Unbiased Concurrent Evaluation on a Budget

<omitted>

ABSTRACT

Eliciting expert judgments for evaluating the performance of structured prediction systems (e.g., search engines, recommender systems) is labor-intensive and costly. This raises the question of how to get high-quality performance estimates given a relatively small budget of judgments. In this paper, we provide theoretically justified – yet highly practical and efficient – methods for selecting a subset of items to judge, addressing four commonly encountered evaluation scenarios: estimating the performance of a single system, comparing a system to a benchmark system, and estimating the relative and absolute performances of k systems. Our approach is based on designing effective importance sampling estimators that give provably unbiased performance estimates and that can re-use previously collected judgments. To this effect, we exploit the fact that many performance metrics of interest can be expressed as an expectation and derive theoretically optimal importance sampling distributions for estimating this expected value. We empirically demonstrate the effectiveness of our framework, showing that it eliminates the bias inherent in deterministic selection strategies (e.g. the pooling method) and that it can drastically reduce the number of judgments necessary compared to naive sampling approaches.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Performance, Measurement, Experimentation, Design

Keywords

importance sampling, concurrent evaluation, ranking

1. INTRODUCTION

Evaluating structured prediction systems using ground-truth expert judgments is challenging. These systems (e.g. a search engine) map an input $x \in \mathcal{X}$ (e.g. query) to a structured output $\vec{\boldsymbol{y}}$ (e.g. ranking), and one would like to estimate the expected utility that these $\vec{\boldsymbol{y}}$ provide over a distribution of x. Since assessing the utility of a complete structured object $\vec{\boldsymbol{y}}$ is difficult for human assessors, the utility of y is typically aggregated from the utilities of its parts. For rankings, this leads to performance measures like Precision@10 (the fraction of relevant documents in the top 10 of ranking y), Discounted Cumulative Gain (DCG) [10], and virtually all other performance measures used in information retrieval. These measures reduce the cognitive complexity of the assessment problem for the human assessors, since now only individual documents need to be judged for relevance. Other structured prediction tasks, like sequence labeling, friend recommendation, parsing, and network prediction, use similar part-based performance measures.

Unfortunately, even with a performance measure that decomposes, collecting judgments on the sub-parts is still costly and time-consuming. This raises the following question: Given a specific performance measure (like Precision@10), how should we pick which sub-parts get judged, when we are limited by an annotation budget? We assert that a good evaluation strategy should satisfy three criteria:

1. Statistically unbiased: If the true performance of a system is U_S, the performance estimator should also report U_S in expectation.

2. Statistically efficient: The variance in the performance estimator should be low.

3. Re-usable judgments: Getting new judgments is expensive and slow. Not having to get new judgments every time a new system is evaluated can drastically reduce cost and speed up the development cycle.

Using Precision@10 as an example, the naïve approach of getting all top-10 documents judged for a sample of queries does not fare well under these criteria. As we will see later, its statistical efficiency is suboptimal. Furthermore, it does not yield re-usable judgments. In fact, if we attempt to re-use judgments collected by the naïve approach, when presented a new system's ranking which differs from the judged system in the top-10 positions, we underestimate the new system's performance unless we get additional judgments.

In this paper, we provide practical methods for collecting expert judgements and estimating performance on a fixed budget that fulfill all three criteria. Our approach exploits that many decomposable performance measures can be expressed as an expectation of an appropriately defined categorical random variable. Based on this idea, [24, 17] have previously proposed a stratified sampling technique for estimating system performance. We extend this idea in several directions. In particular, we propose performance estimators based on importance sampling for the following four evaluation settings:

- Estimate the absolute performance of a single system.
- Given a baseline system and a proposed system, evaluate the difference in performance between the two.
- Given k systems, evaluate the absolute performance of all systems concurrently.
- Given k systems, evaluate the relative differences in performance.

For all evaluation settings, we propose notions of optimality and derive theoretically optimal sampling distributions. These distributions have closed-form solutions from which it is straightforward to sample, making our approach easy to apply in practice. Measures computed from these samples are guaranteed to be unbiased and efficient, and with marginal overhead (i.e., bookkeeping of the *propensities* during sampling) we can re-use these judgements to evaluate future systems. Finally, we provide guidance for statistical testing with our performance estimators. We empirically demonstrate the effectiveness of our approach, showing that it can offer drastically reduced variances when compared to standard techniques.

2. SAMPLING-BASED EVALUATION

The evaluation problem we address in the following can be formalized as follows. We have a number of systems $\mathcal{S} = \{S_1, \dots S_k\}$ which we wish to evaluate on a distribution of inputs $P(X)$. The performance of a system's prediction $\vec{\boldsymbol{y}} = S(\boldsymbol{x})$ for a single s is denoted as $U(\boldsymbol{x}, \vec{\boldsymbol{y}})$. The true performance of S over $P(X)$ is the expected utility

$$U_S = \mathbb{E}_P[U(X, S(X)] = \int U(\boldsymbol{x}, S(\boldsymbol{x}))dP(\boldsymbol{x}). \tag{1}$$

We consider performance measures $U(\boldsymbol{x}, \vec{\boldsymbol{y}})$ that decompose into parts of $\vec{\boldsymbol{y}}$, where each part y of $\vec{\boldsymbol{y}}$ has a utility $u(\boldsymbol{x}, y)$. Without loss of generality, assume $u(\boldsymbol{x}, y) \in [0, 1]$. For the ranking example, $\mathcal{S}$ would be a collection of retrieval systems, $P(X)$ the distribution of users and their queries, the parts y are individual documents in the ranking predicted by a system S, and $u(x, y)$ is the relevance of document y for query x. The $u(x, y)$ are unknown, but may be ascertained at some expense, e.g., by querying a human expert.

For many performance measures U, their decomposition can be expressed as a weighted linear sum of the utilities of the parts

$$U(x, \vec{\boldsymbol{y}}) = \sum_{y \in \vec{\boldsymbol{y}}} \lambda_{y,\vec{\boldsymbol{y}}}\, u(x, y), \tag{2}$$

with $\lambda_{y,\vec{\boldsymbol{y}}} \geq 0$ and known. An example of a decomposable utility function is Precision@k, where $u(x, y) = \mathrm{rel}(x, y) \in \{0, 1\}$ denotes binary relevance of document y for query x and the weights $\lambda_{y,\vec{\boldsymbol{y}}}$ are $1/k$ if y appears among the top k documents in the ranking $\vec{\boldsymbol{y}}$, and zero otherwise. Table 1 shows more examples of performance measures that decompose as a linear function of the individual parts of $\vec{\boldsymbol{y}}$. Similarly, Table 2 presents examples of performance measures where the natural decomposition of $\vec{\boldsymbol{y}}$ is into pairs of variables, i.e., $y = (\tilde{y}_1, \tilde{y}_2)$. The large number of commonly used performance measures in the two tables illustrates how widely applicable the decomposition in Eqn. (2) is.

Metric	$u(x, y)$	$\lambda_{y,\vec{\boldsymbol{y}}}$
Prec@k	$\mathrm{rel}(x, y) \in \{0, 1\}$	$\mathbf{1}_{\mathrm{rank}(y)\leq k}/k$
Accuracy	$\mathrm{correct}(x, y) \in \{0, 1\}$	$1/\lvert\vec{\boldsymbol{y}}\rvert$
Mean	$\mathrm{value}(x, y) \in [0, 1]$	$1/\lvert\vec{\boldsymbol{y}}\rvert$
MAE	$\lvert\mathrm{error}(x, y)\rvert \in [0, 1]$	$1/\lvert\vec{\boldsymbol{y}}\rvert$
MSE	$(\mathrm{error}(x, y))^2 \in [0, 1]$	$1/\lvert\vec{\boldsymbol{y}}\rvert$
DCG	$\mathrm{rel}(x, y) \in [0, 1]$	$1/\log(\mathrm{rank}\ (y \text{ in } \vec{\boldsymbol{y}}))$

Table 1: Popular metrics for ranking and recommendation that can be written as expectations over single item judgements.

Metric	$u(x, y)$	$\lambda_{y,\vec{\boldsymbol{y}}}$
#ofSwaps	$\mathrm{swapped}(x, y) \in \{0, 1\}$	$1/\lvert\vec{\boldsymbol{y}}\rvert$
wSwaps	$\mathrm{swapped}(x, y) \in \{0, 1\}$	$1/(\lvert\vec{\boldsymbol{y}}\rvert \cdot \mathrm{rank}(\tilde{y}_1) \cdot \mathrm{rank}(\tilde{y}_2))$
AP	$\mathrm{rel}(x, \tilde{y}_1) \cdot \mathrm{rel}(x, \tilde{y}_2) \in \{0, 1\}$	$\mathbf{1}_{\mathrm{rank}(\tilde{y}_2)\leq \mathrm{rank}(\tilde{y}_1)}\ /\ (R \cdot \mathrm{rank}(\tilde{y}_1))$

Table 2: Metrics for ranking and recommendation that can be written as expectations over pairs of items. $R = \sum_{\tilde{y}} \mathrm{rel}(x, \tilde{y})$**.**

2.1 Sampling Method

We approach the problem of estimating U_S through sampling not only over $P(X)$, but also over the parts of $\vec{\boldsymbol{y}} = S(x)$. We motivate sampling by the observation that U_S can be expressed as an expectation. Define

$$\Pr(y \mid x; S) = \frac{\lambda_{y,S(x)}}{\sum_y \lambda_{y,S(x)}}.$$

To simplify the exposition, we assume that $\sum_y \lambda_{y,S(x)} = 1$ for all systems S and inputs x. Our results can be extended in a straightforward manner to hold when this assumption is false. Then,

$$U_S = \mathbb{E}_{x\sim P(X)}\mathbb{E}_{y\sim \Pr(Y|x;S)}[u(x, y)]. \tag{3}$$

In principle, we can use any unbiased Monte Carlo technique to estimate U_S in Equation 3, and [24, 17] have used stratified sampling for estimating AP and nDCG. We choose importance sampling for three reasons. First, importance sampling makes it straightforward to incorporate prior knowledge into the sampling distribution. This could be knowledge about the utility values $u(x, y)$ or about the systems being evaluated. Second, we can obtain confidence intervals with only little additional overhead. Third, the importance sampling framework naturally extends to scenarios involving concurrent evaluation of multiple systems, as shall be shown later.

Central to importance sampling is the idea of defining a sampling distribution $Q(x, y)$ that focuses on the regions of the space that are most important for an accurate estimate of an expectation. For a sample of n observations $\{(x_i, y_i)\}_{i=1}^n$ drawn i.i.d. from a sampling distribution $Q(x, y)$, and a probability distribution $P(x, y; S) = \frac{\Pr(y|x;S)}{|X|}$, the importance sampling estimator for Eq. (3) is

$$\hat{U}_{S,n} = \frac{1}{n} \sum_{i=1}^{n} u(x_i, y_i) \frac{P(x_i, y_i; S)}{Q(x_i, y_i)}. \tag{4}$$

In order for this estimator to be unbiased, we must ensure that $Q(x, y) > 0$ whenever $u(x, y)P(x, y; S) > 0$ [16]. However, it is not necessary that any particular sample includes the assessment of all parts y of all system outputs $\vec{\boldsymbol{y}} = S(\boldsymbol{x})$ to provide an unbiased estimate. For example, when estimating Precision@10, we may get far fewer than 10 judgments per query and still have an unbiased estimator.

Given the performance measure and the output $\vec{\boldsymbol{y}}$ of a system S, $P(x, y; S)$ is known and can easily be computed for any x, y. Informally speaking, $P(x, y; S)$ is the scaled weight that a system S assigns to a part y under input x. The only thing left to choose is the sampling distribution $Q(x, y)$. The remainder of this paper will present optimal sampling distributions for a variety of different scenarios. Intuitively, the variance we observe when sampling reflects our uncertainty about the true judgements $u(x, y)$, captured by the degree of mismatch between our sampling distribution $Q(x, y)$ and $P(x, y; S)$. In the next sections we will explore criteria for selecting $Q(x, y)$ based on the available knowledge in order to maximize the statistical efficiency of the estimate under budget constraints.

Once the sampling distribution $Q(x, y)$ is determined, the actual performance estimation method is extremely simple:

1. Given a budget of n assessments, draw n pairs (x, y) (e.g. query x, document y) from $Q(x, y)$.

2. Collect assessments $u(x, y)$ of these n pairs.

3. Compute $\hat{U}_{S,n}$ according to Eq. (4).

A natural question to ask is whether there can be any deterministic subset-selection scheme that results in an unbiased estimate of U_S. After all, deterministic schemes are very easy to implement and understand. However, we can see from Equation (2) that the answer to this question, in general, is no. More specifically, any deterministic scheme that does not include all pairs (x, y) with $\lambda_{y,\vec{\boldsymbol{y}}}\, u(x, y) > 0$ will be biased since it misses mass from the total estimate. From Table 1, we see that even standard performance measures like accuracy assign strictly positive $\lambda_{y,\vec{\boldsymbol{y}}}$ for all y, so a deterministic scheme on a tight budget cannot, in general, include all such pairs. The experiments in Section 3.2 will revisit this issue.

2.2 Outline of Results

The following provides an outline of the remainder of the paper. We start by presenting results and experiments for evaluating a single system in Section 3, discussing the design of optimal sampling distributions and comparing it against conventional deterministic evaluation techniques like pooling [15]. In Section 4, we extend these results to the absolute evaluation of multiple systems in parallel, and consider the setting of comparative evaluation of two and more systems in Section 5. Section 6 discusses two ways of obtaining confidence intervals under our approach. Lastly, Section 7 briefly talks about how to extend our framework to handle noisy feedback and settings with multiple experts.

3. EVALUATING A SINGLE SYSTEM

We first consider the problem of estimating the performance U_S of a single system S via importance sampling. The key question to address is: How should we design the sampling distribution $Q(x, y)$ in order to reduce the variance of the estimator?

As mentioned in the introduction, a good evaluation method should be unbiased. To achieve unbiasedness, we have to constrain the sampling distribution to fulfill $P(x, y; S) > 0 \Rightarrow Q(x, y) > 0$. This constraint on the sampling distribution can be easily enforced, since $P(x, y; S)$ is known.

The second desirable property mentioned in the introduction is statistical efficiency. We formalize this by seeking an estimator that has low variance. A key result for importance sampling is that the following sampling distribution is optimal for minimizing variance [22]:

$$Q^*(x, y) \propto u(x, y) \cdot P(x, y; S). \tag{5}$$

Unfortunately, this sampling distribution cannot be computed since it requires knowledge of the $u(x, y)$. These are, of course, unknown a priori, since they are precisely the relevance judgments that we seek to elicit. However, we can use estimates of $\tilde{u}(x, y)$ based on prior side information (e.g. the TFIDF score of document y for query x) to derive the sampling distribution,

$$Q(x, y) \propto \tilde{u}(x, y) \cdot P(x, y; S), \tag{6}$$

provided that these estimates are "thick-tailed" enough to not violate the conditions necessary for unbiasedness [26]. In the simplest case we can approximate $u(x, y)$ with a constant $\tilde{u}(x, y) = 1$ for all x and y.

We now empirically explore the effectiveness of these importance samplers.

3.1 Data and Experiment Setup

We use two datasets for our experiments, *SYNTH* and *TREC*. The SYNTH dataset is designed to mimic judgements as they occur in a recommender system, whereas TREC contains binary relevance judgements of various information retrieval systems. In total, the TREC dataset comprises 149 systems that submitted rankings $\vec{\boldsymbol{y}}$ of size 1000 as response to 50 queries x from the TREC-8 ad hoc track [21]. Note that for all experiments of this paper, we do not work with the true distribution $P(X)$ over all possible inputs, but with the empirical distribution $\hat{P}(x)$ that is defined by all the inputs x in the respective dataset.

To eliminate unwanted side effects when dealing with unjudged documents, we only kept the TREC-8 systems that had all the top 100 documents of each of the 50 queries fully judged. We truncated all rankings to length 100, so that all the documents in the dataset also had judgements. Imputing zeros for all unjudged documents would possibly preserve more information, but would introduce a bias in favor of judged documents. After the filtering, we had a total of 64 systems in our final dataset.

To create SYNTH, we generated a sample of 6000 rankings $\vec{\boldsymbol{y}}_i$ with 2000 items each. The ground truth judgments

$u(x, y) \in \{0, ...4\}$ for each ranking were drawn from a categorical distribution whose parameters were drawn from a Dirichlet distribution with $\alpha = (0.54, 0.25, 0.175, 0.03, 0.005)$. We created systems of different quality in the following way. Given a fixed set of true relevances generated as described above, S_{OPT} denotes the perfect ranking system where each ranking $\bar{\boldsymbol{y}}$ is sorted according to the true relevances $u(x, y)$. The $S_{SHIFT-m}$ ranking system shifts all rankings of S_{OPT} by m entries to the right; elements that get shifted beyond the last position get re-introduced at the first. $S_{REVERSE-m}$ reverses the order of the top m elements in S_{OPT}. The final set of systems in SYNTH was $\mathcal{S} = \{S_{OPT}, S_{REVERSE-75}, S_{REVERSE-150}, S_{SHIFT-5}, S_{SHIFT-7}\}$.

3.2 Sampling vs. Deterministic Budgeting

We start by comparing our sampling approach to conventional evaluation techniques used in information retrieval (IR). The TREC evaluations in IR have historically used a deterministic method for dealing with a budget constraint on the number of relevance assessments. This method, called *pooling* [15], works by creating a pool of documents to be judged from the union of the top-b (usually, $b = 100$) documents in the participating ranking systems. The complete pool is judged and all documents that did not enter the pool are assumed to be irrelevant.

The depth of the pool vs the number of queries to judge is the key design choice when having to adhere to a budget. We use two baseline methods that choose this trade-off differently. Given a budget of b, the *TOP* baseline gets judgments for the top $\frac{b}{|X|}$ items of each query and computes the average performance based on these judgments. The *DEEP* baseline, on the other hand, gets judgments as deep as possible into the pool. To meet the same budget, it randomly selects $\left\lceil \frac{b}{|\bar{\boldsymbol{y}}|} \right\rceil$ queries and judges each to full depth.

Table 3 compares estimated DCG@k ($k = 2000$ on SYNTH; $k = 100$ on TREC) for the deterministic baselines TOP and DEEP with the estimates from our sampling method, where $Q(x, y)$ is constructed according to Eq. (6) with an estimated $\tilde{u}(x, y)$ as detailed in the next section. We refer to this method as UBIS($\tilde{u} \cdot P$) – UnBiased Importance Sampling. All methods used a per-query budget of 5, resulting in total budget of 30,000 assessments for SYNTH and 250 assessments for TREC. We repeated each experiment 100 times, and averaged the $\hat{U}_S$ values across all 100 runs. The error intervals correspond to the empirical standard standard deviations of $\hat{U}_S$ observed by repeating the entire experiment 100 times.

Unsurprisingly, Table 3 shows that TOP is heavily biased and systematically underestimates the true U_S given in the first column. This happens because TOP assumes that all items with rank greater than b are not relevant. The bias of TOP is so strong that even its average estimates do not reflect the correct ordering of the ranking functions by true $\hat{U}_S$ on SYNTH.

Table 3 shows that our UBIS($\tilde{u} \cdot P$) method is indeed unbiased as expected, but DEEP is unbiased as well since it exhaustively judges every document up to rank k. However, the two estimators differ in their variance. For UBIS($\tilde{u} \cdot P$), the $\hat{U}_S$ estimates of all ranking functions are separated by at least two standard deviations, indicating that UBIS($\tilde{u} \cdot P$) can reliably distinguish the ranking performance of the systems. The standard deviations of the DEEP method, however, are much larger than the estimated differences. This again is not surprising, since DEEP suffers from between-query variability when sampling only a few queries in the rankings. Also, the variances of DEEP on TREC are larger than on SYNTH – this is consistent with the fact that queries in SYNTH were constructed to be fairly homogenous.

3.3 Incorporating Prior Knowledge into Q

We will show in this section how incorporating prior knowledge into the sampling distribution can increase efficiency of our estimators. We compare three sampling distributions $Q(x, y)$ that include increasing amounts of prior knowledge.

UBIS(1) uses the least informed distribution, setting $Q(x, y)$ to the uniform distribution $Q(x, y) \propto 1$.

A more informed sampling distribution, which we call $\tilde{u}(x, y) = 1$, is of the form

$$Q(x, y) \propto P(x, y) + \epsilon, \tag{7}$$

where $\epsilon = 0.05$ is a small constant which is added to ensure that we have sufficiently heavy tails. By including $P(x, y)$ into the sampling distribution, it gains the knowledge that documents higher in the rankings contribute with larger weight to the expectation we seek to estimate.

Finally, Eq. 6 indicates that we can also include prior knowledge about how likely y is to be relevant to x. This leads to the method UBIS($\tilde{u} \cdot P$), which uses the distribution

$$Q(x, y) \propto \tilde{u}(x, y) \cdot P(x, y) + \epsilon \tag{8}$$

with $\epsilon = 0.05$. We design $\tilde{u}(x, y)$ to reflect that more relevant documents are typically at the top of the ranking. For SYNTH, we use $\tilde{u}(x, y) = 4 \cdot (1 - \frac{\text{rank}(x,y)}{2000})$ – reflecting the fact that relevances ranged between 0 and 4. We have an analogously decreasing function $\tilde{u}(x, y) = \frac{16}{\text{rank}(x,y)+34}$ for TREC – this was calibrated by computing the average label at rank k across all systems.

Table 4 compares the three sampling approaches. The experiments use the same settings as for Table 3, i.e., a per-query budget of 5 and averaging across 100 runs for all DCG@k estimates. As expected, there is no evidence of bias in any of the approaches. However, we see that variance increases as we transition from UBIS($\tilde{u} \cdot P$) to UBIS(P), and it increases even further if we sample uniformly in UBIS(1). We conclude from this that the choice of Q is of great practical relevance. This is in line with importance sampling theory that indicates that $\tilde{u}$ should be as close to the true relevances as possible.

3.4 Reusing Previously Collected Data

Using DCG without a depth cutoff as the performance measure of interest was a worst-case scenario for the deterministic budgeting methods with respect to their bias. If we had chosen DCG@5, the TOP method would have given us an unbiased estimate (since the query budget is 5). However, the following shows that we would still incur a bias if we used the ratings collected by TOP for one ranker when evaluating a new ranker.

Consider the problem of Precision@10 estimation for different ranking systems. We simulate the effects of reusing previously collected data in the following way. We start by obtaining a number of judgements for a set of training systems $\mathcal{S}_{train}$. In particular, in the TOP approach we get judgments for all top 10 documents of all queries and all systems in $\mathcal{S}_{train}$. This provides us with unbiased estimates of

	U_S	TOP	DEEP	UBIS($\tilde{u} \cdot P$)
OPT	284.40	16.98 ± 0.00	284.48 ± 1.69	284.54 ± 1.22
REVERSE-75	277.63	10.00 ± 0.00	277.79 ± 1.75	277.58 ± 1.07
REVERSE-150	271.32	8.51 ± 0.00	271.21 ± 1.59	271.18 ± 0.97
SHIFT-5	274.94	4.72 ± 0.00	275.03 ± 1.69	274.73 ± 1.10
SHIFT-7	269.87	0.00 ± 0.00	269.90 ± 1.75	269.98 ± 1.12
mds08a3	6.96	1.93 ± 0.00	7.32 ± 5.05	6.86 ± 0.76
nttd8ale	9.01	2.41 ± 0.00	8.79 ± 5.30	8.97 ± 0.87
weaver2	7.48	1.94 ± 0.00	8.40 ± 6.59	7.53 ± 0.83

Table 3: Mean DCG estimates across 100 trials for all systems in SYNTH (top) and three randomly chosen systems in TREC (bottom). TOP underestimates the true DCG of all systems, and the bias is inconsistent leading to an incorrect ordering of the SYNTH systems. Our sampling approach consistently achieves a lower variance DCG estimate compared to DEEP for all systems.

	U_S	UBIS($\tilde{u} \cdot P$)	UBIS(P)	UBIS(1)
OPT	284.40	284.54 ± 1.22	284.92 ± 1.98	284.63 ± 3.05
REVERSE-75	277.63	277.58 ± 1.07	277.70 ± 1.45	277.43 ± 2.64
REVERSE-150	271.32	271.18 ± 0.97	271.24 ± 1.33	271.48 ± 2.20
SHIFT-5	274.94	274.73 ± 1.10	275.01 ± 1.67	274.51 ± 2.63
SHIFT-7	269.87	269.98 ± 1.12	270.00 ± 1.45	269.79 ± 2.45
mds08a3	6.96	6.86 ± 0.76	7.05 ± 0.84	6.81 ± 0.97
nttd8ale	9.01	8.97 ± 0.87	9.04 ± 0.98	9.00 ± 1.18
weaver2	7.48	7.53 ± 0.83	7.38 ± 0.92	7.55 ± 1.05

Table 4: Our importance sampling approach outperforms uniform sampling (last column) and improves with more informed priors. We used the same settings as for the results in Table 3.

Precision@10 for all systems in $\mathcal{S}_{train}$. However, we now re-use these judgements to evaluate a number of new systems $\mathcal{S}_{test}$, making the pooling assumption that all unjudged documents are not relevant.

The second approach, UBIS($\tilde{u} \cdot P$), uses the sampling distributions for each of the training systems to draw documents to get judged, with additional book-keeping to keep track of the propensities $Q(x_i, y_i)$ for each sampled query-document pair (x_i, y_i). To evaluate a new system, we re-use these judgements and use Eq. (4) to derive Precision@10 estimates for each of the new systems.

In the following case study, $\mathcal{S}_{train}$ consists of 10 TREC systems and $\mathcal{S}_{test}$ contains 54 TREC systems. We use the same queries from TREC as in the previous section. Both approaches were allowed a per-query budget of 10. If we now use these *pooled* TOP judgements to evaluate each of the 34 new systems, whose average TRUE Precision@10 is listed in Table 5, we see that TOP systematically underestimates their performance. In contrast to this, using the judgements sampled by UBIS($\tilde{u} \cdot P$), we can get an unbiased estimates for these systems. This holds for any set of training and testing systems, but the variance of the estimator will increase if training and test systems produce increasingly different ranking.

4. EVALUATING MULTIPLE SYSTEMS CONCURRENTLY

For several applications of interest, we are interested not just in evaluating the absolute performance of a single system, but simultaneously evaluating multiple systems. We first define what we mean by optimality as a measure of statistical efficiency in this setting. When minimizing this measure, we obtain optimal sampling distributions in closed form. Previously used heuristics for concurrent evaluation of absolute performances do not have such theoretical guarantees (i.e., [1]). As demonstrated in the previous section, having access to a good prior on $u(x, y)$ will yield more efficient estimators in this setting as well.

method	mean estimates
TRUE	0.418 ± 0.000
TOP	0.356 ± 0.000
UBIS	0.413 ± 0.103

Table 5: Mean estimates and empirical standard deviation for Precision@10 across 54 test systems and 25 trials with $b = 10 \cdot |\vec{\boldsymbol{y}}|$. TOP has no standard devation, but systematically underestimates the performance of new systems.

4.1 Method

When we wish to create a leaderboard of k systems with each of their absolute performance listed, a natural measure of statistical efficiency is to consider the average variance in the estimated performances of each system. That is, we wish to find a Q^* that minimizes

$$\sum_{S \in \mathcal{S}} \operatorname{Var}\left[U_{S,n}\right]. \tag{9}$$

THEOREM 1. *The distribution Q^* that minimizes (9) is*

$$Q^*(x,y) \propto u(x,y)\sqrt{\sum_{S\in\mathcal{S}} P(x,y;S)^2}. \qquad (10)$$

PROOF. Let us start by proving the following fact. Let $\{b_S \in \mathbb{R}^+ : S \in \mathcal{S}\}$ be a collection of known scalar weights. Then,

$$\mathrm{Var}\left[\sum_{S\in\mathcal{S}} b_S \cdot \hat{U}_{S,n}\right] =$$
$$\frac{1}{n}\left[\sum_{x,y} \frac{u^2(x,y)\left(\sum_{S\in\mathcal{S}} b_S \cdot P(x,y;S)\right)^2}{Q(x,y)}\right]$$
$$-\frac{1}{n}\left[\sum_{S\in\mathcal{S}} b_S \cdot \hat{U}_{S,n}\right]^2,$$

where,

$$u^2(x,y)\left(\sum_S b_S P(x,y;S)^2\right) = Q(x,y) = 0$$
$$\Rightarrow \frac{u^2(x,y)\sum_S b_S P(x,y;S)^2}{Q(x,y)} = 0$$

This follows immediately from the definition of variance and the fact that (x_i, y_i) are drawn i.i.d from Q.

Using that fact, we know that

$$\sum_{S\in\mathcal{S}} \mathrm{Var}\left[\hat{U}_{S,n}\right] = \frac{1}{n}\mathbb{E}_Q\left[\frac{u^2(x,y)\sum_{S\in\mathcal{S}} P(x,y;S)^2}{Q(x,y)}\right]$$
$$-\frac{1}{n}\sum_{S\in\mathcal{S}} \mathbb{E}_P\left[u(x,y)P(x,y;S)\right]^2$$

Since Q appears only in the first term, it is sufficient to choose Q to minimize only the first term as well, subject to the constraint that $Q(\cdot,\cdot)$ is a probability distribution, i.e., that $Q(x,y) \geq 0$ and $\sum_{x,y} Q(x,y) = 1$. Observe that the probability simplex is convex, and the objective function is convex over this region. Thus, to prove optimality, it is sufficient to show that the KKT conditions are satisfied. Moreover, the choice of $Q^* \propto u(x,y)\cdot \sqrt{\sum_{S\in\mathcal{S}} P(x,y;S)^2}$ satisfies the Karush-Kuhn-Tucker (KKT) conditions. $\Box$

Note also, that in the worst case, $u(x,y) = 1$ yields the largest possible variance, so the minimax optimum Q^* for all possible u is achieved by $Q^*(x,y) \propto \sqrt{\sum_{S\in\mathcal{S}} P(x,y;S)^2}$.

We can now derive practical sampling distributions $Q(x,y)$ analogous to the case of single-system evaluation.

$$Q(x,y) \propto \tilde{u}(x,y)\sqrt{\sum_{S\in\mathcal{S}} P(x,y;S)^2}. \qquad (11)$$

Again, we can include an approximated prior $\tilde{u}$. With this sampling distribution, the overall evaluation procedure is as follows:

1. Given a budget of n assessments, draw n pairs (x,y) (e.g. query x, document y) from $Q(x,y)$.

2. Collect assessments $u(x,y)$ of these n pairs.

3. For each system S_i: compute $\hat{U}_{S_i,n}$ according to Eq. (4).

4.2 Experiments

In this experiment, we will compare our theoretically motivated way of evaluating the absolute performance of k systems to a heuristic that has been used in literature. Yilmaz et al. [25] propose that one should choose $Q^*(x,y) \propto u(x,y)\sum_{S\in\mathcal{S}} P(x,y;S)$. In other words, we just average all $P(x,y;S)$. We refer to this as *MEAN*. The estimator we get by using Eqn. (11) will be denoted as *SQRT*.

For the experiments, we construct the approximated prior $\tilde{u}$ by averaging the approximated priors of each system (which we defined to be dependent on the rank of y in S):

$$\tilde{u}(x,y) = \frac{1}{|\mathcal{S}|}\sum_{S\in\mathcal{S}} \tilde{u}(x,y;S)$$

Note that for this experiment, we do not employ sampling but report the averaged variances of our estimators on the complete dataset, i.e., Eqn. (9) divided by k. Looking at the full variances allows us to better observe the smaller differences between our methods since we eliminate finite sample variance. Note also that this corresponds to having an estimator with infinitely many samples, or equivalently, averaging the results over infinitely many trials with a fixed budget. This is why we denote the estimators as SQRT-∞ and MEAN-∞.

	SYNTH		TREC	
	$\tilde{u}(x,y)$	$u^*(x,y)$	$\tilde{u}(x,y)$	$u^*(x,y)$
SQRT-∞	6.11	0.42	38.29	1.74
MEAN-∞	6.43	0.46	16.56	2.80

Table 6: Average values of Eqn. 9 across systems in TREC and SYNTH. When using the true u^*, the theoretically optimal Q^* indeed achieves smaller variance across datasets.

Table 6 shows the limiting variances of the two methods under both perfect and imperfect priors. In the theoretically ideal setting where we have knowledge of the true $u^*(x,y)$ we see that SQRT-∞ consistently has a lower variance than MEAN-∞ as theory predicts. This picture can change when we move to an inferior prior $\tilde{u}(x,y)$. For SYNTH, we still see that SQRT-∞ is superior to simple averaging, however, the TREC systems do substantially better under the MEAN-∞ distribution. We believe that this is due to the imperfect fit of $\tilde{u}(x,y)$ for TREC; the SYNTH dataset had better matching priors since relevance scores were more homogeneous in that dataset. Indeed, when constructing $Q^*(x,y)$ according to Equation (10), it makes any differences between the k systems more pronounced than simple averaging due to the squaring inside the formula. As a consequence, for imperfect $\tilde{u}$, this can also amplify any negative effects that come from $\tilde{u}$ as we saw on TREC.

5. COMPARATIVE EVALUATION OF MULTIPLE SYSTEMS

Another typical scenario that arises in practice is the relative comparison of two or more systems to each other. In the first part of this section, we discuss the scenario of computing the difference in performance between two systems S and S'. We then extend this scenario in Section 5.2 to estimating the performance differences between multiple systems.

5.1 Estimating Difference of Two Systems

When we are evaluating how much better one system is than the other, the key quantity of interest is $U_S - U_{S'}$. Our measure of optimality is, consequently, $\mathrm{Var}\left[U_S - U_{S'}\right]$. Realizing that

$$\hat{U}_{S,n} - \hat{U}_{S',n} = \frac{1}{n}\sum_{i=1}^{n} u(x_i, y_i)\frac{P(x_i, y_i; S) - P(x_i, y_i; S')}{Q(x_i, y_i)}, \tag{12}$$

the optimal distribution is

$$Q^*(x, y) \propto u(x, y) \cdot \left|P(x, y; S) - P(x, y; S')\right|. \tag{13}$$

Proof. Define

$$w = u(x, y)\frac{P(x, y; S) - P(x, y; S')}{Q(x, y)}$$

Note that $\mathbb{E}_Q\left[w\right] = U_S - U_{S'}$. The variance of w is,

$$\mathbb{E}_Q\left[w^2\right] - \mathbb{E}_Q\left[w\right]^2 = \sum_{x,y}\frac{u^2(x, y)\left[P(x, y; S) - P(x, y; S')\right]^2 Q(x, y)}{Q^2(x, y)} - \left[\sum_{x,y} u(x, y)\left[P\left(x, y; S\right) - P\left(x, y; S'\right)\right]\right]^2.$$

Only the first term depends on Q, and we require that $u^2(x, y)\left[P(x, y; S) - P(x, y; S')\right]^2 > 0 \Rightarrow Q(x, y) > 0$. Clearly the proposed Q^* satisfies this property. Also the first term is lower bounded using Jensen's inequality,

$$\mathbb{E}_Q\left[w^2\right] \geq \mathbb{E}_Q\left[|w|\right]^2 = \mathbb{E}_{Q^*}\left[w^2\right]$$

for the Q^* defined above. Hence, the constructed Q^* minimizes $\mathrm{Var}\left[U_S - U_{S'}\right]$. □

This theoretically optimal sampling distribution derived makes sense intuitively. Items that have very similar weights $P(x_i, y_i; \cdot)$ in both systems will get sampled with a very low probability, since they have negligible effect on the performance difference.

In the experimental section, we will compare the efficiency of Q^* against a naïve approach to pairwise comparison – estimate each of $\hat{U}_{S,n}$ and $\hat{U}_{S',n}$ using sampling distributions $Q^*, Q^{*\prime}$ tailored for each of them, each estimator responsible for sampling half the evaluation budget, and reporting the difference of the estimated values.

5.2 Estimating Differences between Multiple Systems

A more general scenario is the comparative evaluation of more than two systems. A simple measure to optimize for when evaluating the relative performance of k systems among each other is to estimate $U_S - \frac{1}{k}\sum_{S' \in \mathcal{S}} U_{S'}$. In other words, we try to minimize the variance of each system's performance with respect to the average performance over all systems. The objective of interest is, hence, to minimize

$$\sum_{S \in \mathcal{S}} \mathrm{Var}\left[\hat{U}_{S,n} - \frac{1}{k}\sum_{S' \in \mathcal{S}} \hat{U}_{S',n}\right]. \tag{14}$$

Following an argument similar to the one of Theorem (1), the optimal sampling distribution for this scenario is

$$Q^*(x, y) \propto u(x, y)\sqrt{\sum_{S \in \mathcal{S}}\left[P(x, y; S) - \frac{1}{k}\sum_{S' \in \mathcal{S}} P(x, y; S)\right]^2}. \tag{15}$$

	SYNTH	TREC
TRUE	6.35 ± 0.00	0.12 ± 0.00
SINGLE	6.34 ± 0.53	0.14 ± 1.20
PW	6.34 ± 0.18	0.13 ± 1.07

Table 7: Pairwise sampling vs. individual sampling for $\binom{5}{2}$ pairs of systems and $b = 5\cdot|\vec{\boldsymbol{y}}|$. The DCG@k estimates were averaged across 100 trials and all pairs. Using the sampling distribution in Eqn (13) achieves lower standard error in the estimates compared to spending half the budget on estimating each individual system in the pair.

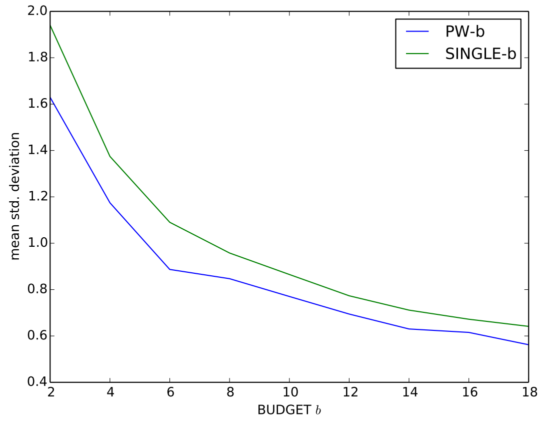

Figure 1: Mean standard deviation for $\binom{5}{2}$ random pairs in the TREC dataset. Across different budgets, sampling according to Eqn. (13) yields a lower variance estimator compared to the naive approach that splits the evaluation budget to estimate each system.

5.3 Experiments

In the first experiment, we look at the problem of estimating the difference in performance of two systems S and S'. Our baseline *SINGLE* draws $b/2$ samples according to (5) for each system and computes $F_S - F_{S'}$ from all of the samples, whereas our *PW* method gets to spend a budget of b on the optimal distribution from Eqn. (13). Both, SINGLE and PW use the same informative priors $\tilde{u}$ as in the previous experiments.

Table 7 presents the average pairwise differences of $\binom{5}{2}$ pairs from five randomly chosen systems for a budget of $b = 5 \cdot |\vec{\boldsymbol{y}}|$. We can see that for SYNTH, this cuts the mean standard deviation more than in half; for TREC the reduction is less. We attribute this to the fact that the systems in TREC are less similar to each other and hence, fewer weights will cancel out in the subtraction. However, this reduction in variance for TREC is still substantial as Figure 1 shows. The figure compares how much more efficient PW is on TREC

	SYNTH		TREC	
	$\tilde{u}(x,y)$	$u^*(x,y)$	$\tilde{u}(x,y)$	$u^*(x,y)$
SQRT-∞	0.74	0.40	35.32	1.56
KREL-∞	0.11	0.09	25.21	1.46

Table 8: Limiting variance as defined in Eqn. (14) **for all systems in SYNTH and TREC. SQRT-∞ refers to the estimator that is (theoretically) optimal for absolute performance of multiple systems as derived in Eqn.** (11)**. KREL-∞ uses the optimal estimator for relative performances of multiple systems developed in Eqn.** (15)**.**

relative to SINGLE. In order for SINGLE to roughly achieve the same standard deviation as PW, we seem to need a substantially larger budget – an increase by roughly 50%. Note that curves behave roughly like $1/\sqrt{b}$ due to the Central Limit Theorem (CLT) as for all sampling-based approaches.

We now turn to the evaluation of k systems with respect to their difference to the mean. We know from Eqn. (15) what the optimal sampling distribution Q^* has to look like. Let us denote *KREL* to denote the estimator that uses the latter equation to construct Q. We compare KREL against the SQRT estimator for estimating absolute performance of k systems from the last section. Again, the prior on u is set using the same $\tilde{u}$ mentioned earlier.

Similar to Section 4.2, Table 8 reports the limiting variances as defined by the objective in (14) of each estimator for both datasets. KREL-∞ does better on SYNTH than SQRT-∞; however, for TREC, KREL-∞ provides a smaller improvement over SQRT-∞ as for SYNTH. Again, this is similar to what we had seen in the pairwise difference experiments since queries in SYNTH are more homogenous.

6. CONFIDENCE INTERVALS

There are two major approaches for obtaining $1-\alpha$ confidence intervals in our setting. The first one is *approximate* confidence intervals, assuming that our observations come from a normal distribution. However, for small sample sizes and skewed distributions, this approximation can be inappropriate. In that case, one can obtain *exact* confidence intervals using distribution-free inequalities, like Hoeffding's inequality. Exact in this case refers to their property of having a coverage property of *at least* $1-\alpha$. Their main drawback is, however, that the obtained intervals are usually fairly loose as they make only few assumptions about the underlying random variables.

6.1 Approximate intervals

For large sample sizes n, our importance estimate $\hat{U}_{S,n}$ converges in distribution to a normal distribution. From the central limit theorem, we then obtain

$$\hat{U}_{S,n} \pm z_{\alpha/2}\frac{\hat{\sigma}_{S,n}}{\sqrt{n}} \tag{16}$$

as the approximate $1-\alpha$ confidence interval. For example, a 95% confidence interval would be $\hat{U}_{S,n} \pm 1.96\frac{\hat{\sigma}_{S,n}}{\sqrt{n}}$. We can estimate $\hat{\sigma}^2$ from the same samples $\{(x_i, y_i)\}_{i=1}^n$ that we used to compute $\hat{U}_{S,n}$ as follows

$$\hat{\sigma}_{S,n} = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{P(x_i,y_i;S)u(x_i,y_i)}{Q(x_i,y_i)} - \hat{U}_{S,n}\right)^2.$$

Note that the rate of convergence of $\hat{U}_{S,n}$ depends on the skewness $\mathbb{E}(w-F_S)^3/\sigma^3$ of the underlying distribution (i.e., the distribution that is defined by our importance sampler) as can be seen from the Edgeworth expansion [6]:

$$P\left(\hat{U}_{S,n} - F_S \le \frac{\sigma x}{\sqrt{n}}\right) = \Phi(x) + \frac{\mathbb{E}(w-F_S)^3}{6\sigma^3\sqrt{n}}(1-x^2)\phi(x) + o(1/\sqrt{n}).$$

In other words, the larger the skewness of our importance sampler, the slower we converge to the normal distribution. As a consequence, our estimated confidence intervals are more likely to be inaccurate. However, approximate intervals provide an easy and sample-dependent way of assessing the accuracy of our estimates. This is especially convenient when assessing new systems: The more similar a new system is to the sampling distribution that was used to obtain the judgements, the more confident we also are in our estimates.

6.2 Exact intervals

As we saw in the previous subsection, approximate intervals cannot give us guarantees about our estimates. For some applications, however, it might be necessary to have stronger guarantees. Hoeffding's inequality [8] gives us exact confidence intervals in a straight-forward way. It states that the sample mean $\overline{X} = \frac{1}{n}(X_1 + \cdots + X_n)$ of n independent random variables X_i with ranges R_i is bounded by

$$\mathbb{P}\left(\left|\overline{X} - \mathrm{E}\left[\overline{X}\right]\right| \ge t\right) \le 2\exp\left(-\frac{2n^2t^2}{\sum_{i=1}^n R_i^2}\right).$$

Rearranging the terms above gives us the following $1-\alpha$ confidence interval

$$\hat{U}_{S,n} \pm 2\sqrt{\frac{\ln(2/\alpha)\sum_{i=1}^n R_i^2}{2n^2}}.$$

If all the ranges are the same, the interval simplifies to

$$\hat{U}_{S,n} \pm 2R\sqrt{\frac{\ln(2/\alpha)}{2n}}. \tag{17}$$

A simple 95% confidence interval is $\hat{U}_{S,n} \pm 2R\sqrt{\frac{1.84}{n}}$.

As we can see, our confidence intervals depends on R – the range of the random variable whose expectation we are computing. By remembering $u(x,y) \in [0,1]$, we obtain a simple upper bound on R as follows

$$\begin{aligned} R &\le \max_{x,y}\frac{P(x,y;S)u(x,y)}{Q(x,y)} \\ &\le \max_{x,y}\frac{P(x,y;S)}{Q(x,y)}. \end{aligned}$$

Note that we can compute this quantity easily after we chose $Q(x,y)$ and before sampling. Also, note that the size of the confidence interval is mainly affected by our choice of $Q(x,y)$. If we pick a Q that is uniform, we will minimize that upper bound at the expense of ignoring all prior information we have. As we saw in Section 3.3, this leads to poor performance in practice.

In contrast to the approximate intervals from the previous subsection, the intervals obtained from Hoeffding's inequality are sample-independent. Hence, we ignore any information that we obtain during the sampling process. Also, they do not consider variance, which leads to non-zero confidence intervals even in the case of a zero-variance estimator. One interesting direction for future work would be to look at empirical Bernstein bounds [14] or the Anderson-Massart inequalities [13] as a possible remedy.

6.3 Experiments

For the same 100 runs as in Table 4, we computed both approximate and exact 95% confidence intervals according to Equations 16 and 17. We also computed the coverage probability of the approximate CLT confidence intervals, defined as the number of times the true value was within the confidence interval. As we can see from Table 9, the coverage probabilities of the approximate intervals are usually very close to 95%. The confidence intervals from HI are much more conservative. We can see that they are usually of the order of two of more times of the CLT intervals, although they guarantee coverage probabilities of at least 95%.

We can also verify that the empirical standard deviation of UBIS($\tilde{u}\cdot P$) is roughly 1/1.96 of the confidence intervals that the CLT gives us, although both methods used very different means of estimating the standard devation. Also, as a final remark, note that our confidence intervals hold independent of how large the input domain of $P(X)$ is. This is useful in scenarios where we are sampling from an constant stream of inputs x, as could be the case for live search engines.

	UBIS($\tilde{u}\cdot P$)	CLT	P_{cov}	HI
OPT	284.54 ± 1.22	± 2.21	0.92	± 367.79
REVERSE-75	277.58 ± 1.07	± 2.10	0.96	± 367.79
REVERSE-150	271.18 ± 0.97	± 2.05	0.95	± 367.79
SHIFT-5	274.73 ± 1.10	± 2.18	0.93	± 367.79
SHIFT-7	269.98 ± 1.12	± 2.18	0.94	± 367.79
mds08a3	6.86 ± 0.76	± 1.50	0.95	± 7.71
nttd8ale	8.97 ± 0.87	± 1.65	0.94	± 7.71
weaver2	7.53 ± 0.83	± 1.58	0.95	± 7.71

Table 9: Confidence intervals for the UBIS($\tilde{u}\cdot P$) estimates of Table 4.

7. NOISY USER FEEDBACK

Often in practice, one does not have a single, noise-free human expert, but multiple experts assigning labels to items. A typical example for this is crowd-sourcing, where labels get consolidated from multiple, probably less experienced, judges. However, even with a single expert, there is the possibility that he or she may not be always consistent. As we show, our framework naturally lends itself to noisy settings. To incorporate these effects in our model, we assume that we are getting noisy labels for each pair (x, y). We capture noise for each (x, y) by saying that we draw the judgements $u(x, y) \sim P_{J|x,y}$ and define

$$v(x, y) = \mathbb{E}_{u\sim P_{J|x,y}}[U].$$

This essentially states that we assume that there is some true rating defined as the average of some probability function. Similar to before, we assume $v(x, y) \in [0, 1]$. Our estimator (4) stays the same, since computing the expectation is linear with respect to sums. Also, our theoretical results for picking the optimal sampling distributions remain essentially unaltered. The only thing that changes is that we replace the true $u(x, y)$ by the true expected utility, $v(x, y)$.

8. RELATED WORK

Having incomplete judgments due to budget constraints has become an increasingly large problem in IR since document collections keep growing in size [5]. In terms of fixed deterministic evaluation strategies, there have been two main lines of work. The first group of schemes tries to judge only a subset of documents. Examples include pooling [15] where only the top m documents of each ranking are judged, or more sophisticated selection strategies (e.g., [7]). The second group of methods handles the problem of having incomplete judgments by defining new metrics, such as condensed versions of standard metrics [19] or other new metrics [2].

As we have shown, all fixed deterministic schemes are inevitably biased, and ignore incomplete information by deeming it not relevant. There have been have been efforts to correct for this bias post-hoc [3, 23] although most of the methods are fairly complicated, making them unattractive for practitioners.

Carterette et al. [4] describe an adaptive selection algorithm for telling two systems apart using average precision (AP). In each step, their approach picks a sample so as to maximize the knowledge about the differences of the AP values between two rankings. However, as this strategy is also deterministic; the judged samples cannot be re-used to get unbiased estimates for new systems.

Closest to our approach is the one of [24] and [17]. There, the authors propose to estimate AP using stratified sampling or importance sampling. The authors describe a heuristic scheme to do absolute evaluation of k systems and show the unbiasedness of their approach on the same TREC dataset. We not only introduced a variety of new scenarios, but also showed what the theoretically optimal sampling distribution is in each of the latter. Moreover, our framework is able to handle a broad class of metrics and can be easily applied to scenarios other than ranking evaluation.

One implicit assumption that most above-mentioned work has made is that the relevance scores we obtain via $u(x, y)$ are noise-free. One possibility to justify such an assumption is to heuristically rectify noisy user feedback prior to the input to the algorithm. Along these lines, there has been a lot of work that tries to consolidate noisy relevance scores from multiple judges (e.g., [9, 18]. However, as we have shown above, our algorithms can be extended in a natural way in order to deal with noisy judgements.

Somewhat related to our method is the scenario in which one wants to re-use interaction logs of a system for evaluation [11, 12]. Although these methods also employ importance sampling as a component, they assume that one has no control over the sampling distribution Q.

9. CONCLUSIONS AND FUTURE WORK

We introduced a simple and general framework for evaluating structured prediction systems. We empirically demonstrated the effectiveness of our framework on two datasets. By relying on importance sampling as a key technique, we naturally obtain unbiased estimates and make re-using previously obtained judgements easy and effective. We derived theoretically optimal sampling distributions for a number of different scenarios, justifying our estimation methods for relative and absolute performance evaluation of multiple systems. Additionally, we discussed how to obtain confidence intervals and how to integrate noisy judgements with our framework.

Our plan for the future is to extend this work in three directions. First, it would be interesting to see how to better approximate $u(x, y)$ by learning it from previously collected judgements. This could potentially also be done in an adaptive fashion – similar to adaptive importance sampling. After a couple of rounds of obtaining judgements, one would then update $\tilde{u}(x, y)$ with the new judgements and start another sampling epoch. Second, we would like to look at various other questions that come up with k systems, e.g., sampling so as to maximize the probability of finding the best out of k systems. Again, it might be possible to do this adaptively. Lastly, it is appealing to not use manual relevance judgments for evaluation, but to use ratings that were provided by the users (e.g., Netflix). However, these ratings were not logged under a $Q(x, y)$ that was controlled by us. We therefore would need to estimate the $Q(x, y)$ that generated the data in order to de-bias the log data. Such approaches have proven successful in counterfactual contextual bandit evaluation [20]. It is an open question how effectively this can be done in real-world settings.

10. REFERENCES

[1] J. A. Aslam, V. Pavlu, and E. Yilmaz. A statistical method for system evaluation using incomplete judgments. In *SIGIR*, pages 541–548, 2006.

[2] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *SIGIR*, pages 25–32, 2004.

[3] S. Büttcher, C. L. Clarke, P. C. Yeung, and I. Soboroff. Reliable information retrieval evaluation with incomplete and biased judgements. In *SIGIR*, pages 63–70. ACM, 2007.

[4] B. Carterette, J. Allan, and R. Sitaraman. Minimal test collections for retrieval evaluation. In *SIGIR*, pages 268–275, 2006.

[5] B. Carterette, V. Pavlu, E. Kanoulas, J. A. Aslam, and J. Allan. If I had a million queries. In *ECIR*, pages 288–300, 2009.

[6] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 2. Wiley, 2nd edition, 1971.

[7] J. Guiver, S. Mizzaro, and S. Robertson. A few good topics: Experiments in topic set reduction for retrieval evaluation. *TOIS*, 2009.

[8] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[9] M. Hosseini, I. J. Cox, N. Milic-Frayling, G. Kazai, and V. Vinay. On aggregating labels from multiple crowd workers to infer relevance of documents. In *ECIR*, pages 182–194, 2012.

[10] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *TOIS*, 20:41–48, 2002.

[11] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *WSDM*, pages 297–306, 2011.

[12] L. Li, J. Y. Kim, and I. Zitouni. Toward predicting the outcome of an A/B experiment for search relevance. In *WSDM*, 2015.

[13] Massart, Pascal and Picard, Jean. *Concentration inequalities and model selection : Ecole d'été de probabilités de Saint-Flour XXXIII, 2003*, Lecture Notes in Mathematics. Springer, 2007.

[14] A. Maurer and M. Pontil. Empirical bernstein bounds and sample-variance penalization. In *COLT*, 2009.

[15] R. Nuray and F. Can. Automatic ranking of retrieval systems in imperfect environments. In *SIGIR*, 2003.

[16] A. B. Owen. *Monte Carlo theory, methods and examples.* 2013.

[17] V. Pavlu and J. Aslam. A practical sampling strategy for efficient retrieval evaluation. Technical report, College of Computer and Information Science, Northeastern University, 2007.

[18] D. D. Peng Ye. Combining preference and absolute judgements in a crowd-sourced setting. In *ICML Workshop: Machine Learning Meets Crowdsourcing*, 2013.

[19] T. Sakai. Alternatives to bpref. In *SIGIR*, pages 71–78, 2007.

[20] A. Strehl, J. Langford, L. Li, and S. M. Kakade. Learning from logged implicit exploration data. In *NIPS*, pages 2217–2225. 2010.

[21] E. M. Voorhees and D. Harman. Overview of the ninth text retrieval conference (TREC-8). In *NIST Special Publication 500-246*, 1999.

[22] L. Wasserman. *All of statistics: a concise course in statistical inference.* Springer, 2004.

[23] W. Webber and L. A. Park. Score adjustment for correction of pooling bias. In *SIGIR*, pages 444–451, 2009.

[24] E. Yilmaz and J. A. Aslam. Estimating average precision with incomplete and imperfect judgments. In *CIKM*, pages 102–111, 2006.

[25] E. Yilmaz, E. Kanoulas, and J. A. Aslam. A simple and efficient sampling method for estimating AP and NDCG. In *SIGIR*, pages 603–610, 2008.

[26] C. Yuan and M. J. Druzdzel. How heavy should the tails be? In *FLAIRS*, pages 799–805, 2005.