained with reference to the "learning matrix" shown in Table 1. Suppose that there are p systems $\{S_1, ..., S_p\}$ to be compared using a measure that averages over k topics $\{T_1, ..., T_k\}$, and that for each topic T_m there are $t(m)$ documents $\{D_{m,1}, ..., D_{m,t(m)}\}$ over whose relevance the effectiveness measure will be computed. These documents could be the whole collection, but in our experiments they are the judgment pool that was created for TREC. Because different topics have different judgment pools, we subscript the documents separately for each topic. Suppose further that $V_{S_j,D_{m,n}}$ is the score that system S_j assigns to document n for topic m. We refer to this score as the *retrieval status value* of the respective document.

The EM algorithm iteratively estimates two sets of values: the hidden variables, which are the pseudo-judgment scores for each document on each topic; and the parameters, which are the loss-minimizing weights for each system. For each document $D_{m,n}$, a real-valued pseudo-judgment $J_{m,n}$ reflects the EM algorithm's current degree of belief that document n is relevant to topic m. Similarly, the system weight w_j represents the algorithm's present degree of belief that the results produced by system j are correct. The better a system is at contributing to the estimation of accurate pseudo-judgments,

		$\mathbf{w_1}$	...	$\mathbf{w_p}$	
		S_1	...	S_p	
T_1	$D_{1,1}$	$V_{S_1,D_{1,1}}$		$V_{S_p,D_{1,1}}$	$\mathbf{J_{1,1}}$
	...	...		...	**...**
	$D_{1,t(1)}$	$V_{S_1,D_{1,t(1)}}$		$V_{S_p,D_{1,t(1)}}$	$\mathbf{J_{1,t(1)}}$
...	...	...		...	**...**
T_k	$D_{k,1}$	$V_{S_1,D_{k,1}}$		$V_{S_p,D_{k,1}}$	$\mathbf{J_{k,1}}$
	...	...		...	**...**
	$D_{k,t(k)}$	$V_{S_1,D_{k,t(k)}}$		$V_{S_p,D_{k,t(k)}}$	$\mathbf{J_{k,t(k)}}$

Table 1. EM Learning Matrix: **S** for systems, **w** for system weights, **T** for topics, **D** for documents, **J** for pseudo-judgments, and **V** for predicted scores.

the higher its weight should be. We require that the p system weights be bounded to $[0,1]$ and that they always sum to 1, ($||w_i||_1 = 1$); thus they represent a distribution.

The iteration begins by setting the initial system weights $w^1 = \{w_1^1, ..., w_p^1\}$ to the uniform distribution, as $1/p$, and then performing the first E-step. In the t-th E-step, suppose we have system weights $w^t = \{w_1^t, ..., w_p^t\}$. We then compute the pseudo-judgment $J_{m,n}$ for each document $D_{m,n}$ as:

$$J_{m,n}^t = \sum_{j=1}^{p} w_j^t \cdot f(V_{S_j,D_{m,n}}) \; . \tag{1}$$

The score transformation function $f(\cdot)$ allows us to implement a range of estimators for the E-stage by transforming the system's retrieval status value $V_{S_j,D_{m,n}}$ prior to performing the weighted linear combination across systems. In this paper, we try three score transformation functions:

$f_{\text{Score}}(v)$ A scaled version of the system-produced document score, $f_{\text{Score}}(v) = v/v_{\max}$, where $v_{\max}$ is the maximum score assigned by system S_j to any document for topic T_m. This scaling treats the retrieval status value as being measured on an interval scale, while avoiding giving one system (or one topic) more emphasis than another at the outset.

$f_{\text{BordaCount}}(v)$ The (single-ranking) Borda count of the score, computed as $f_{\text{BordaCount}}(V_{S_j,D_{m,n}}) = R_{j,m} - r(D_{m,n})$, where $R_{j,m}$ is the number of results that system S_j returns for topic T_m, and $r(D_{m,n})$ is the rank of document $D_{m,n}$ when retrieval status values $V_{S_j,D_{m,n}}$ are sorted decreasing, breaking ties arbitrarily.

$f_{\text{Vote}}(v)$ A binarized version of the score, in which if the document $D_{m,n}$ is in the top 1000 results returned by system S_j, then $V_{S_j,D_{m,n}}$ is set to 1; otherwise, $V_{S_j,D_{m,n}}$ is set to 0. The pseudo-judgment score for document $D_{m,n}$ at iteration t is therefore the w^t-weighted average of the number of systems returning that document.

The t-th M-step is then initiated by computing the loss function L_j for each system S_j in iteration t as the square of the Euclidean distance between the (scaled) vector of system scores and the corresponding vector of pseudo-judgments from the t-th E-step:

$$L_j^t = \sum_{m \in [1,k], n \in [1,t(m)]} \left(w_j^t \cdot V_{S_j,D_{m,n}} - J_{m,n}^t \right)^2 \; . \tag{2}$$

L_j^t will be small when a system's results exhibit little (scaled) disagreement with the pseudo-judgments; to the extent that the pseudo-judgments are reasonable estimates of the actual judgments, L_j^t can be interpreted as an inverse estimate of the effectiveness of system S_j. We therefore update the system weights by first computing the *inverse loss* I_j^t of system j, by inverting the sign of the loss and then applying an additive offset to guarantee that the result is non-negative:

$$I_j^t = \sum_{m,n,j} (w_j^t \cdot V_{S_j,D_{m,n}})^2 - \sum_{m,n} \left(w_j^t \cdot V_{S_j,D_{m,n}} - J_{m,n}^t\right)^2 . \tag{3}$$

We can regard the inverse loss as an improved estimate for the relative system weight; so all that remains to get the actual system weight for the $(t+1)$-th step is to normalize the values to sum to 1:

$$w_j^{t+1} = \frac{I_j^t}{\sum_{j=1}^{p} I_j^t} . \tag{4}$$

The iteration stops when the algorithm converges to a point where the weights and pseudo-judgment scores no longer change.[1] The final pseudo-judgment vector J can then be used to estimate the relevance judgments. When binary relevance judgments are required for computation of system effectiveness measures (as in our experiments), we treat the λ_m documents with the highest pseudo-judgment values for topic m as relevant and the others as not relevant. Because we use this value only for reporting results, it has no effect on our EM iteration. In this paper, we set λ_m to the true number of relevant documents in the NIST TREC judgment pools. In our experiments, we also give λ_m to the baseline techniques against which we compare. We leave parameter estimation for λ_m (or the design of parameter-free measures for system comparison directly from pseudo-judgments) for future work.

3.2 Incomplete Relevance Judgments

The dependence of the M-step in our algorithm on the prior generation of informative pseudo-judgments in the immediately previous E-step suggests that we might be able to improve our results by substituting some actual judgments for selected pseudo-judgments. This leads naturally to two questions. If some incomplete set of actual judgments were available, how should they be used? And if those actual judgments were to be created on demand, which documents should be judged? In this section, we address these two questions in turn.

Using Human Judgments Suppose that after the E-step in the t-th iteration (but before the M-step) we obtain a human judgment for document $D_{m,n}$, denoted $H_{m,n} \in \{0,1\}$. Then we can simply set $J_{m,n}^t = H_{m,n}$. Indeed, since once we learn $H_{m,n}$ we expect that there would be no value to forgetting it, we perform the same substitution after each subsequent E-step. Moreover, because the human judgments represent ground truth, we

[1] Convergence Proof: `http://www.umiacs.umd.edu/~ninggao/publications`

can reasonably give them greater influence when we estimate system weights in the M-step. We therefore modify the computation of the loss function to implement this idea, replacing Equation (2) with:

$$L_j^t = \sum_{m\in[1,k],n\in[1,t(m)]} T_{m,n} \cdot (w_j^t \cdot f(V_{S_j,D_{m,n}}) - J_{m,n}^t)^2 \,, \tag{5}$$

where

$$T_{m,n} = \begin{cases} \gamma & \text{if human judgment} \\ 1 & \text{if pseudo-judgment .} \end{cases} \tag{6}$$

Here $T_{m,n}$ encodes the weight to place on the (human or pseudo) judgment for $D_{m,n}$. When we have a human judgment $H_{m,n}$, we set $T_{m,n}$ to $\gamma\,(\gamma > 1)$. For this paper we have arbitrarily set $\gamma = 2$, leaving the question of optimizing γ for future work. With this one small change, our EM algorithm proceeds as before.

Requesting Human Judgments In general, we can request some number N_m^t of new judgments for topic m after each E-step t (and before the subsequent M-step), selecting those judgments using selection policy P_i, where a selection policy is a rule for selecting previously unjudged documents for human judgment. In this paper we set N_m to be 1% of the available TREC relevance judgments[2] at each iteration, and we consider four selection policies, consistently selecting using the same policy at each iteration. Note that after 100 steps, all available judgments will be used, and thus we will produce the same results as reported at TREC. We then compare policies based on which most rapidly approach those TREC results. The four policies are:

P_1 Select the N_m as-yet unjudged documents with highest pseudo-judgment scores $J_{m,n}$, breaking ties arbitrarily.

P_2 For each document $D_{m,n}$ compute (across systems) the arithmetic mean $M_{m,n}$ and the standard deviation $SD_{m,n}$ for $(f(V_{S_1,D_{m,n}}),\ldots,f(V_{S_p,D_{m,n}}))$ and then select the as-yet unjudged documents for which $M_{m,n} + \beta SD_{m,n}$ is largest. Here β is a tunable parameter, arbitrarily set to 2 in our experiments.

P_3 Randomly select N_m as-yet unjudged documents by simple random sampling.

P_4 Select the documents which would maximize the weighted average system loss if they were not relevant. This is the Hedge loss function defined by Aslam et al. [1].

In our experiments, we select documents to be judged only from the TREC judgment pool, as we do not know the correct judgment for documents outside the pool. Both P_1 and P_2 reflect the fact that the distinction between positive judgments and unknown judgments is consequential for Mean Average Precision, our evaluation measure, while the distinction between negative judgments and unknown judgments is inconsequential for that measure. Our use of standard deviation in P_2 reflects our intuition that a greater diversity of system responses for the same document might be an indication of poorly estimated pseudo-judgments, and also of documents likely to be discriminative between systems.

[2] The mean of N_m across all topics and test collections is 22.5 (min 11, max 46).

4 Experiments

We start with a description of our test collections and evaluation measures in Section 4.1. We then present zero-judgment results in Section 4.2, followed by incomplete-judgment results for each judgment selection policy in Section 4.3.

4.1 Test Collections and Evaluation Measures

We use four TREC Ad Hoc test collections, from TREC 5, 6, 7 and 8. In each case, we use all topics, and all the Ad Hoc runs officially submitted to NIST that searched the full collection (known as Category A runs). For each test collection, the top 100 documents returned by each system (for TREC 5), or for selected systems (for TREC 6 through 8), form the pool for which human judgments, created by NIST, are available. For comparability with prior work, we focus on the degree to which system comparisons made on the basis of numerical differences in Mean Average Precision (MAP) reflect those which would have been made had the full TREC judgment pool been available. We use *trec_eval* version 9.0 to compute MAP truncated at 1,000 documents (*MAP@1k*) as the measure of effectiveness, treating unjudged documents as not relevant, and binarizing pseudo-judgments as described above.

Our first measure of reliability is Kendall's τ [10] between the system ranking using MAP@1k and the ranking produced by the method under analysis. When the purpose of our evaluation is to measure relatively small improvements over already-good systems, we would prefer to have a reliability metric that is more influenced by reversals among the best systems than among relatively poor ones. Kendall's τ weights both equally; the τ_{ap} [15] measure, however, places greater weight on more highly ranked systems. The τ_{ap} measure is asymmetric; we designate the full-pool TREC rankings as the objective and our EM algorithm's system ranking as the comparison.

4.2 Zero-Judgment Results

In this section, we present the reliability of our EM algorithm for system comparisons without human judgments. Various voting and learning methods are employed. Vote, Score, and BordaCount denote the results after the first E-step using f_{Vote}, f_{Score} and $f_{\text{BordaCount}}$, respectively, while Vote.EM, Score.EM, and BordaCount.EM denote the corresponding results after convergence. BordaCount is equivalent to the Borda count method of Nuray and Can [11]. Additionally, we reimplement two other previously published baselines for the zero-judgment task, the Condorcet method [11] and Soboroff's random-voting method [12].

Table 2 indicates the effect of EM, with (+) when EM numerically improves the reliability measure τ or τ_{ap} over the first E-step alone, and (-) to indicate a numerically adverse affect. Over 24 such comparisons (two measures, three score transformation functions, four test collections), 17 favored EM and the remaining 7 favored the first E-step, a significant improvement under a paired Wilcoxon test at $p < 0.05$.

Taking TREC MAP@1k as ground truth scores, Figure 1 shows the difference between the estimated and ground truth MAP@1k score. Systems are ordered by TREC MAP@1k on the X-axis with best systems on the left. For each system, the value on

	τ				τ_{ap}			
	TREC 5	TREC 6	TREC 7	TREC 8	TREC 5	TREC 6	TREC 7	TREC 8
Score	0.469	0.551	0.473	0.471	**0.337**	0.289	0.227	0.156
Score.EM	**0.479** (+)	**0.563** (+)	0.496 (+)	0.490 (+)	0.326 (-)	**0.314** (+)	**0.257** (+)	0.169 (+)
BordaCount	0.464	0.537	0.471	0.462	0.322	0.276	0.229	0.147
BordaCount.EM	0.453(-)	0.555 (+)	0.487 (+)	0.461 (-)	0.321 (-)	0.294 (+)	0.237 (+)	0.146 (-)
Vote	0.468	0.537	0.473	0.459	0.336	0.291	0.227	0.143
Vote.EM	0.452 (-)	0.554 (+)	**0.497** (+)	0.467 (+)	0.322 (-)	0.292 (+)	0.253 (+)	0.152 (+)
Condorcet	0.428	0.491	0.485	0.486	0.287	0.245	0.248	0.165
Soboroff	0.416	0.480	0.477	**0.533**	0.285	0.233	0.245	**0.227**

Table 2. Zero-judgment (highest value in **bold**), training and testing on all systems.

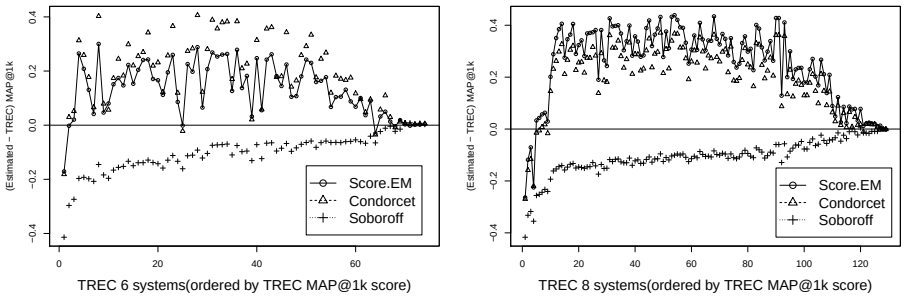

Fig. 1. Difference between the Estimated and TREC MAP@1k, Zero-judgment, Systems ordered by TREC MAP, best systems on the left and worst systems on the right.

the Y-axis is calculated by (estimated MAP@1k - TREC MAP@1k). Perhaps the most easily noted effect is that our Score.EM method and Condorcet tend to substantially overestimate the value of MAP@1k, while Soboroff's random-voting method tends to substantially underestimate MAP@1k. However, because our focus is on system comparison, systematic tendencies towards overestimation or underestimation are not consequential. The methods tend to give higher estimated MAP scores to the systems that are in the middle range, but lower scores to the best and worst systems at the two ends. This may be because the systems in the middle use similar searching strategies and return similar results, while the best and worst systems may return results that are not retrieved by other systems. The effect of majority voting causes underestimation for systems that behave distinctively, and all the zero-judgment methods tend to make similar errors.

4.3 Incomplete Judgment Results

Reliability of System Comparisons The difficulty that zero-judgment techniques exhibit with even detecting, much less distinguishing between, the best systems, motivates

our interest in approaches that can use partial, but still incomplete, human judgments. Figure 2 shows how τ_{ap} grows as we add increasing numbers of human judgments. We start from the Score.EM method, which was the best overall choice in the zero-judgment case, and we consider four ways of selecting documents to be judged at each step, policies P_1, P_2, P_3 and P_4 from Section 3.2. We compare to the following four state-of-the-art baselines: xinfAP [16], BPref-10 [3], RTC [4] and Hedge [1].[3]

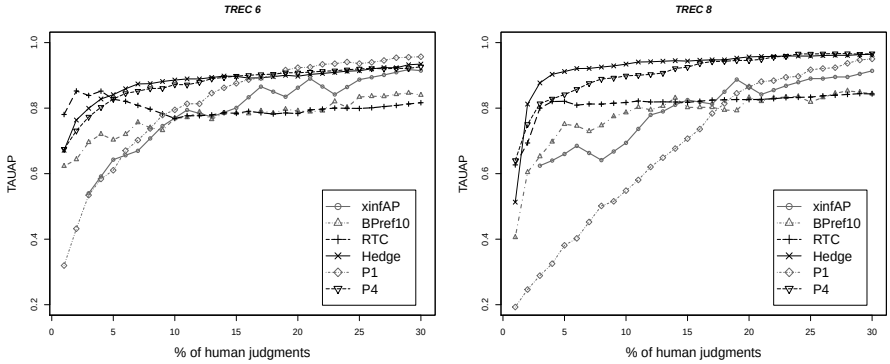

Fig. 2. τ_{ap} by percent of pooled judgment used for different TREC Ad Hoc collections.

	τ				τ_{ap}			
	TREC 5	TREC6	TREC 7	TREC 8	TREC 5	TREC 6	TREC 7	TREC 8
P_1	21	14	21	16	31	19	31	25
P_2	22	**12**	22	16	28	20	33	24
P_3	78	72	75	79	89	80	84	86
P_4	**5**	17	5	3	**5**	**16**	**12**	11
Hedge	**5**	17	**3**	**2**	13	19	21	**4**
RTC	16	80	2	26	29	$*$	77	$*$
BPref	$*$	68	26	37	$*$	$*$	$*$	78
xinfAP	23	21	18	19	26	27	32	29

Table 3. Human judgments (%) required to achieve $\tau = 0.9$ or $\tau_{\text{ap}} = 0.9$ ($*$ means unachievable).

[3] In our implementation of Hedge, the tunable parameter β is set constantly as 0.9; the results are reported on MAP@1k; the unjudged documents are considered as not relevant; losses are transformed linearly into the range of [0,1]; for each topic, there will be one document selected to be judged in each iteration.

For our RTC results, we used the code package *mtc-eval.tgz* from the RTC developers.[4] The computational complexity of our proposed EM-based method is $O(CSTn)$, where S is the number of systems, T the number of topics, n the number of unique documents for a topic in the pool, and C the number of iterations before the convergence. Empirically, C is smaller than 40. The time complexity of RTC is $O(S^2Tn^3)$. Running MAP@1000 on TREC 5 would take more than two days on a four-core 3.10GHz PC. Therefore, we only return RTC results for MAP@100.

As can be seen from Figure 2, Hedge and P_4 achieve better τ_{ap} with fewer human judgments. On TREC 6, RTC briefly dominates other methods until more than 5% of the judgments are available. Table 3 shows the percentage of human judgments needed for the methods to achieve $\tau = 0.9$ or $\tau_{\text{ap}} = 0.9$. As can be seen, on three of the four collections P_4 needs fewer human judgments than the other methods when measured by τ_{ap}. Hedge does best when measured by τ on three collections, suggesting that Hedge is better at distinguishing between lower-ranked systems than between the highly ranked systems that we care most about. BPref never achieves $\tau = 0.9$ or $\tau_{\text{ap}} = 0.9$ for some collections, even with full judgments.

Though weighting the documents in different ways, Hedge, RTC, and using policies P_1, P_2, P_4 with our EM method all tend to select for judgment documents that are highly ranked by systems. The computational complexity of these three methods compares as follows: RTC $>$ Hedge $>$ EM. With a limited budget for judging documents (e.g., a few hundred), RTC will be the best choice. If more documents could be judged (e.g., a few thousand), Hedge's way of selecting documents should be considered. On the other hand, P_4's performance on τ_{ap} shows that the strongest choice overall is to use Hedge's loss functions with EM's learning framework.

Effect of EM Learning Figure 3 shows the improvement on four collections of P_1 with human judgments, over a non-learning baseline, with available judgments increases from 1% to 20%, measured by τ_{ap}. In the non-learning baseline, the pseudo-judgments of the as-yet unjudged documents are taken from the zero-judgment iteration of the EM algorithm; the effect of additional human judgments is only to assign true relevance judgments to documents for evaluation, not to train the learner. In other words, there is no E-step or M-step in the non-learning baseline. The values shown are the differences from this baseline of the τ_{ap} of the full EM P_1 method (Section 3.2), which refines its pseudo-judgements with each iteration of new data.

A notable result is that learning with P_1 actually hurts performance with 1% judgments. The reasons for this are unclear. Possibly, a small number of human judgments are insufficient to allow the the model to distinguish the better (often, "manual") systems from the less effective (often "automatic") systems. As more human judgments are added, the P_1 method seems to quickly overcome the spurious effect of this mistuning. The improvements are substantial with 2% to 9% of the judgments, and decrease progressively from 10% to 20%. In part, this is because there is less room for improvement; as more documents are assessed, there are fewer for which pseudo-judgments are required, and the effectiveness of the pseudo-judgment method therefore becomes progressively less important.

[4] `http://ir.cis.udel.edu/~carteret/downloads.html`

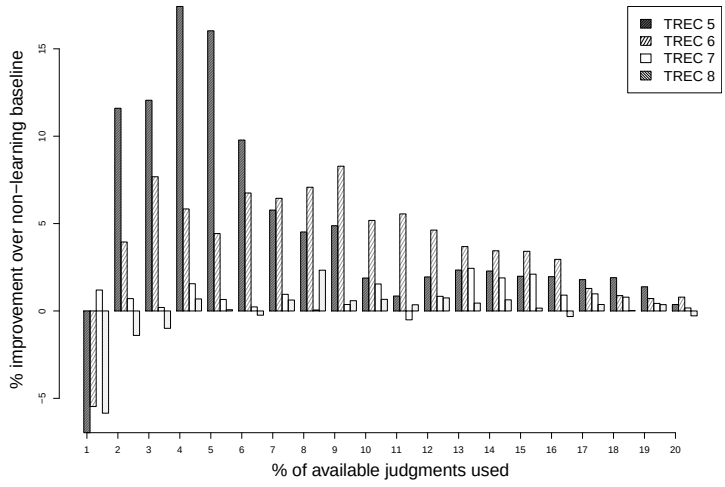

Fig. 3. % improvement by using EM learning, measured by τ_{ap}, with human judgments from 1% to 20%.

5 Conclusion

In this paper, we have introduced a novel method for automated or semi-automated system evaluation, based on the expectation-maximization (EM) algorithm. Our EM method predicts document relevance by system rankings, and system quality by the predicted relevance of the documents they return, iterating till convergence.

For zero-judgment evaluation, we have compared our method with the Condorcet method of Nuray and Can [11] and the random-voting method of Soboroff et al. [12]. Our methods beat the Condorcet method on all four of the test collections that we tried, and beat Soboroff's random-voting method on three of four. For incomplete judgments, we find that with a limited judging budget, a few hundred documents for example, RTC will be the better choice. However, the high computational complexity of RTC may make it impractical for interactive use with large collections. If more documents are judged, a few thousand for example, our results indicate that using EM's learning framework with Hedge's loss function would be a good choice.

Acknowledgments This material is based in part on work supported by the National Science Foundation under Grant No. 1065250 and by the Human Language Technology Center of Excellence at Johns Hopkins University. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Aslam J, Pavlu V, and Savell R (2003). A unified model for metasearch and the efficient evaluation of retrieval systems via the hedge algorithm. In: Proc. 26st Annual International ACM SIGIR, pp. 393–394.

[2] Aslam J, Pavlu V, Yilmaz E (2006) A statistical method for system evaluation using incomplete judgments. In: Proc. 29th Annual International ACM SIGIR, pp 541–548

[3] Buckley C, Voorhees E (2004) Retrieval evaluation with incomplete information. In: Proc. 27th Annual International ACM SIGIR, pp 25–32

[4] Carterette B (2007) Robust test collections for retrieval evaluation. In: Proc. 30th Annual International ACM SIGIR, pp 55–62

[5] Carterette B, Allan J (2005) Incremental test collections. In: Proc. 14th ACM International Conference on Information and Knowledge Management, pp 680–687

[6] Cormack GV, Palmer CR, and Clarke CL (1998). Efficient construction of large test collections. In: Proc. 21st Annual International ACM SIGIR, pp. 282–289.

[7] Dai K, Pavlu V, Kanoulas E, Aslam JA (2012) Extended expectation maximization for inferring score distributions. In: Proc. 34th European Conference on IR Research, pp 293–304

[8] Hauff C, Hiemstra D, Azzopardi L, de Jong F (2010) A case for automatic system evaluation. In: Proc. 31st European Conference on IR Research, pp 153–165

[9] Hosseini M, Cox IJ, Milic-Frayling N, Kazai G, and Vinay V (2012) On aggregating labels from multiple crowd workers to infer relevance of documents. In: Proc. 34th European Conference on IR Research, pp 182–194

[10] Kendall MG (1948) Rank Correlation Methods, 1st edn. Charles Griffin, London

[11] Nuray R, Can F (2006) Automatic ranking of information retrieval systems using data fusion. In: Information Processing & Management 42(3), pp 595–614

[12] Soboroff I, Nicholas C, Cahan P (2001) Ranking retrieval systems without relevance judgments. In: Proc. 24th Annual International ACM SIGIR, pp 66–73

[13] Spärck Jones K, van Rijsbergen CJ (1975) Report on the need for and provision of an 'ideal' test collection. Tech. rep., University Computer Laboratory, Cambridge

[14] Yilmaz E, Aslam J (2006) Estimating average precision with incomplete and imperfect judgments. In: Proc. 15th ACM International Conference on Information and Knowledge Management, pp 102–111

[15] Yilmaz E, Aslam J, Robertson S (2008) A new rank correlation coefficient for information retrieval. In: Proc. 31st Annual International ACM SIGIR, pp 587–594

[16] Yilmaz E, Kanoulas E, Aslam JA (2008) A simple and efficient sampling method for estimating ap and ndcg. In: Proc. 31st Annual International ACM SIGIR, pp. 603–610.

[17] Zobel J (1998) How reliable are the results of large-scale information retrieval experiments? In: Proc. 21st Annual International ACM SIGIR, pp 307–314