Predicting Query Performance for Fusion-Based Retrieval

Gad Markovits
gadm@tx.technion.ac.il

Anna Shtok
annabel@tx.technion.ac.il

Oren Kurland
kurland@ie.technion.ac.il

Faculty of Industrial Engineering and Management, Technion — Israel Institute of Technology
Haifa 32000, Israel

David Carmel
IBM Research lab
Haifa 31905, Israel
carmel@il.ibm.com

ABSTRACT

Estimating the effectiveness of a search performed in response to a query in the absence of relevance judgments is the goal of query-performance prediction methods. Post-retrieval predictors analyze the result list of the most highly ranked documents. We address the prediction challenge for retrieval approaches wherein the final result list is produced by *fusing* document lists that were retrieved in response to a query. To that end, we present a novel fundamental prediction framework that accounts for this special characteristics of the fusion setting; i.e., the use of intermediate retrieved lists. The framework is based on integrating prediction performed upon the final result list with that performed upon the lists that were fused to create it; prediction integration is controlled based on inter-list similarities. We empirically demonstrate the merits of various predictors instantiated from the framework. A case in point, their prediction quality substantially transcends that of applying state-of-the-art predictors upon the final result list.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Retrieval models

General Terms: Algorithms, Experimentation

Keywords: query-performance prediction, fusion

1. INTRODUCTION

The effectiveness of retrieval methods often varies across queries [35, 15]. Thus, the ability to automatically infer which queries are more difficult for a retrieval method than others can be very important. These observations motivated a large body of work on predicting query performance [5]; that is, estimating the effectiveness of a search performed in response to a query in the absence of relevance judgments.

Pre-retrieval predictors [18, 16] analyze the query expression and utilize corpus-based statistics. By definition, then, their prediction is not based on the ranking at hand, the effectiveness of which is the goal of prediction. Post-retrieval predictors [9, 5], on the other hand, analyze also the *result list* of the most highly ranked documents. Hence, they often yield prediction quality that (substantially) transcends that of pre-retrieval predictors [5].

We address the challenge of post-retrieval query performance prediction for *fusion*-based retrieval [14, 8]. The goal is predicting the effectiveness of a document list that is produced by fusing (merging) a few lists that were retrieved, from the same corpus, in response to the query.

The motivation for addressing the challenge just mentioned is twofold. First, fusion methods post effective retrieval performance on average [8]; e.g., when fusing lists retrieved by using different query and document representations or retrieval methods [8]. Furthermore, although fusion-based retrieval was shown to somewhat improve retrieval-performance robustness across queries [2], we empirically show that commonly used fusion methods still suffer from substantial performance variance. Hence, predicting which queries are more difficult for fusion-based methods than others is motivated by the same reasons that drove forward work on query-performance prediction in general.

The second factor motivating the pursue of the query-performance prediction challenge for fusion-based retrieval is the unique characteristics of this retrieval paradigm. Specifically, we address the following question, which to the best of our knowledge, has not been been tackled before: *Can using information induced from the intermediate results of the fusion-based retrieval process (i.e., the document lists that are fused) in addition to that which is the focus of previous work on post-retrieval prediction (i.e., that induced from the final result list) help improve prediction quality?*

We present a novel probabilistic query-performance prediction framework for fusion-based retrieval that addresses the question stated above. The framework is based on integrating prediction performed upon the final result list, the effectiveness of which is the goal of prediction, with that employed over the lists that are fused; prediction integration is controlled by using inter-list similarity measures. The underlying idea is that lists that are similar — with respect to the documents they contain and their ranking — should have similar performance. We instantiate various predictors from the framework by varying the prediction-integration approach, the single-list predictors employed, and the inter-list similarity measure used.

Empirical evaluation performed with several tracks of TREC, and with various fusion methods applied to runs submitted to these tracks, attests to the merits of our prediction approach. For example, the prediction quality of the predictors we derive substantially transcends that of applying state-of-the-art post-retrieval predictors upon the final result list.

Our main contributions can be summarized as follows.

1. Previous work on post-retrieval prediction [5] focuses on analyzing the final — and single retrieved — result list. In contrast, our prediction framework targets the unique characteristics of the fusion-based retrieval process; specifically, by integrating prediction based on the final result list with that based on the intermediate lists fused to create it.

2. Empirically demonstrating the substantial merits of our prediction approach; specifically, with respect to the (standard) practice of applying prediction only upon the final result list. Furthermore, our approach is shown to be highly effective in predicting query performance for a variety of fusion methods.

2. RELATED WORK

There is a large body of work on query-performance prediction [5]; specifically, on devising post-retrieval predictors that analyze properties of the result list of top-retrieved documents. For example, clarity-based predictors [9] measure the divergence of a model induced from the result list from that of the corpus. Other prediction methods measure different notions of the robustness of the result list [37, 33, 39, 3, 40]. Some predictors analyze the retrieval scores of documents in the result list [13, 40, 27, 11, 12, 31].

The prediction methods mentioned above analyze the properties of a single retrieved list. In contrast, our prediction framework predicts the effectiveness of the final result list by analyzing both its properties and those of the lists that were fused to create it. We use adaptations of some previously proposed predictors [9, 40, 31] for single-list-based prediction in our framework.

Query-performance predictors were used to improve fusion effectiveness [4, 28]. Specifically, predictors were applied upon the lists that are fused so as to weigh them in the fusion process. In contrast, our goal is to predict the effectiveness of the final document list that is the result of fusion.

The similarity between the final result list and the intermediate lists fused to create it was used as a predictor in its own right for the effectiveness of the intermediate lists [32, 3, 13]. In contrast, our methods predict the effectiveness of the final result list; specifically, by using the inter-list similarities to control the integration of prediction applied upon the intermediate lists and upon the final result list.

Some recent work [20] showed that several post-retrieval predictors [9, 13, 40, 30, 31] essentially rely on the following principle. Given a result list, the effectiveness of which we want to predict, a pseudo effective and/or ineffective "reference" document lists are created; then, the similarity between the reference lists and the result list at hand serves for prediction. In contrast, our goal is to predict the effectiveness of a result list that was produced by fusing several given result lists. To that end, we propose a novel probabilistic prediction framework wherein inter-list similarities serve a specific role.

There is recent work [21] that derives the prediction principle mentioned above, of using reference lists, in a probabilistic manner. In contrast to our approach, prediction applied directly upon the final result list, which we empirically show to be highly important for the fusion setting we address here, is not integrated in the proposed model. In fact, the fundamental predictor we present, which serves for deriving various predictors, can be viewed as a generalized form of the reference-list-based prediction model presented in this work [21].

3. PREDICTION FRAMEWORK

Throughout this section we assume that the following have been fixed. A query q, a corpus of documents $\mathcal{C}$; and, m document lists, $L_1, \ldots, L_m$, each of which contains n documents that were retrieved from $\mathcal{C}$ by *some* retrieval method in response to q.

Suppose that *some* fusion method [8] was employed over the retrieved lists to produce a final result list, L_{res}, of n documents. Our goal is to predict the effectiveness of L_{res} with respect to the information need expressed by q in the absence of relevance judgments.

The query-performance prediction framework that we present does not assume knowledge of the fusion method, nor knowledge about the retrieval methods used to create the lists $L_1, \ldots, L_m$. Hence, the framework is quite general and yields high quality prediction for various fusion methods as we show in Section 4. Yet, in Section 3.5 we discuss the connection between the prediction framework and a specific family of fusion approaches.

The prediction framework is based on utilizing *some* post-retrieval query-performance predictor $\mathcal{P}_{base}$. The predictor $\mathcal{P}_{base}$ operates upon q and a document list L that was retrieved in response to q; corpus-based information can also be utilized. The prediction value assigned by $\mathcal{P}_{base}$, denoted $\mathcal{P}_{base}(L;q)$, is an estimate for the effectiveness of L with respect to q. In Section 3.4 we discuss several adaptations of previously proposed predictors that we use for $\mathcal{P}_{base}$.

We can state the prediction goal as that of estimating, in the absence of relevance judgments, $p(L_{res}|q)$, i.e., the probability that L_{res} satisfies the information need expressed by q. The resultant estimate reflects the presumed effectiveness of L_{res}. Following previous work on post-retrieval query-performance prediction [5], we can apply $\mathcal{P}_{base}$ *directly* upon L_{res}, so as to derive an estimate for $p(L_{res}|q)$. This is the **Direct** predictor:

$$\mathcal{P}_{Direct}(L_{res};q) \overset{def}{=} \mathcal{P}_{base}(L_{res};q). \qquad (1)$$

However, we would like to exploit the special characteristics of the fusion-based retrieval setting to potentially improve prediction quality; that is, utilize also information induced from the lists $L_1, \ldots, L_m$ that were fused to produce L_{res}. A case in point, if many of these lists are predicted to be highly effective, and L_{res} is not very different — in the documents it contains and their ranking — from these lists, then the effectiveness predicted for L_{res} should be high.

Accordingly, we use the lists $L_1, \ldots, L_m$ as proxies for L_{res} in estimating its effectiveness. Specifically, we can write

$$p(L_{res}|q) = \sum_{L_i} p(L_{res}|q, L_i)p(L_i|q). \qquad (2)$$

Equation 2 is based on the premise that the effectiveness of L_{res} can be predicted (estimated) using a mixture of estimates (predictions) used for the effectiveness of the lists (L_i) that were fused to create it. In what follows we further develop Equation 2 and present a variety of predictors that can be derived based on this premise. Later on, specifically in Section 3.5, we discuss the formal connections between some of the derived predictors and a specific family of fusion methods (namely, linear models). Yet, we hasten to point out that in Section 4 we show that the proposed predictors post high prediction quality for a variety of fusion approaches, among which is a highly effective non-linear method.

We estimate $p(L_{res}|q, L_i)$ in Equation 2 using a linear mixture governed by a free parameter λ: $(1-\lambda)\hat{p}(L_{res}|q) + \lambda\hat{p}(L_{res}|L_i)$;[1] here and after, $\hat{p}(x)$ denotes an estimate for $p(x)$. Applying some probabilistic algebra, and assuming that $\hat{p}(L_i|q)$ is a probability distribution over the lists L_i (i.e., $\sum_{L_i} \hat{p}(L_i|q) = 1$), yields the following generic prediction approach

$$\mathcal{P}_{Gen}(L_{res}; q) \stackrel{def}{=} (1-\lambda)\hat{p}(L_{res}|q) + \lambda \sum_{L_i} \hat{p}(L_{res}|L_i)\hat{p}(L_i|q); \tag{3}$$

$\hat{p}(L|q)$ is an estimate for the probability that the list L satisfies the information need expressed by q. Thus, the resultant prediction for the effectiveness of L_{res} is based on integrating a direct estimate for its effectiveness ($\hat{p}(L_{res}|q)$) with estimates for the effectiveness of the lists L_i from which it was fused ($\hat{p}(L_i|q)$); these estimates are weighted by L_i's "association" with L_{res} ($\hat{p}(L_{res}|L_i)$).

3.1 Deriving specific predictors

To instantiate specific prediction methods from Equation 3, we use the prediction value $\mathcal{P}_{base}(L; q)$ for the estimate $\hat{p}(L|q)$. To estimate the association between L_{res} and L_i ($\hat{p}(L_{res}|L_i)$), we use an inter-list similarity measure, $sim(\cdot, \cdot)$. We discuss two similarity measures in Section 3.2.[2] The resultant **AMean** predictor instantiated from Equation 3, which is based on an arithmetic mean of (weighted) prediction values, is defined as

$$\mathcal{P}_{AMean}(L_{res}; q) \stackrel{def}{=} (1-\lambda)\mathcal{P}_{base}(L_{res}; q) + \lambda \sum_{L_i} sim(L_i, L_{res})\mathcal{P}_{base}(L_i; q).$$

Setting $\lambda = 0$ amounts to the Direct predictor from Equation 1. Higher values of λ result in increased importance attributed to prediction based on the lists L_i.

As an alternative to the arithmetic-mean-based approach employed by AMean, we explore the **GMean** predictor that is based on a *form* of a geometric mean:[3]

$$\mathcal{P}_{GMean}(L_{res}; q) \stackrel{def}{=} \mathcal{P}_{base}(L_{res}; q)^{(1-\lambda)} \Big(\prod_{L_i} sim(L_i, L_{res})\mathcal{P}_{base}(L_i; q)\Big)^{\lambda}.$$

Consequently, if L_{res} or any of the lists L_i is predicted by $\mathcal{P}_{base}$ to be of very low effectiveness, the resultant prediction value is lower for GMean than for AMean as GMean is more "conservative". Similar considerations motivated the use of the GMAP effectiveness evaluation metric [36] (i.e., the geometric mean of per-query average precision values) as a conservative alternative to MAP (the arithmetic mean).

3.2 Inter-list similarity measures

There is a wide variety of inter-list similarity measures that can be used in the prediction methods presented above. Here, we study two measures that attribute more importance to documents at high ranks of the lists than to those at low ranks. This is because retrieval effectiveness, the goal of prediction, is often measured, as will be the case in Section 4, using metrics that are heavily affected by top ranks (e.g., average precision). The first inter-similarity measure that we use, **KL**, utilizes rank information, is asymmetric, and only considers the most highly ranked documents in the list. The second measure, **Cosine**, uses retrieval scores, is symmetric, and considers all documents in the list.

The KL measure. Let $\mathcal{C}_{init} \stackrel{def}{=} \bigcup_i L_i$ denote the *set* of documents in the lists that are fused. We use $p_L(\cdot)$ to denote the representation of L as a probability distribution over $\mathcal{C}_{init}$. To induce $p_L(\cdot)$, we follow a previous proposal [3]. Specifically, let d_r ($\in L$) be the document at rank r of L. ($r = 1$ is the highest rank.) Then, $p_L(d_r) \stackrel{def}{=} \frac{1}{2c}(1 + \frac{1}{r} + \frac{1}{r+1} + \cdots + \frac{1}{c})$, where c is a cutoff parameter. For d that is at rank r of L with $r > c$, or that is not in L, $p_L(d)$ is set to 0. The similarity between lists L_i and L_j is defined based on the asymmetric KL divergence measure: $sim_{KL}(L_i, L_j) \stackrel{def}{=} \exp(-KL\left(p_{L_i}(\cdot) \;\middle|\middle|\; p_{L_j}(\cdot)\right))$ [3].

The Cosine measure. The second inter-list similarity measure is based on a vector-space representation of L, denoted $\vec{L}$, defined over $\mathcal{C}_{init}$. Specifically, the l'th component of the vector $\vec{L}$ is set to the score of document d_l ($\in \mathcal{C}_{init}$) if d_l appears in L and to 0 otherwise. The cosine is used as a symmetric inter-list similarity measure: $sim_{Cosine}(L_i, L_j) \stackrel{def}{=} \cos(\vec{L_i}, \vec{L_j})$.

3.3 A uniform inter-list similarity measure and alternative prediction-aggregation methods

To study the merits of differentially weighting the prediction values assigned to the lists L_i based on their similarity with L_{res}, we devise variants of the AMean and GMean

[1] A somewhat conceptually similar estimation technique, and consequent derivation, were used in work on cluster-based retrieval [19].

[2] Normalizing the similarity measure to yield a probability distribution showed no empirical merit in terms of prediction quality. We also note that the base predictors used in Section 3.4, as the vast majority of previously proposed predictors [5], do not induce probability distributions. Hence, while our framework is probabilistic, it is instantiated using estimates that are not probability distributions.

[3] GMean is the geomtric mean of the prediction value assigned to L_{res} and the *weighted* prediction values assigned to the lists L_i. We do not use the geometric mean of the latter (i.e., use the inter-list similarity values for the exponent of the prediction values) as doing so turned out to yield worse prediction quality than that attained by GMean as defined here. We also note that GMean cannot be directly derived from Equation 3 as opposed to AMean.

methods that use a uniform inter-list similarity measure; i.e., the similarity between L_i and L_{res} is not accounted for. Specifically, we study the **UniAMean** predictor:

$$\mathcal{P}_{UniAMean}(L_{res};q) \overset{def}{=} (1-\lambda)\mathcal{P}_{base}(L_{res};q) + \frac{\lambda}{m}\sum_{L_i}\mathcal{P}_{base}(L_i;q); \quad (4)$$

and, the **UniGMean** predictor[4]:

$$\mathcal{P}_{UniGMean}(L_{res};q) \overset{def}{=} \mathcal{P}_{base}(L_{res};q)^{(1-\lambda)}\prod_{L_i}\mathcal{P}_{base}(L_i;q)^{\frac{\lambda}{m}}; \quad (5)$$

m is the number of lists.

The methods presented above assign a prediction value that is based on the aggregation of the prediction values for all the lists that are fused. To consider the alternative of using the prediction for only "representative" lists, we upper-bound the prediction value assigned by UniAMean in Equation 4 using the **UniMax** predictor:

$$\mathcal{P}_{UniMax}(L_{res};q) \overset{def}{=} (1-\lambda)\mathcal{P}_{base}(L_{res};q)+\lambda\max_{L_i}\mathcal{P}_{base}(L_i;q); \quad (6)$$

and lower-bound it using the **UniMin** predictor:

$$\mathcal{P}_{UniMin}(L_{res};q) \overset{def}{=} (1-\lambda)\mathcal{P}_{base}(L_{res};q)+\lambda\min_{L_i}\mathcal{P}_{base}(L_i;q). \quad (7)$$

3.4 Base predictors

The last step for instantiating predictors using the methods described above is defining the predictor $\mathcal{P}_{base}$ that operates on a document list L. However, most previously proposed post-retrieval predictors were devised for specific retrieval methods [5]; in fact, most state-of-the-art predictors were shown to be effective when the retrieval was based on surface-level document-query similarities (e.g., [9, 13, 40, 17, 30, 31]). Here, we have to employ $\mathcal{P}_{base}$ upon lists $L_1, \ldots, L_m$ that were retrieved by *some* retrieval methods that may not be those for which previous predictors were designed for; and, upon L_{res} which is created using a fusion method. Thus, we use some recently proposed adaptations [31] of three (highly) effective post-retrieval predictors that operate on a list (L) retrieved by *some* method. In what follows we use $s(d;L)$ to denote the (non-negative) score of document d in L. For the lists L_i that are fused, we normalize the original retrieval scores so they sum to 1; $s(d;L)$ is the normalized retrieval score in this case. (See Section 4.1 for further details.) The predictors we present focus on $L^{[k]}$, the k highest ranked documents in L; k is a free parameter.

Clarity. Clarity measures the focus of a retrieved list L with respect to the corpus by computing the divergence between their induced language models [9]. The higher the divergence, the more distant the models are; and, L is presumed to be more effective.

The (relevance) language model utilized by Clarity, which is induced from L, is based on a language-model-based retrieval used to create L. Here, we use Clarity for the general retrieved list case. Let $w(d;L)$ be a weight assigned to document d with respect to a list L in which it appears. Then, the relevance model $R_{L^{[k]}}$ induced from $L^{[k]}$ is defined by: $p(t|R_{L^{[k]}}) \overset{def}{=} \sum_{d\in L^{[k]}} p(t|d)\frac{w(d;L)}{\sum_{d'\in L^{[k]}} w(d';L)}$; t is a term in the vocabulary; $p(t|d)$ is the probability assigned to t by a language model induced from d. (Language model induction details are provided in Section 4.1.) Clarity is the KL divergence between the relevance model and the (unsmoothed) corpus language model ($p(\cdot|\mathcal{C})$):

$$Clarity(q;L) \overset{def}{=} KL\left(p(\cdot|R_{L^{[k]}}) \,\middle|\middle|\, p(\cdot|\mathcal{C})\right). \quad (8)$$

We use two variants of Clarity. The first, **sClarity**, weighs document d by its retrieval score in L: $w_{sClarity}(d;L) \overset{def}{=} s(d;L)$. The **rClarity** measure, as in previous proposals [10, 13], weighs d using a linearly decreasing function of its rank r ($\leq k$) in L; specifically, $w_{rClarity}(d;L) \overset{def}{=} 2(k+1-r)$.

WIG. The WIG predictor is based on the premise that high retrieval scores at top ranks of the list, specifically, with respect to that of the corpus, indicate effective retrieval [40]. WIG was proposed in the markov random field (MRF) retrieval framework [25]; and, was also shown to be effective for predicting the effectiveness of a language-model-based retrieval [30]. Here, for the case of a list retrieved by *some* method, we use for prediction the mean of retrieval scores of the highest ranked documents [31][5]:

$$WIG(q;L) \overset{def}{=} \frac{1}{k}\sum_{d\in L^{[k]}} s(d;L). \quad (9)$$

We note that the corpus retrieval score cannot be used, as opposed to the original implementation of WIG [40], since we do not have knowledge of the retrieval method used to create L; or, it might be created by fusion as is the case for L_{res}. While the corpus retrieval score originally served to ensure inter-query compatibility of prediction values due to lack of inter-query compatibility of retrieval scores, here retrieval scores in the lists L_i are sum-normalized to $[0,1]$.[6]

NQC. The NQC predictor measures the standard deviation of retrieval scores of top-retrieved documents [31]. It was argued that increased standard deviation amounts to potentially reduced *query drift* in the result list; and hence, implies improved retrieval effectiveness [31]. While the argument was based on the assumption that retrieval scores reflect document-query surface-level similarities, we use NQC for a list retrieved by *some* method [31]. Specifically, let

[4]UniGMean is the geometric mean of (i) the prediction value assigned to L_{res}, and (ii) the geometric mean of the prediction values assigned to the lists L_i.

[5]The original WIG measure also employs query-length normalization [40]. This normalization is important for inter-query compatibility of retrieval scores in the MRF and language modeling frameworks. We found that such normalization hurts the prediction quality when using lists retrieved by *some* method (e.g., TREC runs) as is the case here.

[6]Sum-normalization of the fusion scores in L_{res} degraded prediction quality for the different predictors, and hence was not used here and after.

$\mu \overset{def}{=} \frac{1}{k}\sum_{d\in L^{[k]}} s(d;L)$ be the mean score in $L^{[k]}$, then[7]

$$NQC(L;q) \overset{def}{=} \sqrt{\frac{1}{k}\sum_{d\in L^{[k]}}(s(d;L)-\mu)^2}. \qquad (10)$$

3.5 On the connection between the prediction framework and (specific) fusion methods

Suppose that the final result list, L_{res}, contains a single document, d. Then, using Equation 3 with $\lambda = 1$ yields

$$\mathcal{P}_{Gen}(d;q) \overset{def}{=} \sum_{L_i} \hat{p}(d|L_i)\hat{p}(L_i|q). \qquad (11)$$

That is, the predicted (estimated) relevance of d to q is determined based on d's association with the lists L_i ($\hat{p}(d|L_i)$), wherein the association is weighted by the predicted effectiveness of L_i ($\hat{p}(L_i|q)$). If $\hat{p}(d|L_i)$ is based on d's retrieval score (or rank) in L_i, then we attain the general form of linear fusion functions [14, 34]. For example, the commonly used CombSUM method [14], which is one of the fusion approaches used for the evaluation presented in Section 4, is based (in spirit) on using d's normalized score in L_i for $\hat{p}(d|L_i)$ and a uniform estimate for $\hat{p}(L_i|q)$ — i.e., considering the L_i's to be equi-effective.

Thus, we see that the proposed prediction framework can conceptually be viewed as a generalization (including a geometric-mean-based form), and adaptation, of the basic linear fusion principle to the prediction task. Yet, the framework utilizes not only information induced from the lists that are fused, as is the (natural) case in the fusion-based retrieval process, but also information from the final result list (L_{res}); specifically, if $\lambda < 1$. (See Equation 3.) This is an important fundamental difference, among others, that rises due to the inherent difference between the prediction and fusion (retrieval) tasks.

4. EVALUATION

4.1 Experimental setup

For experiments we use the ad hoc tracks of TREC3 and TREC8, the Web tracks of TREC9 and TREC10, and the robust track of TREC12. Thus, a variety of query sets and corpora (newswire and Web) serves for evaluation.

We randomly sample $m = 5$ runs from *all* those submitted to a track and that contain at least 100 documents as results for each query. The $n = 100$ highest ranked documents for a query in a run serve as one of the 5 lists L_i to be fused[8]. We repeat this runs-sampling process 20 times and report the average prediction quality of each predictor over the samples. Statistically significant differences of prediction quality are determined using the two-tailed paired t-test computed at a 95% confidence level with respect to the 20 samples.

To measure prediction quality, we follow common practice [5]. That is, we compute Pearson's correlation between the values assigned by a predictor to the queries in a track, and the true average precision (AP at cutoff 100) values for these queries — determined using TREC's relevance judgments — attained by the fusion method at hand.

Fusion methods. Fusion methods that utilize retrieval scores require inter-list score compatibility. Therefore, we normalize the retrieval score of a document in a list with respect to the sum of all scores in the list. As before, we use $s(d;L_i)$ to denote the resultant normalized retrieval score of document d in L_i; if $d \notin L_i$, we set $s(d;L_i) \overset{def}{=} 0$. In what follows, $F_x(d)$ denotes the score assigned to d by a fusion method x.

The **CombSUM** linear fusion method [14] scores d by the sum of its retrieval scores: $F_{CombSUM}(d) \overset{def}{=} \sum_{L_i} s(d;L_i)$. **CombMNZ** [14, 22], which is a non-linear fusion method, further rewards documents that appear in many lists: $F_{CombMNZ}(d) \overset{def}{=} \#\{L_i : d \in L_i\}F_{CombSUM}(d)$.

The **Borda** method [38], in contrast to CombSUM and CombMNZ, considers only rank information. Specifically, d is scored by the number of documents not ranked higher than it in the lists. Formally, if $d \in L$, we use $r(d;L)$ to denote d's rank in L. (The rank of the top most document is 1.) Then, d's Borda score is: $F_{Borda}(d) \overset{def}{=} \sum_{L_i:d\in L_i} \#\{d' \in L_i : r(d';L_i) > r(d;L_i)\}$. The **RRF** fusion method [6], which was shown to be effective, also relies only on ranks: $F_{RRF}(d) \overset{def}{=} \sum_{L_i:d\in L_i} \frac{1}{\nu + r(d;L_i)}$, where $\nu = 60$ [6].

Base predictors, free parameters, and reference comparisons. We follow previous recommendations with regard to effective free-parameter settings for the Clarity predictor [30]. Specifically, for all predictors that use the rClarity and sClarity measures, we set k, the cutoff at which the relevance model is computed, to $n = 100$, the number of documents in the list; the language models of documents from which the relevance model is constructed are not smoothed; and, the number of terms used is 100. For predictors that use the WIG and NQC measures, k, the number of most highly ranked documents considered by these measures, is set to a value in $\{5, 10, 25, 50, 100\}$.[9]

The interpolation parameter λ (see Equation 3) is set to a value in $\{0, 0.1, \ldots, 1\}$. The cutoff parameter c used for computing inter-list similarity with the KL approach is set to 20 following previous recommendations [3].

For predictors — both those proposed here and the reference comparisons — that incorporate free parameters, i.e., k and/or λ, we first set the free parameters to values optimizing prediction quality over the query set for a track. Thus, we start by using all prediction methods with their free parameters set to (generally) effective values. This practice enables to study the influence on prediction quality of various aspects (e.g., the inter-list similarity measure used, and the approach employed for aggregating prediction values) when the effect of free-parameter values is ameliorated. Then, in

[7]As is the case for WIG, the original NQC measure normalizes retrieval scores with respect to that of the corpus [31]. In contrast, here we cannot use corpus-based normalization of retrieval scores, but the scores in the lists L_i are sum normalized.

[8]We use 100 rather than 1000 documents as results for each query, as fusion methods are most effective when the number of documents in the lists that are fused is relatively small [34]. We use the 100 documents most highly ranked by the fusion method for the final result list L_{res}.

[9]NQC's prediction of the effectiveness of a *TREC run* is better when retrieval scores are not normalized, while that of WIG is better when retrieval scores are normalized [31]. However, as some of the fusion methods employed require score normalization, retrieval scores in the runs are always normalized as described above.

	TREC3		TREC8		TREC9		TREC10		TREC12	
	MAP	σ(AP)	MAP	σ(AP)	MAP	σ(AP)	MAP	σ(AP)	MAP	σ(AP)
CombSUM	.195	.153	.244	.218	.176	.165	.175	.151	.242	.217
CombMNZ	.196	.153	.241	.214	.182	.175	.175	.149	.239	.215
Borda	.189	.152	.235	.211	.188	.182	.180	.156	.243	.218
RRF	.197	.155	.238	.212	.189	.182	.182	.155	.243	.218
Best Run	.209	.165	.252	.223	.192	.189	.164	.142	.240	.221
Worst Run	.068	.078	.100	.126	.066	.091	.093	.116	.140	.169

Table 1: Retrieval performance of the fusion methods: the mean of AP (MAP) values, and their standard deviation (σ(AP)), computed over the queries in a track. The best and worst runs are selected from the 5 that are fused according to MAP. Numbers represent averages over the 20 samples of 5 runs.

Section 4.2.5 we perform an in-depth study of the effect of free-parameter values on prediction quality. In Section 4.2.6 we present the prediction quality of our methods, and that of the reference comparisons, when free-parameter values are set using a train set of queries.

The Direct predictors, which manifest the common practice of applying post-retrieval predictors upon the final result list, serve as our reference comparisons. This comparison sheds light on whether integrating prediction employed upon the lists that are fused with prediction performed upon the final result list — the underlying idea of our approach — can improve over using only the latter.

Analysis of retrieval performance. As noted in Section 1, the main factor driving the development of query-performance prediction methods is the significant variability of retrieval performance across queries [35, 15]. We now turn to study this variability for the fusion methods. In comparison, we present the variability for the best and worst (MAP) performing runs among the 5 fused. Specifically, we report the MAP, which is the mean of the average precision (AP) values attained for the queries, and, their standard deviation. The numbers are reported in Table 1 and represent averages over the 20 samples of 5 runs.

We see that in most cases, both the MAP and the standard deviation of AP posted by the fusion methods are somewhat lower than those posted by the best-performing run. Yet, they are much higher than those posted by the worst-performing run, which can be quite ineffective as suggested by its MAP. Furthermore, the standard deviation of AP for the fusion methods, as well as for the best run, is very high. These results show that fusion methods, as other retrieval approaches, post performance that can substantially vary across queries.

In the evaluation presented below we focus on predicting the query-performance of CombMNZ. CombMNZ is a standard baseline in many reports on fusion approaches [22, 23, 1, 26, 24, 29]. In Section 4.2.4 we study the effectiveness of our predictors for the other fusion methods.

4.2 Experimental results

4.2.1 Main result

Our first order of business is studying the general effectiveness of our prediction framework; specifically, that of the two main prediction methods, AMean and GMean. To that end, and unless otherwise specified, we use the KL inter-list similarity measure, as it will be shown in Section 4.2.2 to outperform the Cosine measure. As noted above, we focus on predicting the query-performance of the CombMNZ fusion method. Table 2 presents our main result. The numbers in Table 2, and those in all the following tables and graphs, represent prediction quality measured using Pearson's correlation. (Refer back to Section 4.1 for details.)

$\mathcal{P}_{base}$	Predictor	TREC3	TREC8	TREC9	TREC10	TREC12
WIG	Direct	.586	.642	.385	.444	.570
	AMean	**.664**^{d}	.687	$\underline{\mathbf{.465}}^d$	$\underline{\mathbf{.501}}^d$	$\underline{\mathbf{.634}}^d$
	GMean	.659^d	$\underline{\mathbf{.711}}^d$	.446^d	.485^d	.626^d
NQC	Direct	.549	.630	.352	.446	.555
	AMean	.684^d	.630	.364^d	.446	.570
	GMean	$\underline{\mathbf{.685}}^d$	**.673**^{d}	**.416**^{d}	**.446**	**.582**^{d}
sClarity	Direct	.484	.423	.089	.054	.385
	AMean	**.586**^{d}	**.614**^{d}	**.290**^{d}	**.316**^{d}	**.543**^{d}
	GMean	.576^d	.595^d	.211^d	.251^d	.518^d
rClarity	Direct	.479	.357	.072	.027	.332
	AMean	**.608**^{d}	**.602**^{d}	**.287**^{d}	**.326**^{d}	**.541**^{d}
	GMean	.594^d	.578^d	.211^d	.256^d	.506^d

Table 2: Main result table. The prediction quality of the AMean and GMean methods. Prediction quality, here and after, is measured by Pearson correlation with the ground-truth retrieval performance. (Refer to Section 4.1 for details.) CombMNZ serves for the fusion method, and the KL inter-list similarity measure is used. The best result per track and a base predictor is boldfaced. Underline marks the best result for a track; 'd' marks a statistically significant difference with Direct.

We first observe in Table 2 that when using the base predictors alone upon the final result list (Direct), WIG is the best-performing method. NQC outperforms both variants of Clarity, which can be quite ineffective for the Web tracks (TREC9 and TREC10). Yet, even in these cases, using the Clarity measures in our AMean and GMean methods can yield relatively effective prediction. Recall that, originally, WIG [40] and NQC [31] were devised for retrieval methods utilizing surface-level document-query similarities. Here we see that their adapted variants, which were discussed in Section 3.4, yield relatively high prediction quality when employed upon the final result list (Direct) that is the result of fusion. This finding is novel to this study.

Perhaps the most important observation based on Table 2 is that the AMean and GMean predictors outperform the Direct predictor in a vast majority of the relevant comparisons (base predictor × track); most of the improvements are substantial and statistically significant. This finding attests to the merits of the underlying idea of our prediction framework — integrating prediction based on the final result list (Direct) with that based on the intermediate lists that are fused to create it. In comparing AMean and GMean we see that the former is more effective than the latter when using

$\mathcal{P}_{base}$	Predictor	TREC3	TREC8	TREC9	TREC10	TREC12
WIG	Direct	.586	.642	.385	.444	.570
	UniAMean	$\mathbf{.711}^d$	.642	$.396^d$	.452	$.582^d$
	AMean (KL)	$.664^d_u$	**.687**	$\underline{\mathbf{.465}}^d_u$	$\underline{\mathbf{.501}}^d_u$	$\underline{\mathbf{.634}}^d_u$
	AMean (Cos)	$.655^d_u$	$.682^d_u$	.389	.448	$.580^d$
	UniGMean	$\underline{\mathbf{.717}}^d$	$.681^d$	$.423^d$	$.460^d$	$.592^d$
	GMean (KL)	$.659^d_u$	$\underline{\mathbf{.711}}^d$	$\mathbf{.446}^d_u$	$\mathbf{.485}^d_u$	$\mathbf{.626}^d_u$
	GMean (Cos)	$.594_u$	$.666^d$	$.414^d$	.451	$.574_u$
NQC	Direct	.549	.630	.352	.446	.555
	UniAMean	$.655^d$	.630	.356	.446	.560
	AMean (KL)	$.684^d_u$	.630	$\mathbf{.364}^d_u$	**.451**	**.570**
	AMean (Cos)	$\mathbf{.711}^d_u$	**.645**	.352	.446	.561
	UniGMean	$\mathbf{.695}^d$	$.666^d$	$.402^d$	**.453**	$\mathbf{.583}^d$
	GMean (KL)	$.685^d$	$.673^d$	$\mathbf{.416}^d$	.446	$.582^d$
	GMean (Cos)	$.667^d_u$	$\mathbf{.678}^d$	$.391^d$	.446	$.571^d_u$

Table 3: Prediction quality of using the various inter-list similarity measures. ('Cos' stands for 'Cosine'.) The best result per track in a AMean/GMean block is boldfaced; the best result per track is underlined. 'd' and 'u' mark statistically significant differences with Direct and with using a uniform similarity measure (the Uni predictors), respectively. CombMNZ is the fusion method.

WIG and Clarity for base predictors, while the reverse holds when using NQC.

Using the base predictors in AMean and GMean yields relative prediction-quality patterns similar to those of using these alone upon the final result list (Direct); that is, WIG (almost always) outperforms NQC which outperforms both Clarity variants. Hence, in the evaluation below we mainly focus on using WIG and NQC for base predictors. Indeed, the best prediction quality for a track in Table 2 is always attained by either a WIG-based or a NQC-based predictor. Specifically, using WIG in AMean yields the best prediction quality for 3 out of the 5 tracks.

4.2.2 *Inter-list similarity measures*

We next compare the prediction quality of using the KL, Cosine, and uniform inter-list similarity measures. Table 3 reports the prediction quality numbers.

We can see in Table 3 that in most relevant comparisons (base predictor × track × AMean/GMean) the KL measure yields prediction quality that is better than that of using the Cosine and uniform measures. (The main exception is the comparison with UniGMean for NQC.) Furthermore, KL yields more statistically significant improvements over Direct. The superiority of KL over the uniform measure, which is in quite a few cases (specifically, for WIG) statistically significant, attests to the merits of differentially weighting the prediction values for the lists that are fused based on their similarity to the final result list.

We also see in Table 3 that using WIG in AMean, with the KL measure, yields prediction quality that is, in general, highly effective with respect to that of the other predictors; specifically, the prediction quality is the best for 3 out of the 5 tracks as was the case in Table 2.

The Cosine measure is often less effective than the uniform measure; yet, both yield prediction quality that transcends that of Direct in almost all cases; sometimes, the improvements are statistically significant. All in all, these findings demonstrate the effectiveness of our framework when applied with different inter-list similarity measures.

$\mathcal{P}_{base}$	Predictor	TREC3	TREC8	TREC9	TREC10	TREC12
WIG	Direct	.586	.642	.385	.444	.570
	UniAMean	$.711^d$	.642	$.396^d$	.452	$.582^d$
	UniGMean	$\underline{\mathbf{.717}}^d$	$\underline{\mathbf{.681}}^d$	$\underline{\mathbf{.423}}^d$	$\underline{\mathbf{.460}}^d$	$\underline{\mathbf{.592}}^d$
	UniMin	$.590^d$	.642	.390	.446	.571
	UniMax	$.610^d$	.652	.385	.445	.580
NQC	Direct	.549	.630	.352	.446	.555
	UniAMean	$.655^d$	.630	.356	.446	.560
	UniGMean	$\mathbf{.695}^d$	$\mathbf{.666}^d$	$\mathbf{.402}^d$	.453	$\mathbf{.583}^d$
	UniMin	$.612^d$	.643	.391	**.455**	.559
	UniMax	$.584^d$	.630	.352	.446	.555

Table 4: Using the minimal and maximal prediction values (UniMin and UniMax) vs. using the mean of all prediction values (UniGMean and UniAMean). Boldface: the best result per track in a block of a base predictor. Underline: the best result per track. 'd' marks a statistically significant difference with Direct. CombMNZ serves for the fusion method.

4.2.3 *Comparing prediction-aggregation methods*

Insofar, we focused on two approaches for aggregating the prediction values assigned to the lists that are fused and to the final result list; i.e., those based on the arithmetic (AMean) and geometric (GMean) mean. We now compare these with the methods that integrate the prediction value assigned to the final result list with the minimal and maximal prediction values for the lists that are fused (UniMin and UniMax, respectively). As UniMin and UniMax utilize the uniform similarity measure, we use UniAMean and UniGMean for the comparison.

We see in Table 4 that UniMin and UniMax outperform the Direct predictor in several cases. However, these predictors are almost always outperformed — often, quite substantially — by AMean and GMean. This exemplifies the benefit in utilizing the mean prediction value for the lists that are fused with respect to using the minimal and maximal prediction values.

4.2.4 *Prediction for additional fusion methods*

The evaluation presented thus far focused on predicting query performance for the CombMNZ fusion method. In Table 5 we present the prediction quality of our AMean and GMean predictors for the CombSUM, Borda, and RRF fusion methods in addition to CombMNZ. (Refer back to Section 4.1 for details.) WIG and NQC serve as the base predictors and KL serves as the inter-list similarity measure.

Table 5 shows that AMean and GMean outperform the Direct predictors in most relevant comparisons (base predictor × track × fusion method); specifically, across fusion methods. The best prediction quality (boldfaced numbers) attained for Borda and RRF, which use rank information for fusion, is almost always higher than that obtained for CombSUM and CombMNZ, which utilize retrieval scores.

Thus, we see that our prediction approach is effective for different fusion methods whether they are based on retrieval scores (CombSUM and CombMNZ) or ranks (Borda and RRF) and whether they are linear (CombSUM, Borda and RRF) or not (CombMNZ). Furthermore, we see that the conceptual connection between our prediction approach and linear fusion methods, which was discussed in Section 3.5, does not necessarily imply that prediction quality is better for linear fusion methods. For example, the best prediction

		TREC3				TREC8				TREC9			
$\mathcal{P}_{base}$	Predictor	CombMNZ	CombSUM	Borda	RRF	CombMNZ	CombSUM	Borda	RRF	CombMNZ	CombSUM	Borda	RRF
WIG	Direct	.586	.613	.582	.570	.642	.638	.564	.560	.385	.403	.414	.420
	AMean	**.664**^{d}	**.665**^{d}	**.656**^{d}	**.662**^{d}	.687	.682	.689^d	.687^d	**.465**^{d}	**.481**^{d}	.461^d	.466^d
	GMean	.659^d	**.665**^{d}	.652^d	.657^d	**.711**^{d}	**.707**^{d}	**.716**^{d}	**.714**^{d}	.446^d	.469^d	**.489**^{d}	**.493**^{d}
NQC	Direct	.549	.597	.584	.544	.630	.594	.596	.588	.352	.309	.418	.421
	AMean	.684^d	.681^d	.682^d	.687^d	.630	.594	.601	.612	.364^d	.310	.420^d	.421
	GMean	**.685**^{d}	**.695**^{d}	**.702**^{d}	**.698**^{d}	**.673**^{d}	**.634**	**.695**^{d}	**.688**^{d}	**.416**^{d}	**.405**^{d}	**.471**^{d}	**.477**^{d}

		TREC10				TREC12			
$\mathcal{P}_{base}$	Predictor	CombMNZ	CombSUM	Borda	RRF	CombMNZ	CombSUM	Borda	RRF
WIG	Direct	.444	.436	.459	.449	.570	.599	.523	.525
	AMean	**.501**^{d}	**.504**^{d}	**.516**^{d}	**.509**^{d}	**.634**^{d}	**.636**^{d}	.633^d	.637^d
	GMean	.485^d	.478^d	.502^d	.494^d	.626^d	.623^d	**.639**^{d}	**.640**^{d}
NQC	Direct	.446	.394	.470	**.466**	.555	.567	.530	.527
	AMean	**.451**	.394	.472^d	**.466**	.570	.567	.563^d	.592^d
	GMean	.446	**.396**	**.480**	**.466**	**.582**^{d}	**.574**	**.643**^{d}	**.627**^{d}

Table 5: Predicting the query-performance of additional fusion methods (CombSUM, Borda and RRF). The best result per track, base predictor, and a fusion method, is boldfaced; 'd' marks a statistically significant difference with the Direct predictor; KL serves for the inter-list similarity measure.

quality attained when using NQC in AMean is always higher for CombMNZ than for CombSUM.

4.2.5 *Effect of free-parameter values*

The AMean and GMean prediction methods incorporate two free parameters: k, the number of highest ranked documents in a list — both for the lists used for fusion and for the final result list — that are considered by the base predictor employed; and, λ, which controls the balance between using prediction based on the final result list and that based on the lists that are fused.

Insofar, k and λ were set to values that yield optimal prediction quality for a predictor for a track's query set. (For the Direct predictors, k was set to optimize prediction quality.) In Figure 1 we present the effect on prediction quality of varying their values when WIG and NQC serve for the base predictors and KL is used for the inter-similarity measure. While CombMNZ serves for the fusion method, we note that the patterns presented are the same for the other fusion methods. When studying the effect of varying the value of k we set λ to its optimal value and vice versa.

The effect of k. We see in Figure 1 that the prediction quality of using WIG as a base predictor is relatively stable when varying the value of k.[10] Optimal prediction quality is obtained when k is set to a relatively small value. The latter observation is in accordance with findings about using WIG to predict the effectiveness of a single retrieved list [40]. The prediction quality of using NQC monotonically rises with increasing values of k, until optimal prediction quality is attained for $k = 100$. This finding is in line with those about using NQC to predict the effectiveness of a single retrieved list [31]. Finally, we observe that the effect of varying the value of k is quite consistent across tracks and between using AMean and GMean.

The effect of λ. Recall from Equation 3 that $\lambda = 0$ amounts to relying only on the prediction employed upon the final result list; and, increasing the value of λ results in more weight put on the predicted effectiveness of the lists that are fused.

When using WIG as the base predictor, whether in AMean or in GMean, the optimal prediction quality is almost always attained for $\lambda = 1$; i.e., not using the prediction performed upon the final result list, and relying only on the predicted effectiveness of the lists fused to create it. This is a striking observation as the goal of prediction is the effectiveness of the final result list. Thus, this finding re-echoes the importance of using prediction performed upon the lists that are fused so as to predict the effectiveness of the final result list — a key principle in our framework.

When using NQC as the base predictor, we see that the prediction quality patterns change between AMean and GMean. For AMean the patterns are not consistent across tracks, while for GMean the optimal prediction quality is often attained for $\lambda \in \{0.1, 0.2\}$. Thus, automatically setting λ's value for a prediction method and a query (and/or a corpus) is an important future venue to explore.

Finally, we see that the prediction quality for the Web tracks (TREC9 and TREC10) is consistently lower than that for the tracks using newswire corpora (TREC3, TREC8, and TREC12). This finding echoes those in previous reports about the query-performance prediction task for a single retrieved list being highly challenging for the WT10g corpus and its accompanying queries [17]. This corpus and queries were the ones used in TREC9 and TREC10.

4.2.6 *Learning free-parameter values*

As already noted, the free parameters (k and/or λ) of all predictors (ours and the reference comparisons) were set at the above to values that yield optimal prediction quality per track's query set. This practice enabled to study the impact of various aspects of prediction (e.g., the inter-list similarity measure, the aggregation method for prediction values, the fusion method used for retrieval) while ameliorating effects of free-parameter values. In Section 4.2.5 we explored the effects of the free-parameter values on prediction quality.

Our goal now is studying the prediction quality when free-parameter values are set using a held-out query set. We randomly split the queries for a track into two equal-sized folds, train and test. Then, the values of the free param-

[10]This also holds when k approaches 100, the number of documents in a list. For $k = 100$, WIG's prediction value for each of the 5 lists that are fused is $\frac{1}{100}$ (see Equation 9). In this case, prediction is based on that performed upon L_{res}, and on L_{res}'s similarity with the 5 lists.

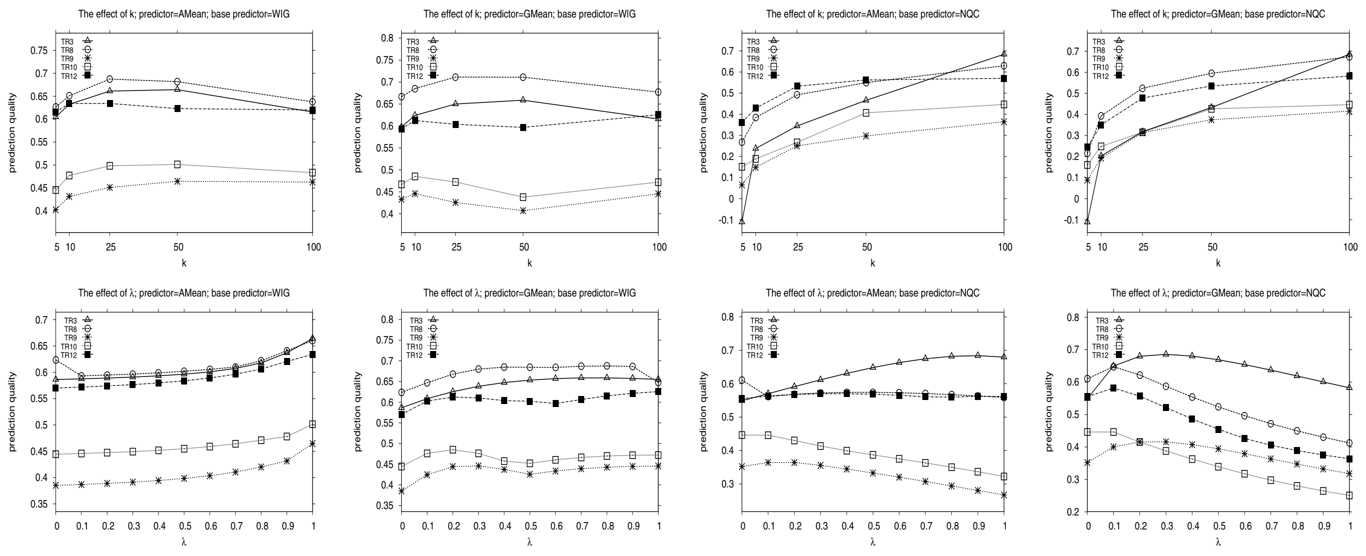

Figure 1: The effect on the prediction quality of AMean and GMean of varying the values of k (first row) and λ (second row); $\lambda = 0$ amounts to the Direct predictor that uses the base predictor only upon the final result list; WIG (two leftmost figures) and NQC (two rightmost figures) are used for base predictors. The KL inter-similarity measure is used in all cases. CombMNZ is the fusion method. 'TR' stands for TREC.

$\mathcal{P}_{base}$	Predictor	TREC3	TREC8	TREC9	TREC10	TREC12
WIG	Direct	.603 (.108)	.661 (.096)	.388 (.104)	.444 (.113)	.640 (.072)
	AMean	**.613** (.092)	$\underline{\textbf{.685}}$ (.089)	**.453**^{d} (.111)	**.478** (.117)	$\underline{\textbf{.643}}$ (.061)
	GMean	.604 (.100)	.662 (.101)	.442^d (.120)	.470 (.125)	.630 (.073)
NQC	Direct	.057 (.237)	.592 (.129)	.211 (.140)	.299 (.152)	.500 (.107)
	AMean	$\underline{\textbf{.656}}^d$ (.118)	**.636** (.124)	.303^d (.140)	.415 (.137)	**.601**^{d} (.078)
	GMean	.624^d (.145)	.613 (.140)	**.341**^{d} (.164)	**.431**^{d} (.139)	.600^d (.080)
sClarity	Direct	.400 (.102)	.526 (.132)	**.320** (.168)	.074 (.140)	.494 (.087)
	AMean	**.565**^{d} (.095)	**.598**^{d} (.095)	.285 (.132)	**.293**^{d} (.117)	**.560**^{d} (.077)
	GMean	.541^d (.121)	.571^d (.112)	.214^d (.158)	.219^d (.131)	.528^d (.090)
rClarity	Direct	.387 (.106)	.494 (.127)	**.308** (.165)	.054 (.139)	.415 (.085)
	AMean	**.588**^{d} (.097)	**.595**^{d} (.093)	.276^d (.131)	**.302**^{d} (.118)	**.559**^{d} (.077)
	GMean	.562^d (.119)	.559^d (.112)	.209^d (.153)	.217^d (.124)	.513^d (.089)

Table 6: Prediction quality when using a train query set for setting the values of free parameters of the predictors. Numbers in parentheses indicate the standard deviation of prediction quality over the 30 test sets of queries. CombMNZ serves for the fusion method, and KL serves for the inter-list similarity measure. Boldface marks the best result for a track and a base predictor. The best result in a column is underlined. Statistically significant differences with Direct are marked with 'd'.

eters that yield optimized prediction quality over the train fold are used for all queries in the test fold. We repeat this procedure 30 times, to avoid irregularities rising from a single random split, and report for each predictor its average prediction quality and its standard deviation over the 30 test folds. (Recall that the evaluation of prediction quality for any query set is based on 20 random samples of 5 runs.) The prediction quality numbers are presented in Table 6 when using the CombMNZ fusion method and the KL inter-list similarity measure. Note that the reported numbers are not necessarily comparable to those presented above. A case in point, the evaluation here is based on average prediction quality for 30 samples of half of the queries in the track, while that above was based on all queries per track.

We see in Table 6 that in a vast majority of the relevant comparisons (4 base predictors $\times$ 5 tracks), our AMean and GMean predictors outperform the Direct predictor. Furthermore, the prediction quality of AMean and GMean is statistically significantly better than that of Direct in 12 and 13 relevant comparisons, respectively, out of the entire 20. These findings, again, attest to the importance of integrating prediction performed upon the lists that are fused with prediction performed upon the final result list (Direct) — the core principle of our prediction framework.

A specific interesting observation is with regard to using NQC over TREC3, and sClarity and rClarity over TREC10. In these cases, the prediction quality of Direct, which is based only on the final result list, is extremely low. This is in part due to the fact that effective free-parameter values do not necessarily generalize across queries in this case. Yet, the prediction quality of using the base predictors in our AMean and GMean predictors can be quite higher, and to a statistically significant degree.

Another observation that we make based on Table 6 is that the standard deviation of the prediction quality of the AMean predictors is almost always lower than that of Direct. We note that while AMean and GMean incorporate two free parameters (k and λ), Direct incorporates only one

(k). The standard deviation for GMean is in most relevant comparisons higher than that for AMean, yet in quite a few relevant comparisons it is still lower than that of Direct. These findings attest to the improved robustness of the prediction quality of our framework, as measured over different query sets, with respect to that of the Direct predictors.

All in all, we see that our prediction framework is effective and quite robust with respect to the Direct predictors when setting free-parameter values using a train query set.

5. CONCLUSIONS AND FUTURE WORK

We addressed the query-performance prediction task for retrieval methods that are based on fusion of retrieved lists. Specifically, we presented a novel prediction framework that integrates prediction performed upon the final result list with that performed upon the lists that are fused to create it; inter-list similarity measures control the integration. The framework is generic in that it does not assume knowledge of the methods used to retrieve the lists that are fused, nor of the fusion method employed. Empirical evaluation demonstrated the merits of our approach.

We plan to explore whether the framework can be improved by exploiting knowledge of the fusion method employed. Furthermore, we intend to study whether the framework can be used for effective *meta fusion*; i.e., fusing result lists produced by different fusion methods based on the lists' predicted effectiveness.

Acknowledgments We thank the reviewers for their helpful comments. This paper is based upon work supported in part by the Israel Science Foundation under grant no. 557/09, by IBM's Ph.D. fellowship and SUR award, and by Google's and Yahoo!'s faculty research awards. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

6. REFERENCES

[1] J. A. Aslam and M. Montague. Models for metasearch. In *Proceedings of SIGIR*, pages 276–284, 2001.

[2] J. A. Aslam and M. H. Montague. Metasearch consistency. In *Proceedings of SIGIR*, pages 386–387, 2001.

[3] J. A. Aslam and V. Pavlu. Query hardness estimation using Jensen-Shannon divergence among multiple scoring functions. In *Proceedings of ECIR*, pages 198–209, 2007.

[4] N. Balasubramanian and J. Allan. Learning to select rankers. In *Proceedings of SIGIR*, pages 855–856, 2010.

[5] D. Carmel and E. Yom-Tov. *Estimating the Query Difficulty for Information Retrieval*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2010.

[6] G. V. Cormack, C. L. A. Clarke, and S. Büttcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of SIGIR*, pages 758–759, 2009.

[7] W. B. Croft, editor. *Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval*. Number 7 in The Kluwer International Series on Information Retrieval. Kluwer, 2000.

[8] W. B. Croft. Combining approaches to information retrieval. In Croft [7], chapter 1, pages 1–36.

[9] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *Proceedings of SIGIR*, pages 299–306, 2002.

[10] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. A language modeling framework for selective query expansion. Technical Report IR-338, Center for Intelligent Information Retrieval, University of Massachusetts, 2004.

[11] R. Cummins. Predicting query performance directly from score distributions. In *Proceedings of AIRS*, pages 315–326, 2011.

[12] R. Cummins, J. M. Jose, and C. O'Riordan. Improved query performance prediction using standard deviation. In *Proceedings of SIGIR*, pages 1089–1090, 2011.

[13] F. Diaz. Performance prediction using spatial autocorrelation. In *Proceedings of SIGIR*, pages 583–590, 2007.

[14] E. A. Fox and J. A. Shaw. Combination of multiple searches. In *Proceedings of TREC-2*, 1994.

[15] D. Harman and C. Buckley. The NRRC reliable information access (RIA) workshop. In *Proceedings of SIGIR*, pages 528–529, 2004.

[16] C. Hauff, D. Hiemstra, and F. de Jong. A survey of pre-retrieval query performance predictors. In *Proceedings of CIKM*, pages 1419–1420, 2008.

[17] C. Hauff, V. Murdock, and R. Baeza-Yates. Improved query difficulty prediction for the web. In *Proceedings of CIKM*, pages 439–448, 2008.

[18] B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. In *Proceedings of SPIRE*, pages 43–54, 2004.

[19] O. Kurland and L. Lee. Corpus structure, language models, and ad hoc information retrieval. In *Proceedings of SIGIR*, pages 194–201, 2004.

[20] O. Kurland, A. Shtok, D. Carmel, and S. Hummel. A unified framework for post-retrieval query-performance prediction. In *Proceedings of ICTIR*, pages 15–26, 2011.

[21] O. Kurland, A. Shtok, S. Hummel, F. Raiber, D. Carmel, and O. Rom. Back to the roots: A probabilistic framework for query-performance prediction. In *Proceedings of CIKM*, 2012.

[22] J. H. Lee. Combining multiple evidence from different properties of weighting schemes. In *Proceedings of SIGIR*, pages 180–188, 1995.

[23] J. H. Lee. Analyses of multiple evidence combination. In *Proceedings of SIGIR*, pages 267–276, 1997.

[24] D. Lillis, F. Toolan, R. W. Collier, and J. Dunnion. Probfuse: a probabilistic approach to data fusion. In *Proceedings of SIGIR*, pages 139–146, 2006.

[25] D. Metzler and W. B. Croft. A Markov random field model for term dependencies. In *Proceedings of SIGIR*, pages 472–479, 2005.

[26] M. Montague and J. A. Aslam. Condorcet fusion for improved retrieval. In *Proceedings of CIKM*, pages 538–548, 2002.

[27] J. Pérez-Iglesias and L. Araujo. Standard deviation as a query hardness estimator. In *Proceedings of SPIRE*, pages 207–212, 2010.

[28] D. Sheldon, M. Shokouhi, M. Szummer, and N. Craswell. Lambdamerge: merging the results of query reformulations. In *Proceedings of WSDM*, pages 795–804, 2011.

[29] M. Shokouhi. Segmentation of search engine results for effective data-fusion. In *Proceedings of ECIR*, pages 185–197, 2007.

[30] A. Shtok, O. Kurland, and D. Carmel. Using statistical decision theory and relevance models for query-performance prediction. In *Proccedings of SIGIR*, pages 259–266, 2010.

[31] A. Shtok, O. Kurland, D. Carmel, F. Raiber, and G. Markovits. Predicting query performance by query-drift estimation. *ACM Transactions on Information Systems*, 30(2):11, 2012.

[32] I. Soboroff, C. K. Nicholas, and P. Cahan. Ranking retrieval systems without relevance judgments. In *Proceedings of SIGIR*, pages 66–73, 2001.

[33] V. Vinay, I. J. Cox, N. Milic-Frayling, and K. R. Wood. On ranking the effectiveness of searches. In *Proceedings of SIGIR*, pages 398–404, 2006.

[34] C. C. Vogt and G. W. Cottrell. Fusion via linear combination of scores. *Information Retrieval*, 1(3):151–173, 1999.

[35] E. M. Voorhees. Overview of the TREC 2004 Robust Retrieval Track. In *Proceedings of TREC-13*, 2004.

[36] E. M. Voorhees. The TREC 2005 robust track. *SIGIR Forum*, 40(1):41–48, 2006.

[37] E. Yom-Tov, S. Fine, D. Carmel, and A. Darlow. Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In *Proceedings of SIGIR*, pages 512–519, 2005.

[38] H. P. Young. An axiomatization of Borda's rule. *Journal of Economic Theory*, 9:43–52, 1974.

[39] Y. Zhou and W. B. Croft. Ranking robustness: a novel framework to predict query performance. In *Proceedgins of CIKM*, pages 567–574, 2006.

[40] Y. Zhou and W. B. Croft. Query performance prediction in web search environments. In *Proceedings of SIGIR*, pages 543–550, 2007.