Knowl Inf Syst (2014) 41:189–221
DOI 10.1007/s10115-014-0759-6

REGULAR PAPER

Automatic ranking of retrieval models using retrievability measure

Shariq Bashir · Andreas Rauber

Received: 28 March 2012 / Revised: 24 September 2012 / Accepted: 20 October 2012 /
Published online: 1 June 2014

Abstract Analyzing retrieval model performance using retrievability (maximizing findability of documents) has recently evolved as an important measurement for recall-oriented retrieval applications. Most of the work in this domain is either focused on analyzing retrieval model bias or proposing different retrieval strategies for increasing documents retrievability. However, little is known about the relationship between retrievability and other information retrieval effectiveness measures such as precision, recall, MAP and others. In this study, we analyze the relationship between retrievability and effectiveness measures. Our experiments on TREC chemical retrieval track dataset reveal that these two independent goals of information retrieval, maximizing retrievability of documents and maximizing effectiveness of retrieval models are quite related to each other. This correlation provides an attractive alternative for evaluating, ranking or optimizing retrieval models' effectiveness on a given corpus without requiring any ground truth available (relevance judgments).

Keywords Retrieval models evaluation · Retrieval bias analysis · Automatic ranking of retrieval models · Genetic programming

1 Introduction

One important area of information retrieval (IR) research is to evaluate the effectiveness of retrieval models using test collections [6]. A test collection consists of a corpus of documents, a set of topic queries and the relevance judgments (a list that describes which documents are relevant for which query topic). While documents and queries are relatively easy to gather, creating relevance judgments requires significant effort and resources. In recent years,an

S. Bashir (✉) · A. Rauber
Institute of Software Technology and Interactive Systems, Vienna University of Technology, Vienna, Austria
e-mail: shariq.bashir@jinnah.edu.pk
URL: http://www.ifs.tuwien.ac.at

A. Rauber
e-mail: rauber@ifs.tuwien.ac.at
URL: http://www.ifs.tuwien.ac.at

increased interest is observed in proposing methods for automatic ranking of retrieval models without human-generated relevance judgments [1,23,34,40,41,47]. This paper presents a method of ranking retrieval models without relevance judgments on the basis of retrievability [5]. Previous studies on automatic ranking of retrieval models either used pseudo-relevance judgments or relied on the similarity between retrieval models on the basis of retrieved documents. We use retrievability inequality between the documents of a collection as a base for performing this ranking. Retrievability (also referred to as findability) of a document provides an indication of how easily a document can be retrieved using a given retrieval model. It basically determines the likeliness that a document can be found given (a) a hypothesis of the queries that are likely to be generated, (2) a retrieved model, and (3) a hypothesis of to what extent down a ranked result list a user is willing to proceed. The combination of these factors makes some documents more retrievable than others. A document that has high retrievability means that a user has a high probability of finding it through a query. Conversely, a document with low retrievability is likely to be difficult to find by the user. The overall retrievability inequality between the documents of a collection expresses the retrieval bias of a retrieval model. It can be expressed using the Gini coefficient [22] (i.e., the summarized retrievability over all documents). The high retrieval bias of a retrieval model (the Gini coefficient close to one) indicates that the given retrieval model makes some documents of the collection more retrievable at the expense of others. However, a low retrieval bias of a retrieval model (Gini coefficient close to zero) indicates that the given retrieval model tries the balance the retrievability of all documents.

Retrievability measure has been used in a number of different contexts (see [5] for more details and [3,7–11] for examples of its usage in practice). However, there has been little work on analyzing the relationship between retrievability and more standard IR effectiveness measures. One important aspect of retrievability measure is that it can be analyzed or estimated without the availability of explicit ground truth (relevance judgments). It thus provides an attractive alternative for the automatic ranking of retrieval models. Additionally, it can be also used for tuning a retrieval model's effectiveness by varying its parameter values or retrieval features over retrievability so that they can perform well for a given collection. However, these can only be possible if there is a significant positive correlation between the retrieval bias (i.e., the summarized retrievability of all documents) and effectiveness measures. This is because high or low retrieval bias of retrieval models does not mean directly that the retrieval models will also perform well on the effectiveness measures. For instance, given the definition of retrievability, a retrieval model that ranks the documents by randomly selecting from the document collection would provide a better retrievability to all documents. This would result in a low retrieval bias, but very poor effectiveness for finding the relevant documents. Conversely, a retrieval model that only ranks the set of known relevant documents at the top-ranked positions, regardless of given queries would provide a high inequality among documents (poor retrievability and high retrieval bias) but better effectiveness for a set of known topics. This indicates that neither extreme is desirable. However, to what extent we need to trade-off between the retrievability and the effectiveness depends upon the correlation between them.

The focus of this paper is twofold. First, we try to identify the correlation between Gini coefficients (retrieval bias) of retrieval models and recall, precision, MAP and b-pref (effectiveness) measures. Second, we aim at optimizing retrieval models on the basis of retrieval bias. We start our analyses, by first examining the relationship between them by ranking retrieval models on retrievability and effectiveness measures. In order to do this, we test a large variety of different retrieval strategies. These include standard retrieval models, language modeling-based retrieval models, term proximity-based retrieval models and low-level features of IR. Once the systematic relationship between the retrievability and effectiveness measures is confirmed, we then propose some strategies for optimizing the effectiveness of retrieval models on the basis of retrievability. These include tuning the parameter values of retrieval models over retrievability and partitioning the collection on the basis of low and high retrievability of documents. We further show that this principle may even be applied to learn a completely new retrieval model using a genetic programming approach, where the fitness function is guided solely by the evolving retrieval model's retrieval bias. This offers a promising approach for fine-tuning and optimizing retrieval models for a specific collection and their characteristics without having to invest enormous amounts of effort into manually labeling the relevant documents of a collection, assessing their relevance. Our analysis on the relationship between retrievability and effectiveness measures shows that the two goals of IR, minimizing retrieval bias and maximizing effectiveness, are quite related to each other. This motivates the hypothesis that a reasonably good retrieval model for a specific collection can be obtained by selecting a retrieval model that minimizes retrieval bias (i.e., when there is the least retrievability inequality between documents according to Gini coefficient given the $r(d)$ values).

The remainder of this paper is structured as follow. Section 2 reviews related work on the bias assessment of retrieval models and automatic ranking of retrieval models effectiveness. Section 3 summarizes the concept of retrievability measurement. Section 4 describes the TREC-CRT (TREC Chemical Retrieval Track 2009) benchmark corpus that is used for experiments in this paper and retrieval bias results for several configurations of this collection with different retrieval models. Section 5 analyzes the relationship between retrievability and effectiveness measures through different experiments. In Sect. 6, retrievability is used as a fitness function in genetic programming for automatically evolving effective retrieval model, demonstrating the approach of optimizing a retrieval function purely based on (unsupervised) retrievability scores. Finally, Sect. 7 briefly summarizes the key lessons learned from this study.

2 Related work

Our study is at the crossover between two IR domains: retrieval models bias assessments and automatic ranking of retrieval models. In this section, we give an overview of the main works of both domains.

2.1 Bias assessment of retrieval models

Bias assessment of search engines/retrieval models has always received a considerable attention in the IR research community. Web coverage and documents retrievability are two well-known measures for discovering search engine bias. Work on web coverage-based bias considered a range of possible biases [4,29,33,43,46]. These include aspects such as whether one site has more coverage than another, whether sites in particular geographical locations are favored [43], or whether search engines are biased given a particular topic. These studies are usually motivated by the view that search engines may be providing biased content, and these measures are aimed at being regulatory in nature.

Unlike web coverage-based bias measures, retrievability extremely focuses only on the documents findability or their accessibility, and this can also be used to detect bias of a retrieval model. The retrievability measures how likely a document can be found at all with a given retrieval model. Retrievability experiments conducted by Azzopardi and Vinay [5] on AQUAINT and .GOV datasets revealed that with a TREC-style evaluation, a proportion of the documents with very low retrievability scores (sometimes more than 80 % of the total collection with higher retrieval bias models) can be removed without significantly degrading effectiveness. This is because the retrieval models are unlikely to ever retrieve these documents due to the high bias they exhibit over the documents of a given collection.

Bashir and Rauber [8] analyzed retrievability of documents specifically with respect to relevant and irrelevant queries to identify whether highly retrievable documents are really highly retrievable, or whether they are simply more accessible from many irrelevant queries rather than from relevant queries. Their experiments revealed that 90 % of documents that are highly retrievable across all types of queries are not highly retrievable when they are searched from relevant queries.

Experiments on query expansion for improving the retrievability of documents are investigated in Bashir and Rauber [9,11]. These showed that the initial users' queries are not efficient for correctly capturing and interpreting the context of search. Therefore, a few numbers of noisy documents at higher rank positions pull the majority of retrievability scores toward them, creating a high retrieval bias. In order to mitigate this limitation, the authors expand the initial queries with query expansion through pseudo-relevance feedback (PRF) documents. Experiments on different collections of patent documents suggest that the query expansion through PRF can be used as an effective approach for increasing the retrievability of documents and decreasing retrieval bias.

Bashir and Rauber [10] proposed an approach for improving the retrievability of documents by partitioning the collection. In their approach, rather than retrieving and ranking documents from a single collection, they first split the two categories of documents *low* and *high* retrievable documents into two partitions. Having split the collection into these two categories, they then perform retrieval by treating these classes as independent partitions and process queries independently for each partition and subsequently combine the result sets. Results showed that this helps in increasing overall retrievability, reducing the dominance of certain documents in query processing and thus reducing the bias of retrieval models.

Bache and Azzopardi [7] also performed retrievability experiments on a patent collection. Their results confirmed the presence of a large amount of retrieval bias on the patent collection. In order to reduce this retrieval bias, they used a series of hybrid retrieval models. The features of these hybrid models were based on the term frequency sensitivity, length normalization and the convexity. Results showed that the hybrid models provide greater access to the documents than the standard retrieval techniques (BM25 and TFIDF).

2.2 Automatic ranking of retrieval models

Evaluating the effectiveness of retrieval models normally requires a test collection, a set of queries and the relevance judgments for each given query. However, for a very large collection, creating relevance judgments is a difficult and extremely time-consuming task, because for each given query all documents need to be judged for relevance. Due to this reason in past few years, increased research is observed in the IR research community in proposing methods to automatically rank the retrieval models without resource to relevance judgments available.

Soboroff et al. [40] proposed a method of ranking retrieval models without human relevance judgments. Their methodology replaced the human relevance judgments with a number of pseudo-relevance judgments. They choose these pseudo-relevance judgments randomly from a pool. They used four different variations of pseudo-relevance judgments selection in order to check the consistency of the random selection mechanism with the human relevance judgments. Their experiments revealed that the retrieval models rankings produced from their method correlate positively to the actual TREC rankings.

Aslam and Savell [2] proposed a method to rank retrieval models on the basis of similarity between retrieval models. The similarity between two models is the ratio of the number of retrieved documents in their intersection and union sets. Each model then ranked according to its average similarity with other models. Wu and Crestani [47] also found the similarity between retrieval models useful for ranking retrieval models. They proposed a reference count (RC) method to rank retrieval models. Their method processed each query with all given retrieval models and count the frequency (reference counts) of all documents in the result lists. The retrieval models are then ranked according to their total reference count sum of retrieved documents. Instead of just considering the frequency of documents, they further considered several variations of RC by assigning higher weights to those documents that frequently appeared at the top rank positions of many result lists. Results showed that the proposed methods are effective; however, neither of them is good for predicting the effectiveness of top-performing retrieval models.

Nuray and Can [34] investigated pseudo-relevance judgments for automatic ranking of retrieval models on a very similar data set as used by [40], specifically TREC-3,5,6,7. However, contrary to [40], they did not utilize all available retrieval models in the derivation (generation) of pseudo-relevance judgments. The authors used several approaches for finding out a good subset of $p\%$ of retrieval models. The best approach that they found is to select those retrieval models that are most different from the average retrieval model. Once a subset of top-performing retrieval models is determined, the top b ranked documents in the result lists of selected retrieval model are merged, and the top $s\%$ documents of the merged list are used for the pseudo-relevance judgments. In order to merge the result lists, they examined three techniques (rank position, borda counting and condorcet voting), and the best performing technique that they found is to rely on condorcet voting. In condorcet voting, both the frequencies and ranks of documents in different result lists play a combined role in choosing the dominant $s\%$ documents for the pseudo-relevance judgments. They showed that their results outperform the results reported in [40] for the datasets evaluated.

Spoerri [41] found that using all available retrieval models at once (as done mostly in the previous approaches) creates a considerable amount of noise as near duplicate models could be entered, and this noise can boost some retrieval models in a biased way. On the basis of this hypothesis, they repeatedly ranked a set of five randomly selected retrieval models on the basis of their result lists overlap structure and then averaged the results across all trials to gain a ranking for all retrieval models. Their reported experiments on TREC-3,6,7,8 showed significantly higher correlations than the previous work, and the best models are consistently ranked in at least the top half of the ranking.

All of the methods reviewed above either used pseudo-relevance judgments or considered the similarity between retrieval models on the basis of top retrieved documents in order to rank retrieval models. Our approach is different to these; we consider the retrievability inequality between documents of a collection (representing retrieval bias) as a measure for ranking retrieval models.

3 Retrievability measure

The following description of retrievability measurement as introduced by Azzopardi and Vinay [5] (adopted from [10]) provides a quick introduction of how it is measured.

Given a collection D, a retrieval models accept a user query q and return a ranked list of documents, which are deemed to be relevant to q. We can thus consider the retrievability of a document as influenced by two factors: (a) how retrievable it is, with respect to the collection D, and (b) the effectiveness of the ranking strategy of the retrieval model. In order to derive an estimate of this quantity, Azzopardi and Vinay [5] used query set-based sampling [13]. The query set Q could either be a historical sample of queries or a artificial simulated substitute similar to users' queries. Then, each user's $q \in Q$ is issued to the retrieval model, and the retrieved documents along with their positions in the ranked list are recorded. Intuitively, retrievability of a document d is high when:

1. there are many probable queries in Q which can be expressed in order to retrieve d and
2. when retrieved, the rank r of the document d is lower than a rank cutoff (threshold) c. This is the point at which the user would stop examining the ranked list. This is a user dependent factor and thus reflects a particular retrieval scenario in order to obtain a more accurate estimate of this measure. For instance, in web search scenario, a low c would be more accurate as users are unlikely to go beyond the first page of the results, while in the context of recall-oriented retrieval settings (for instance, legal or patent retrieval), a high c would be more accurate.

Thus, based on the Q, r and c, we formulate the following measure for the retrievability of d.

$$r(d) = \sum_{q \in Q} p(q) \cdot \hat{f}(k_{dq}, c) \tag{1}$$

$\hat{f}(k_{dq}, c)$ is a generalized utility/cost function, where k_{dq} is the rank of d in the result list of query q, and c denotes the maximum rank that a user is willing to proceed down in the ranked list. The function $\hat{f}(k_{dq}, c)$ returns a value of 1, if $k_{dq} \leq c$, and 0 otherwise. $p(q)$ denotes the likeliness that a user actually issues query q. This probability may be hard to determine explicitly and is thus frequently set to 1, i.e., to give all queries equal probabilities. More complex heuristics considering the length of the query, the specificity of the vocabulary, etc. may be considered. Defined in this way, the retrievability of a document is essentially a cumulative score that is proportional to the number of times the document can be retrieved within that cutoff c over the set Q. This fulfills our aim, in that the value of $r(d)$ will be high when there are a large number of (highly probable) queries that can retrieve the document d at the rank less than c, and the value of $r(d)$ will be low when only a few number of queries retrieve the document. Furthermore, if a document is never returned at the top-ranked c positions, possibly because it is difficult to retrieve by the retrieval model, then the $r(d)$ is zero.

The cumulative measure of the retrievability score of a document on the basis of binary $f(k_{dq}, c)$ function ignores the ranking position of a document in ranked result list, i.e., how accessible the document is in the ranking. A gravity-based measure can be used for this purpose by setting the function to reflect the effort of going further down in the ranked result list, and it is defined as

$$\hat{f}(k_{dq}, \beta) = \frac{1}{(c_{dq})^{\beta}} \tag{2}$$

The rank cutoff factor is changed to β which is a dampening factor that adjusts how accessible the document is in the ranking. In our experiments, we calculate the retrievability score of documents only on the basis of cumulative measure.

The inequality between the retrievability score of documents can be further analyzed using the *Lorenz curve* [22]. In Economics and the Social Sciences, a Lorenz curve is used to visualize the inequality of the wealth in a population. This is performed by first sorting the individuals in the population in ascending order of their wealth and then plotting a cumulative wealth distribution. If the wealth in the population was distributed equally, then we would expect this cumulative distribution to be linear. The extent to which a given distribution deviates from the equality is reflected by the amount of skewness in the distribution. Azzopardi and Vinay [5] employed similar idea in the context of a population of documents, where the wealth of documents is represented by $r(d)$ function. The more skewed the plot, the greater the amount of inequality, or bias within the population. The *Gini coefficient* [22] G is used to summarize the amount of retrieval bias in the Lorenz curve and provides bird's eye view. It is computed as follows.

$$G = \frac{\sum_{i=1}^{|D|}(2 \cdot i - |D| - 1) \cdot r(d_i)}{(|D| - 1) \sum_{j=1}^{|D|} r(d_j)} \tag{3}$$

D represents the set of documents in the collection. If $G = 0$, then no bias is present because all documents are equally retrievable. If $G = 1$, then only one document is retrievable and all other documents have $r(d) = 0$. By comparing the Gini coefficients of different retrieval methods, we can analyze the retrieval bias imposed by the underlying retrieval systems on a given document collection.

The retrievability measure that is defined in Eq. 1 cumulates the retrievability scores of documents over all queries. Thus, in case of exhaustive query generation, long documents potentially have higher numbers of query combinations possible than short documents due to their larger vocabulary sizes. This may favor long documents that are retrievable from only a small fraction of their all possible queries over short documents that are potentially retrievable from a large faction of their queries. We thus propose to normalize the cumulative retrievability scores (normalized retrievability) of documents with the total number of queries they were created from and thus potentially can retrieve a particular document. It is defined as:

$$\hat{r}(d) = \frac{\sum_{q \in Q} p(q) \cdot f(k_{dq}, c)}{|\hat{Q}_{(d)}|} \tag{4}$$

The cumulative $r(d)$ scores of documents are normalized with the $\hat{Q}_{(d)}$. This is the set of all queries that can retrieve d when not considering any rank cutoff factor. This accounts for difference in the vocabulary richness across different documents of collection. The documents with a large vocabulary size produce many more queries. Such documents are thus theoretically retrievable through a much larger set of queries. The standard $r(d)$ score would thus penalize a retrieval model that provides perfectly balanced retrievability to all documents just because some documents are rather vocabulary poor and cannot be retrieved by more than the few queries that can be created from their vocabulary. This is where a normalized retrievability score accounting for the different vocabulary sizes per document, and it provides an unbiased representation of the retrieval bias without automatically inflicting a penalty on the retrieval models that favor or disfavor long documents. Again, the relative retrievability score may be complemented with factors accounting for the likeliness of the individual queries $p(q)$.

4 Experiments setup

4.1 Document collection and query generation for retrieval bias analysis

To analyze the relationship between the effectiveness and the retrieval bias of retrieval models, we use the 1.2 million patent documents from the TREC Chemical Retrieval Track (TREC-CRT) [31].[1] As we are specifically interested in the relationship between effectiveness measures (such as precision, recall and MAP) and retrievability, we will compute the retrievability for only the subset of 35,000 documents. We select this subset randomly from the 1.2 million documents. While it would be possible to compute the retrievability scores for all documents (apart from being almost prohibitively expensive due to the enormous amount of queries that can be generated over the entire vocabulary), it would seem unfair to use more information for retrievability measure (thus begin more robust) than for effectiveness measures. Note, however, that all queries are of course processed over the entire collection of 1.2 million documents. Queries will be generated from only the 35,000 documents.

We consider all sections (title, abstract, claims, description and background summary) of patent documents for both retrieval and query generation. Stop words are removed prior to indexing and words stemming is performed with the Porter stemming algorithm. Additionally, we do not use those terms of the collection that have document frequency greater than 25 % of the total collection size to remove high-frequency stop words. For retrievability analysis, queries are generated with the combinations of those terms that appear more than one time in the document. For these terms, all 3-terms and 4-terms combinations are used in the form of boolean AND queries for creating the exhaustive set of queries Q, with duplicate queries being removed. Additionally, we consider only those queries that have query result list size of more than the rank cutoff $c = 100$.

The experiments presented in Sects. 5 and 6 are verified with several different retrieval models' parameter values and ranking features configurations, where each experiment run requires the processing of all queries in Q. This requires large processing time and resources. Thus, in order to complete the experiments in a reasonable time, we select a subset of two million queries from Q. However, rather than selecting this subset randomly, we select it on the basis of query quality prediction [24,50]. This is further motivated by earlier analysis of the relationship between query quality and retrieval bias [12]. In this method, we first order all queries in Q using the simplified query clarity score (SCS) [16]. Then, we select the two million queries that have the highest SCS scores. These queries are then used for document retrieval against the complete collection of *1.2 million* documents as boolean AND queries with subsequent ranking according to the chosen retrieval model to determine the retrievability scores of documents as defined in Eq. 4.

4.2 Effectiveness analysis

We select the prior art (PA) task of the *TREC-CRT* collection for analyzing the effectiveness of retrieval models. The PA task consisted of 1,000 topic queries that are the full-text patent documents (i.e., consisting of at least claims and abstract or description) taken from both the European Patent Office (EPO) and the US Patent Office (USPTO). The goal of searching a patent database for the prior art search task is to find all previously published related patents on a given topic [21,31,32]. It is a common task for *patent examiners* and *attorneys* to decide whether a new patent application is novel or contains technical conflicts with some already

[1] Available at http://www.ir-facility.org/research/evaluation/trec-chem-09.

patented invention. They collect all related patents and report them in a search report. We use these reports as relevance judgments. Next, we apply a standard approach for query generation in the patent retrieval domain. From each topic, we select only the claim section because it is regarded as being the most representative piece of text, characterizing the scope of invention well due to the rules of the patent system worldwide as done also in [21,25,32,37]. In order to build the prior art queries from the claim sections, we first sort all the term in the claim sections on the basis of their increasing term frequencies. Next, we select the top 30 terms that have highest frequencies and use these terms in the form of a long query for searching the relevant documents. Note that more complex query generation approaches may be used. Yet, as our primary motivation is to analyze the relationship between effectiveness and retrievability, this standard baseline is sufficient.

4.3 Retrieval models

The retrieval models that we use for retrieval bias analysis include standard retrieval models, language modeling-based retrieval models, term proximity-based retrieval models and low-level retrieval features of IR.

4.3.1 Standard retrieval models

- **TFIDF:** The TFIDF (term frequency inverse document frequency) is a retrieval model often used in information retrieval. It is a statistical measure used to evaluate how important a query term is to a document. The importance increases proportionally to the number of times a term appears in the document but is offset by the frequency of the term in the collection. The standard TFIDF retrieval model is described as follows:

$$TFIDF(d,q) = \sum_{t \in q} tf_{t,d} \cdot log \frac{|D|}{df_t} \tag{5}$$

  $tf_{t,d}$ is the term frequency of query term t in d, $|D|$ is the total number of documents in the collection, and df_t represents the total number of documents containing t.
- **NormTFIDF:** The standard TFIDF does not normalize the term frequencies relative to document length, thus sensitive and bias toward large absolute term frequencies. It is possible to address the length bias using document length $|d|$ and defied normalized TFIDF (NormTFIDF) as:

$$NormTFIDF(d,q) = \sum_{t \in q} \frac{tf_{t,d}}{|d|} \cdot log \frac{|D|}{df_t} \tag{6}$$

- **BM25:** Okapi BM25 arguably one of the most important and widely used information retrieval models. It is a probabilistic function and nonlinear combination of three key attributes of a document: term frequency $t_{t,d}$, document frequency df_t and the document length $|d|$. The effectiveness of BM25 is controlled by two parameters k and b. These parameters control the contributions of term frequency and document length. We used the following standard function of BM25 proposed by [36]:

$$BM25(d,q) = \sum_{t \in q} log \frac{|D| - df_t + 0.5}{df_t + 0.5} \cdot \frac{tf_{t,d}(k+1)}{tf_{t,d} + k(1 - b + b\frac{|d|}{|\overline{d}|})} \tag{7}$$

  $|\overline{d}|$ is the average document length in the collection from which the documents are drawn. k and b are two parameters, and they are used with $k = 2.0$ and $b = 0.75$.

- **SMART:** The System for Manipulating and Retrieving Text (SMART) is a retrieval model in information retrieval. It is based on the vector space model. We use the following variation of SMART developed by Singhal [38] at AT&T Labs.

$$SMART(d,q) = \sum_{t \in q} (w_{d,t} \cdot w_{q,t}) \tag{8}$$

$$w_{d,t} = \frac{1 + log(tf_{t,d})}{1 + log(avtf)} \cdot \frac{1}{0.8 + 0.2\frac{utf}{pivot}} \tag{9}$$

$$w_{q,t} = (1 + log(tf_{t,d})) \cdot log\frac{|D|+1}{df_t} \tag{10}$$

  $avtf$ represents the average number of occurrences of each term in the d, utf is the number of unique terms in d, and $pivot$ represents the average number of unique terms per document.

4.3.2 Language models with term smoothing

Language model tries to estimate the relevance of document by estimating the probabilities of terms in the document. The terms are assumed to occur independently, and the probability is the product of the individual query's terms given the document model M_d of document d. M_d is the document under consideration for which the language model tries to estimate the probability that the query q was generated by this language model.

$$P(q|M_d) = \prod_{t \in q} P(t|M_d) \tag{11}$$

$$P(t|M_d) = \frac{tf_{t,d}}{|d|} \tag{12}$$

The overall similarity score for the query and the document could be zero if some of query terms do not occur in the document. However, it is not sensible to rule out a document just because of missing only a few or single term. For dealing with this, language models make use of smoothing to balance the probability mass between the occurrences of terms present in documents and the terms not found in the documents. We use the following four variations of terms smoothing in our experiments.

- **Jelinek-Mercer Smoothing:** Jelinek-Mercer smoothing [48] combines the relative frequency of a query's term $t \in q$ in the document d with the relative frequency of the term in the collection (D). The amount of smoothing is controlled by the λ, and it is set between 0 and 1. Small smoothing values of λ close to 0 add only the contribution of term frequencies, while large λ values reduce the effect of relative term frequencies within the documents, and more importance is given toward the relative frequencies of terms in the collection.

$$P(t|M_d) = (1-\lambda)\frac{tf_{t,d}}{|d|} + \lambda P(t|D) \tag{13}$$

  $P(t|D)$ is the probability of term t occurring in the collection ($\sum_{d \in D} tf_{t,d} / \sum_{d \in D} |d|$). According to the suggested value of λ by Zhai [48], we use λ with 0.7.
- **Dirichlet (Bayesian) Smoothing (DirS):** This smoothing technique makes smoothing dependent on the document length. Since long documents allow us to estimate the language model more accurately, therefore this technique smoothes them less, and this is done with

the help of a parameter μ. Since the value of μ is added in the document length, thus small values of μ retrieve less long documents. If the μ is used with large values, then the distinction for difference between document lengths becomes less extreme, and long documents are more favored over short documents. Again, this favoritism mostly occurs in case of long boolean OR queries.

$$P(t|M_d) = \frac{tf_{t,d} + \mu P(t|D)}{|d| + \mu} \tag{14}$$

According to Zhai [48] suggestion, we use the μ with 2,000.

- **Two-Stage Smoothing (Two-Stage):** This smoothing technique first smoothes the document model using the Dirichlet prior probability with the parameter μ (as explained above), and then, it mixes the document model with the query background model using Jelinek-Mercer smoothing with the parameter λ. The query background model is based upon the term frequency in the collection. The smoothing function is therefore:

$$P(t|M_d) = (1-\lambda)\frac{tf_{t,d} + \mu P(t|D)}{|d| + \mu} + \lambda P(t|D) \tag{15}$$

where μ is the Dirichlet prior probability, and λ is the Jelinek-Mercer parameter. In our experiments, we use the parameter $\mu = 2{,}000$ and $\lambda = 0.7$, respectively.

- **Absolute Discount Smoothing (AbsDis):** This technique makes smoothing by subtracting a constant $\delta \in [0, 1]$ from the counts of each seen term. The effect of δ is similar to Jelinek-Mercer parameter λ, but differs in this sense that it discounts the seen terms probabilities by subtracting a constant δ instead of multiplying them by $(1-\lambda)$.

$$P(t|M_d) = \frac{max(tf_{t,d} - \delta, 0)}{|d|} + \frac{\delta |T_d|}{|d|} P(t|D) \tag{16}$$

T_d is the set of all unique terms of d. We use the δ with 0.7.

4.4 Low-level retrieval features

This set corresponds to standard statistics in information retrieval. These are

- sum of absolute query term frequencies within document ($tf(d,q) = \sum_{t \in q} tf_{t,d}$),
- sum of normalized query term frequencies relative to document length ($ntf(d,q) = \sum_{t \in q} tf_{t,d}/|d|$),
- sum of document frequency of query terms ($sdf(d,q) = \sum_{t \in q} df_t/|D|$),
- document length ($|d|$),
- document vocabulary size ($|T_d|$)
- and the sum of probability of query's term occurring in the collection ($scf(d,q) = \sum_{t \in q} cf_t / \sum_{d \in D} |d|$).

4.5 Term proximity-based retrieval models

The term proximity set consists of several retrieval models that try to capture the closeness of query terms in the documents. The underlying intuition is that the more compact or closed the query terms are in the documents, the more likely it is that they are topically related, and thus the higher the possibility that the term proximity-based retrieval models would help in decreasing the retrieval bias and increasing the effectiveness. For each query, we first use the *JM* language model to retrieve top the 500 documents and then use the term proximity-based retrieval models to re-rank them. This feature set consists of the following eight features. More information about these features can be found [14,18,42,49].

- **(f1)** *SumMinDist f1(d,q):* is the sum of the minimum distances of all the query's term pairs in the document. The minimum distance is measured in terms.
- **(f2)** *SumMaxDist f2(d,q):* Similar to *SumMinDist*, it is the sum of maximum distances of all query's term pairs in the document.
- **(f3)** *AvgDist f3(d,q):* This feature calculates the average of the sum of all query's term pairs distances in the document. It tries to capture where the terms of a query are occurring together. This feature rewards those documents of a result list that contain a large number of query's terms that are frequently very close to each other in some short segments of the text (e.g., in the same phrase, sentence or paragraph).
- **(f4)** *MinDistCount f4(d,q):* This feature counts the frequency of query's term pairs in the document that have a minimum distance of less than four terms.
- **(f5)** *AvgPairDist f5(d,q):* Similar to *AvgDist*, this feature calculates the average of the sum of distances between all query's term pairs and all single terms of a query.
- **(f6)** *CoOccurrence f6(d,q):* This feature counts the frequency of co-occurrence of query's term pairs in the documents within a window of less than four terms.
- **(f7)** *PairCoOccurrence f7(d,q):* Similar to *CoOccurrence*, this feature counts the frequency of the co-occurrences of all query's term pairs with all single terms of a query within a window of less than ten terms.
- **(f8)** *MinCover f8(d,q):* is defined as the shortest segment of a document that covers all terms of a query at least once. The intuition for this feature is that if all terms of a query reside in a smaller segment of the text, then this indicates that the document contains a segment that has a high probability of relevance, and thus, the document has a higher probability of relevance.

Table 1 lists the retrievability inequality between documents providing Gini coefficients (retrieval bias) of retrieval models for a range of rank cutoff factors. Note that high retrieval bias occurs when limiting oneself to short result lists of 5 or 50 documents. The Gini coefficient tends to decrease slowly for all retrieval models as the rank cutoff factor increases. This indicates that the retrieval bias of a retrieval model is mitigated by the willingness of the user's to search deeper down the result list. If a user examines only a small portion of the result list, then he/she will face a greater degree of retrieval bias.

Before performing a comparison on high-level retrieval models, one important factor that we consider important to mention here is that the retrieval bias of retrieval models depend upon the following two attributes of retrieval models: (a) normalized term frequency relative to document length and (b) term document frequency or the term collection frequency relative to the total size of collection. Due to these attributes, the higher of lower retrieval bias of retrieval models depend upon, whether or not the combination of these attributes are presented in the retrieval models, and how these are controlled in the retrieval model (i.e., depends on any parameter or does not depend on any parameter). *SMART* and *NormTFIDF* exhibit higher retrieval bias as compared to other retrieval models. The main reason is that these models do not have above attributes, and thus, in these model, large absolute term frequencies of query terms are preferred over small absolute term frequencies. Due to this reason, the long documents for these models have high percentage of retrievability out of their total queries than short documents, and this increases the overall retrieval bias of these models. *TFIDF* has both of above attributes, and thus, *TFIDF* has lower retrieval bias than *SMART* and *NormTFIDF*. However, as compared to (*BM25*, *DirS*, *TwoStage*, *JM* and *AbsDis*), *TFIDF* does not control these attributes with the help of any parameter, and

Table 1 Gini coefficient scores representing the retrieval bias of different retrieval models on the *TREC-CRT* collection.

Retrieval model	Rank cutoff factors		
	$c = 50$	$c = 100$	$c = 250$
Standard retrieval models and language models			
NormTFIDF	0.70	0.62	0.51
BM25	**0.57**	**0.52**	**0.44**
DirS	0.63	0.57	0.50
JM	0.68	0.62	0.51
AbsDis	0.66	0.60	0.50
TwoStage	0.64	0.56	0.46
TFIDF	0.95	0.91	0.81
SMART	0.96	0.93	0.87
Term proximity-based retrieval models			
SumMinDist	0.58	0.55	0.51
SumMaxDist	0.76	0.68	0.53
AvgDist	0.75	0.67	0.52
MinDistCount	**0.48**	**0.45**	**0.43**
AvgPairDist	0.72	0.66	0.53
CoOccurrence	0.57	0.49	0.43
PairCoOccurrence	**0.39**	**0.39**	**0.40**
MinCover	0.60	0.58	0.52
Low-level retrieval features			
tf(d,q)	0.95	0.92	0.83
ntf(d,q)	**0.71**	**0.63**	**0.51**
sdf(d,q)	0.85	0.85	0.83
$\|d\|$	0.80	0.74	0.61
$\|T_d\|$	0.99	0.99	0.99
scf(d,q)	0.85	0.85	0.83

Results in bold show retrieval models have less retrieval bias than other models

thus, it exhibits higher retrieval bias than *BM25*, *DirS*, *TwoStage*, *JM* and *AbsDis*. Most term proximity-based retrieval models have low retrieval bias; however, *AvgDist*, *AvgPairDist* and *SumMaxDist* exhibit high retrieval bias. The low-level retrieval features do not exhibit low retrieval bias, expect for *ntf(d,q)*, which has moderate retrieval bias.

5 Relationship between retrievability and effectiveness measures

So far, we examined the retrieval bias of different retrieval models. Our results show that the retrieval models differ substantially in terms of the retrieval biases that they impose on the population of documents. However, the question still remains, what is the relationship between minimizing the retrieval bias and maximizing a retrieval model's effectiveness?

In this section, we will now specifically examine to what extent the low or high retrieval bias of retrieval models correlates with their effectiveness. That is, if a retrieval model has less retrieval bias than other models, then does it also mean that it is more effective than the others models? If this holds true, then the retrievability will provide a valuable alternative for the automatic ranking of retrieval models in the case when there are no resource to relevance judgments available for a given collection. In order to examine these premises, we perform the following experiments.

1. In the first experiment, we rank all retrieval models on both measures independently and test to what extent the two rankings agree with each other, i.e., to what extent the low retrieval bias of a retrieval model leads to high effectiveness. We use this experiment as a baseline for other experiments where we make several attempts to increase the effectiveness of retrieval models on the basis of retrievability.
2. In the second experiment, we propose an approach for improving the retrievability of documents (minimizing retrieval bias). In this approach, rather than retrieving and ranking documents from a single collection, we first split the document collection into *low* and *high* retrievable partitions. Having split the collection into these two categories, we then perform retrieval by treating these classes as independent partitions and process queries independently for each partition and combine the result lists afterward. We can show that this helps in increasing overall retrievability, reducing the dominance of certain documents in query processing and thus reducing the bias of retrieval models. This approach thus provides a higher probability of being able to at least potentially find every document of the collection.
3. In the third experiment, we tune the parameter values of retrieval models over different ranges and examine their effect on both measures. The main motivation behind doing this is to thoroughly investigate whether some set of parameter values of a retrieval model have a strong effect on increasing/decreasing retrievability, we may then expect a similar behavior in the effectiveness of the retrieval model for these parameter settings. If this holds true, then retrievability will provide a valuable framework for tuning a retrieval model's effectiveness for a given collection by adjusting their parameter values over retrievability.
4. In the fourth experiment, we present a framework for automatically learning a good retrieval model with the combinations of machine learning and retrievability. This framework works on the basis of genetic programming to learn retrieval model by combining different types of evidences, including structure and proximity features [14,18,42,49]. This automatic method evolves by applying different genetic operations, such as crossover and mutation, on individuals of population over a series of generations. In each generation, retrievability measure is used as a fitness function for evaluating the fitness of the individuals in the solution space. This is different from previous studies where effectiveness measures such as precision, recall or MAP are used for evaluating the individual solution fitness [15,17,19,20,35,45]. These studies require resource to relevance judgments, while our framework is independent of such constraints. This allows us to optimize and fine tune a retrieval system for a specific document collection without the need for relevance judgments.

5.1 Relationship between two measures on the basis of retrieval models ranks

In this experiment, we compare the relationship between the two measures on the basis of retrieval model ranks. In this experiment, we want to examine to what extent the low retrieval bias of retrieval models leads to high effectiveness. In order to analyze this, we test and rank all retrieval models independently on both measures. Table 2 shows the retrieval bias and effectiveness scores, and Table 3 shows retrieval model ranks on both measures and the relationship between them. Although the relationship between two rank lists is not totally perfect, it can be observed from the results that the best retrieval models are consistently

Table 2 Retrieval bias (Gini coefficient) and effectiveness scores of different retrieval models on *TREC-CRT* collection

Retrieval model	Retrieval bias and effectiveness scores				
	G@100	Recall@100	Precision@30	MAP	b-pref
PairCoOccurrence	0.39	0.139	0.068	0.038	0.483
MinDistCount	0.45	0.156	0.085	0.045	0.483
CoOccurrence	0.49	0.147	0.067	0.038	0.483
BM25	0.52	0.156	0.101	0.049	0.428
SumMinDist	0.55	0.126	0.043	0.027	0.483
TwoStage	0.56	0.174	0.110	0.055	0.474
DirS	0.57	0.177	0.110	0.055	0.470
MinCover	0.58	0.130	0.044	0.027	0.483
AbsDis	0.60	0.170	0.108	0.052	0.440
JM	0.62	0.184	0.113	0.058	0.483
NormTFIDF	0.62	0.082	0.045	0.023	0.320
ntf(d,q)	0.63	0.107	0.061	0.028	0.470
AvgPairDist	0.66	0.134	0.047	0.028	0.483
AvgDist	0.67	0.106	0.033	0.023	0.483
SumMaxDist	0.68	0.107	0.033	0.023	0.483
$\|d\|$	0.74	0.001	0.000	0.000	0.256
sdf(d,q)	0.85	0.042	0.027	0.010	0.414
scf(d,q)	0.85	0.002	0.001	0.000	0.237
TFIDF	0.91	0.008	0.003	0.003	0.115
tf(d,q)	0.92	0.016	0.008	0.004	0.428
SMART	0.93	0.074	0.044	0.021	0.276
$\|T_d\|$	0.99	0.001	0.000	0.000	0.245

Retrieval models are ordered by increasing Gini coefficient scores

ranked in at least the top half of the ranking. This indicates somewhat systematic relationship between the retrievability and the effectiveness measures. On the basis of these rankings, if we compare only the standard and the language modeling-based retrieval models, then *BM25* and the four language modeling approaches (*JM*, *DirS*, *AbsDis*, and *TwoStage*) have higher effectiveness than other models possibly due to their lower retrieval bias. *NormTFIDF* has lower retrieval bias than *TFIDF* and *SMART*, and also, it has higher effectiveness. However, *NormTFIDF* has a higher retrieval bias than *BM25*, *JM*, *DirS*, *AbsDis* and *TwoStage* and also lower effectiveness than these models. If we focus only on the term proximity-based retrieval models, then *MinDistCount*, *CoOccurrence* and *PairCoOccurrence* have lower retrieval bias and higher effectiveness than other five term proximity-based retrieval models. *SumMaxDist*, *AvgDist* and *AvgPairDist* have higher retrieval bias than other term proximity-based retrieval models and also lower effectiveness than other models. In low-level retrieval features, *ntf(d,q)* has the lowest retrieval bias than other features, and this feature also has the highest effectiveness than other low-level features. The main reason behind the systematic relationship between the high retrieval bias and the low effectiveness may be the level of retrievability inequality between the documents. When relevant documents show low retrievability, then these are less likely to retrieved at top-ranked positions due to the presence of high retriev-

Table 3 Relationship between retrieval bias and effectiveness on *TREC-CRT* collection

Retrieval model	Correlation analysis				
	G@100(Rank)	Recall@100(Rank)	Precision@30(Rank)	MAP(Rank)	b-pref(Rank)
PairCoOccurrence	1	8	7	7	3
MinDistCount	2	5	6	6	2
CoOccurrence	3	7	8	8	4
BM25	4	6	5	5	14
SumMinDist	5	11	14	12	7
TwoStage	6	3	3	3	10
DirS	7	2	2	2	11
MinCover	8	10	12	11	6
AbsDis	9	4	4	4	13
JM	10	1	1	1	1
NormTFIDF	11	15	11	13	17
ntf(d,q)	12	12	9	9	12
AvgPairDist	13	9	10	10	5
AvgDist	14	14	16	15	9
SumMaxDist	15	13	15	14	8
$\|d\|$	16	21	21	21	19
sdf(d,q)	17	17	17	17	16
scf(d,q)	18	20	20	20	21
TFIDF	19	19	19	19	22
tf(d,q)	20	18	18	18	15
SMART	21	16	13	16	18
$\|T_d\|$	22	22	22	22	20

Retrieval models are ordered by increasing Gini coefficient ranks

able documents. This high level of retrievability inequality between documents decreases the overall effectiveness of retrieval models.

5.2 Decreasing retrieval bias by collection partitioning approach

The experiments above reveal that, indeed, a collection consists of documents that show highly different behavior in retrievability. Some documents are returned within the top-c results for a huge number of queries, possibly suppressing others that almost never show up within the top-c results for any query. This means that these documents are virtually inexistent for a searcher. One of the goals of recall-oriented retrieval domains is to ensure that all relevant documents are potentially found. We thus need to devise ways to ensure that the documents exhibiting low retrievability can also be retrieved by the queries that they are potentially relevant for. In order to do so, we propose to split a collection into two partitions, consisting of high and low retrievable documents. Having split the collection into these two partitions, we then perform retrieval by treating these partitions as independent collections. We process queries independently for each partition and subsequently combine the result lists by taking half documents from one partition and half documents from other partition. This ensures that the final result list will always include also documents having a low retrievability

Table 4 The retrieval bias of standard and partition-based retrieval strategies for the *TREC-CRT* collection

Retrieval model	Retrieval approach	High-quality queries			Medium-quality queries			Low-quality queries		
		$c=50$	$c=100$	$c=250$	$c=50$	$c=100$	$c=250$	$c=50$	$c=100$	$c=250$
NormTFIDF	*Partition*	**0.54**	**0.54**	0.53	**0.57**	**0.56**	**0.55**	**0.74**	**0.71**	**0.68**
	Standard	0.70	0.62	**0.51**	0.75	0.67	0.55	0.93	0.89	0.82
BM25	*Partition*	**0.51**	0.50	0.50	**0.53**	**0.52**	0.51	**0.71**	**0.68**	**0.64**
	Standard	0.57	**0.52**	**0.44**	0.62	0.56	**0.48**	0.87	0.83	0.75
DirS	*Partition*	**0.58**	**0.55**	**0.54**	**0.62**	0.61	0.59	**0.75**	**0.73**	0.69
	Standard	0.60	0.60	0.58	0.66	**0.60**	**0.52**	0.82	0.76	**0.67**
JM	*Partition*	**0.59**	**0.59**	0.58	**0.60**	**0.59**	0.58	**0.75**	**0.72**	**0.69**
	Standard	0.68	0.62	**0.51**	0.71	0.64	**0.53**	0.84	0.79	0.70
AbsDis	*Partition*	**0.58**	**0.57**	0.56	**0.59**	**0.58**	0.56	**0.73**	**0.70**	**0.67**
	Standard	0.66	0.60	**0.50**	0.69	0.62	**0.53**	0.84	0.79	0.71
TwoStage	*Partition*	**0.53**	**0.53**	0.53	**0.56**	**0.55**	0.54	**0.72**	**0.70**	**0.66**
	Standard	0.64	0.56	**0.46**	0.68	0.60	**0.50**	0.85	0.79	0.68
TFIDF	*Partition*	**0.63**	**0.62**	**0.62**	**0.64**	**0.63**	**0.63**	**0.72**	**0.69**	**0.66**
	Standard	0.95	0.91	0.81	0.96	0.93	0.86	0.99	0.99	0.97
SMART	*Partition*	**0.66**	**0.64**	**0.63**	**0.67**	**0.66**	**0.64**	**0.81**	**0.79**	**0.76**
	Standard	0.96	0.93	0.87	0.96	0.93	0.86	0.96	0.93	0.85

Results in bold show retrieval models have less retrieval bias than other models
Partition reflects partition-based retrieval strategy, and *Standard* reflects standard retrieval strategy

score, i.e., that would rarely or never have been returned within a certain rank cutoff in a standard retrieval setting independent of collection partition.

Table 4 lists the Gini coefficient scores for a range of rank cutoff factors on different query sets.[2] For each query, we merge the results from both partitions on the basis of percentage of documents retrieved from the partitions relative to total number of documents retrieved from both partitions. This always includes documents having low retrievability score in the final results list. This reduces the dominance of certain documents in query processing and thus reduces the bias of retrieval models as compared to the standard retrieval approach as clearly visible from the results of Table 4. On high-quality queries, both retrieval strategies have comparable retrievability effectiveness; however, as the query quality decreases, the partition-based retrieval achieves significant improvement over the standard retrieval strategy due to high retrieval bias exhibited by the standard retrieval strategy.

From Table 4 results, we can observe that the partition-based retrieval (PBR) approach significantly reduces the retrieval bias for most of retrieval models. However, contrary to our hypothesis on effectiveness measures, we do not find any significant improvement with PBR (see Table 5). In most of cases, PBR even hurts the effectiveness of retrieval models. In order to find the reason behind this, we take a look on the probability distribution of the relevance judgments of the *TREC-CRT* collection relative to document length. This will help us to find out any amount of bias in the relevance judgments toward long or short documents. Since long documents cover more query material than short documents, thus the probability of

[2] For generating query sets, we first order all queries in Q on the basis of simplified query clarity score (SCS) [16]. Then, we extract three query subsets from Q. First from low SCS range (low-quality queries), second from middle SCS range (medium-quality queries) and third from high SCS range (high-quality queries).

Table 5 The effectiveness analysis of standard and partition-based retrieval strategies for the *TREC-CRT* collection

Retrieval model	Recall@100		Precision@30	
	Standard	Partition	Standard	Partition
NormTFIDF	**0.057**	0.059	**0.032**	0.034
BM25	**0.121**	0.115	**0.079**	0.074
DirS	**0.144**	0.138	**0.093**	0.088
JM	**0.148**	0.141	**0.094**	0.088
AbsDis	**0.145**	0.138	**0.094**	0.088
TwoStage	**0.141**	0.135	**0.089**	0.084
TFIDF	0.015	**0.020**	0.006	**0.009**
SMART	0.036	**0.038**	0.016	**0.017**

Results in bold show retrieval models have less retrieval bias than other models
Partition reflects partition-based retrieval strategy, and *Standard* reflects standard retrieval strategy

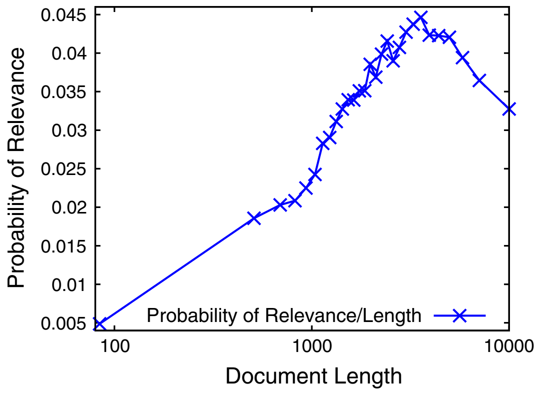

Fig. 1 Probability of relevance relative to document length for the *TREC-CRT* collection

relevance increases as the document length increases. This hypothesis has been supported by a number of empirical studies investigating the relationship between the document relevance and the length [26,28,30,39] on a variety of TREC collections. For the sake of simplicity, we call this bias "relevance bias". Figure 1 shows the probability distribution of relevance judgments relative to document length for the *TREC-CRT* collection. The distribution of Fig. 1 also supports the above hypothesis. The probability of relevance increases as the document length increases. This indicates that in order to achieve high effectiveness, it is necessary to provide high retrieval likelihood for long documents. This would achieve high effectiveness, but hurts the retrieval bias by not providing equal access to all documents of the collection. Please note that the above analysis does not allow any statement with respect to the quality of this length-based relevance bias. It may be a real, factual shift of relevance in long documents, but may also stem from the way relevance judgments are created when this process of relevance judgment creation may involve retrieval models of some kind, that exhibit their own bias, we may see the resulting bias reflected in the ground truth. This may be particularly true for pooling approaches obtain for relevance assessments.

Based on the above observation, we can conclude that the relationship between the retrievability and the effectiveness depends partially upon the amount of relevance bias in the relevance judgments. If this relevance bias is low, then we can expect that decreasing retrieval bias would also support achieving high effectiveness. However, if this relevance bias is high toward either long or short documents, then decreasing retrieval bias after a certain level will hurt the effectiveness.

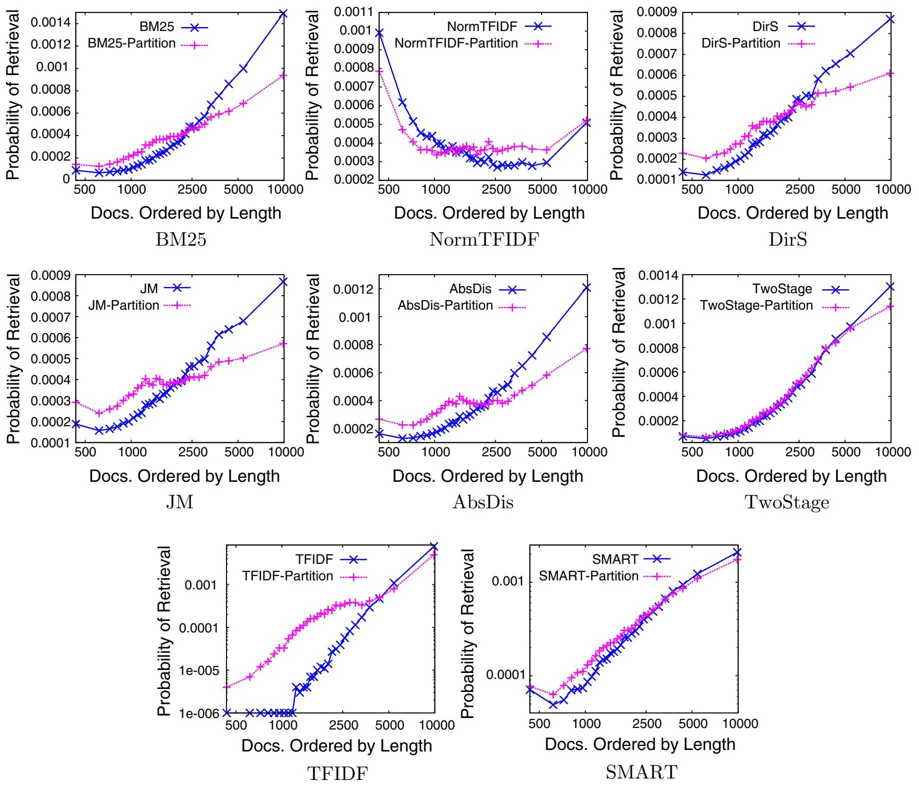

Fig. 2 Difference between the standard retrieval and the partition-based retrieval strategies on the basis of retrieval likehood relative to document length

Now, using the above observation, we compare the effectiveness of PBR and the standard retrieval approach on the basis of retrieval likehood and relevance likehood. We follow the binning analysis strategy proposed in [39] and plot the two patterns of likehood relative to document length. We order all documents of the *TREC-CRT* collection on the basis of their lengths and divide them into 400 equal sized bins. We then compute the probability of the top 100 retrieved documents of each query belonging to a certain bin. This gives us the conditional probability of retrieval $P(d \in Bin_i | d\ is\ retrieved)$ for a particular retrieval strategy. We then compare this retrieval likehood pattern with the relevance likehood pattern. This reveals important information about the term weighting strategy of different retrieval models. From the Fig. 2 results, we can observe that

- *TFIDF* gains significant improvements with PBR on both effectiveness and retrieval bias measures. This is because *TFIDF* does not normalize query term frequencies relative to document length. It thus has a high bias toward long documents in case of standard retrieval approach (see Fig. 2). PBR reduces this dominance of long documents and increases the retrieval likehood of short documents in the low retrievability partition. This increases both the effectiveness and the retrievability.
- *NormTFIDF* normalizes query term frequencies relative to document length. This increases the retrieval likehood of short documents. However, since the relevance judgments are biased toward long documents, with standard retrieval approach we could not

find any significant improvement on the effectiveness. PBR attempts to reduce this dominance. After splitting also does not increase retrieval likelihood, as after partitioning, *NormTFIDF* again starts to increase the retrieval likehood of short documents in both partitions. Yet, we can find some retrieval increment for the middle length documents. These are the document of low retrieval partition, and due to query terms normalization, *NormTFIDF* increases their likehood of retrieval by a large percentage than the long documents.

- *BM25*, *JM*, *AbsDis*, *DirS* and *TwoStage* control query term frequencies normalization with the help of parameters. These models also do not over penalize the short documents as we observed in the case of *TFIDF*. Now, if we look at the results of these models, then PBR significantly hurts the effectiveness. This is because it reduces the retrieval likehood of long documents (see Fig. 2 results) in the form of providing equal access to all documents. The retrieval likehood curves of all these models with the PBR are less skewed than the curves of standard retrieval-based strategy. Due to this fact, the relevance bias that is necessary in order to achieve high effectiveness is penalized.

On the basis of these observations, we can conclude that the retrievability measure has a strong relationship with the effectiveness. However, the strength of this relationship depends upon the bias of relevance judgments. If the relevance judgments are biased toward either long or short documents, then retrievability serves as an indicator only up to a certain level.

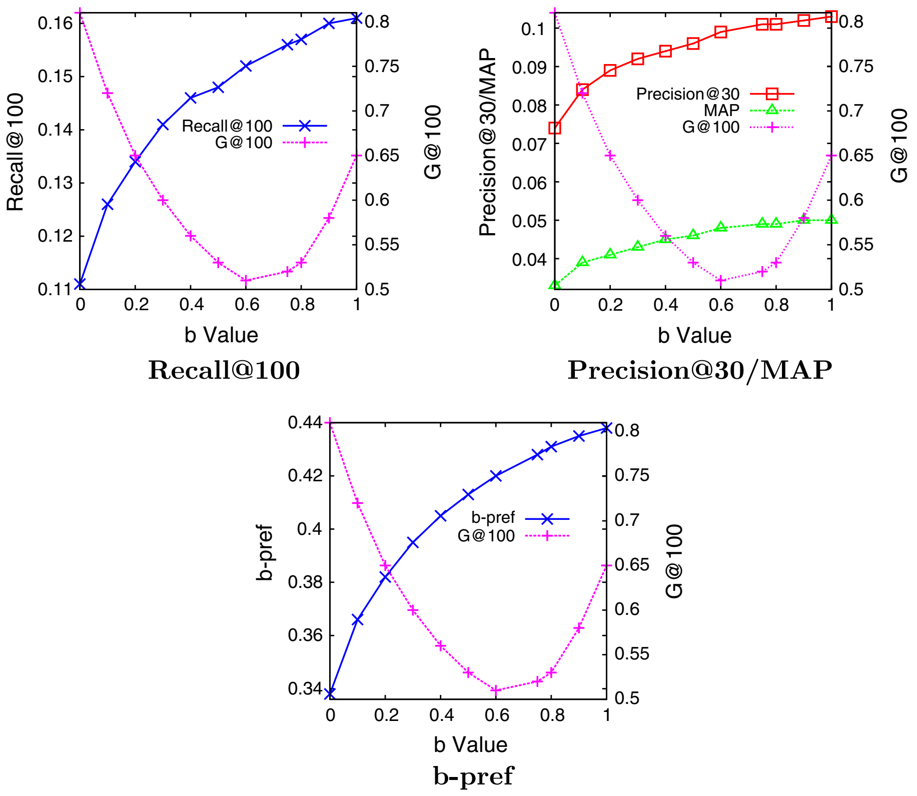

Fig. 3 Graphical relationship between the retrieval bias and the effectiveness across various parameter (*b*) values of *BM25*

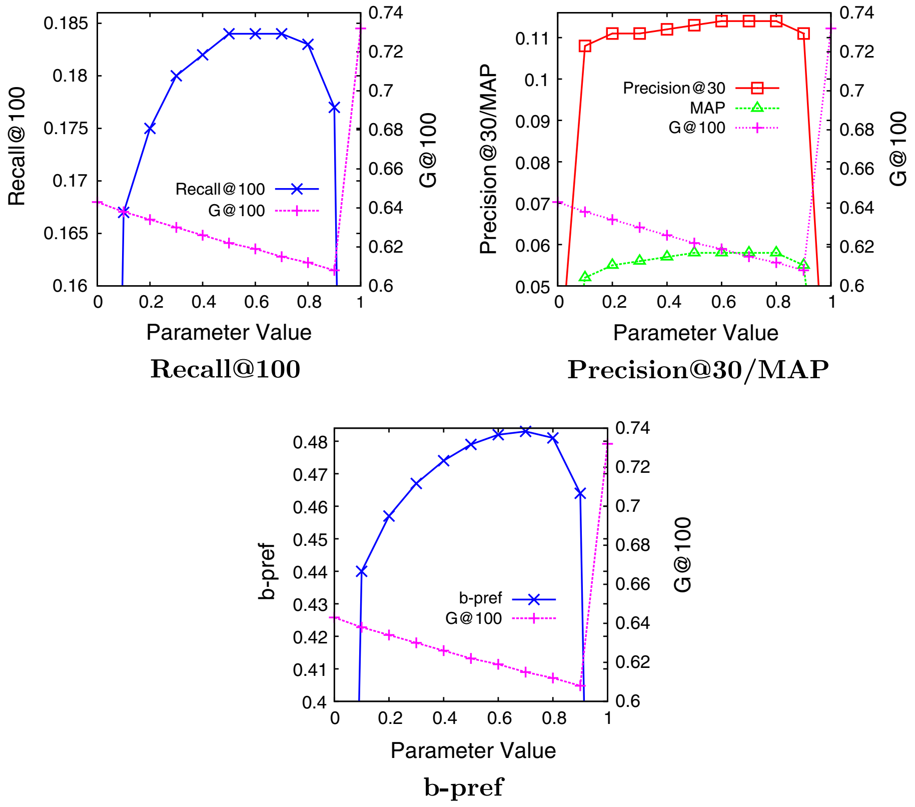

Fig. 4 Graphical relationship between the retrieval bias and the effectiveness across various parameter (λ) values of *JM*

This level reflects the relationship between the probability of retrieval likehood and the probability of relevance likehood. After this threshold, optimizing for retrievability would start penalizing the effectiveness due to providing equal access to all documents.

5.3 Increasing effectiveness of retrieval models by tuning their parameters over retrieval bias

Most of retrieval models are tuned with the help of parameter values. These parameters either control the query terms normalization relative to document length or smooth the document relevance scores in case of unseen query terms. In this experiment, we tune the parameter values of different retrieval models over specified ranges and examine their sensitivity and change with both measures (effectiveness and retrieval bias). Four language modeling approaches with term smoothing (*JM*, *DirS*, *AbsDis* and *TwoStage*) along with *BM25* are used for this purpose. In case of *BM25*, *JM* and *TwoStage*, the parameters b and λ are varied from 0.1 to 1.0 in steps of 0.1, while the parameter μ in case of *DirS* is varied from 500 to 10,000 in steps of 1,000. Figures 3, 4, 5, 6 and 7 show the effect of parameter values on both measures. We can observe that all those parameter value settings that exhibit high retrieval bias do not correspond to the maximum effectiveness. The maximum effectiveness is gained

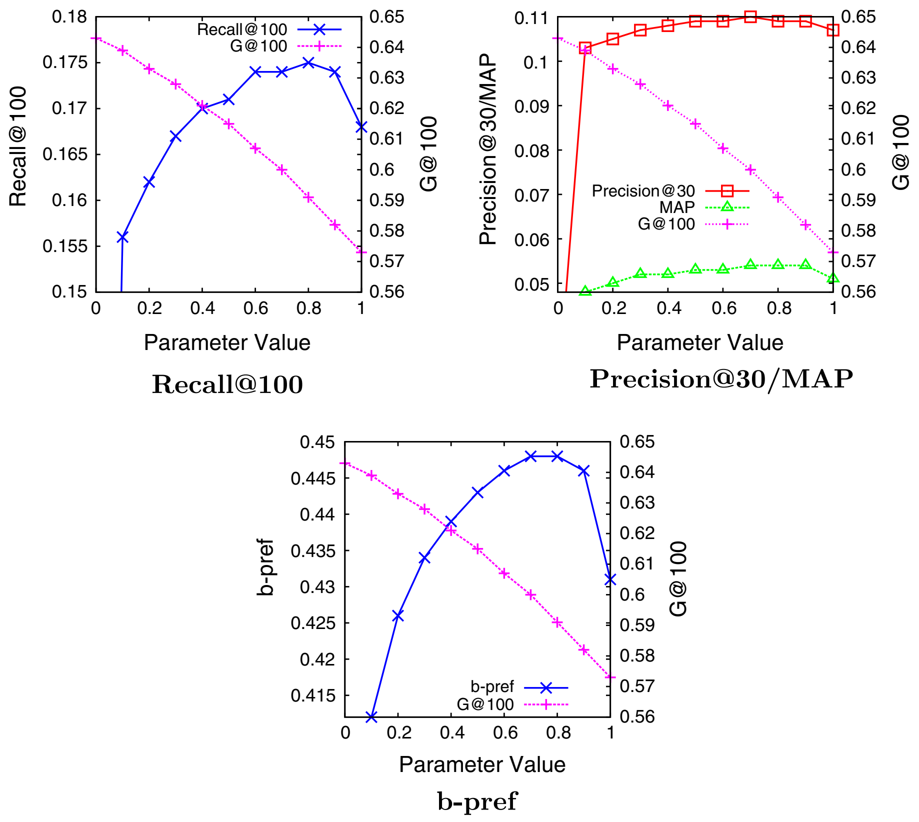

Fig. 5 Graphical relationship between the retrieval bias and the effectiveness across various parameter (δ) values of *AbsDis*

only when the parameter values result in low retrieval bias. Along with the parameter values that exhibit low retrieval bias and high effectiveness, there exist also some parameter values that, while achieving low retrieval bias, also hurt the effectiveness by a small fraction. This decrease in the effectiveness occurs due to the relevance bias on long documents, while the tuned retrieval models aim at providing equal access to all documents. These findings again indicate the presence of a strong relationship between the Gini coefficient and Precision@30, Recall@100, MAP, and b-pref, representing effectiveness of retrieval models.

6 Evolving robust retrieval model using genetic programming and retrievability

The results so far confirm a significant correlation between the retrievability and the effectiveness. In this section, we will now utilize this correlation for automatically producing effective retrieval models with the combination of genetic programming and retrieval bias. Automatically evolving effective retrieval models with the help of genetic programming (GP) is not a new research idea. In the past few years, there have been several attempts on this research [15,17,19,20,35,44,45]. However, in all of these studies, the fitness of the

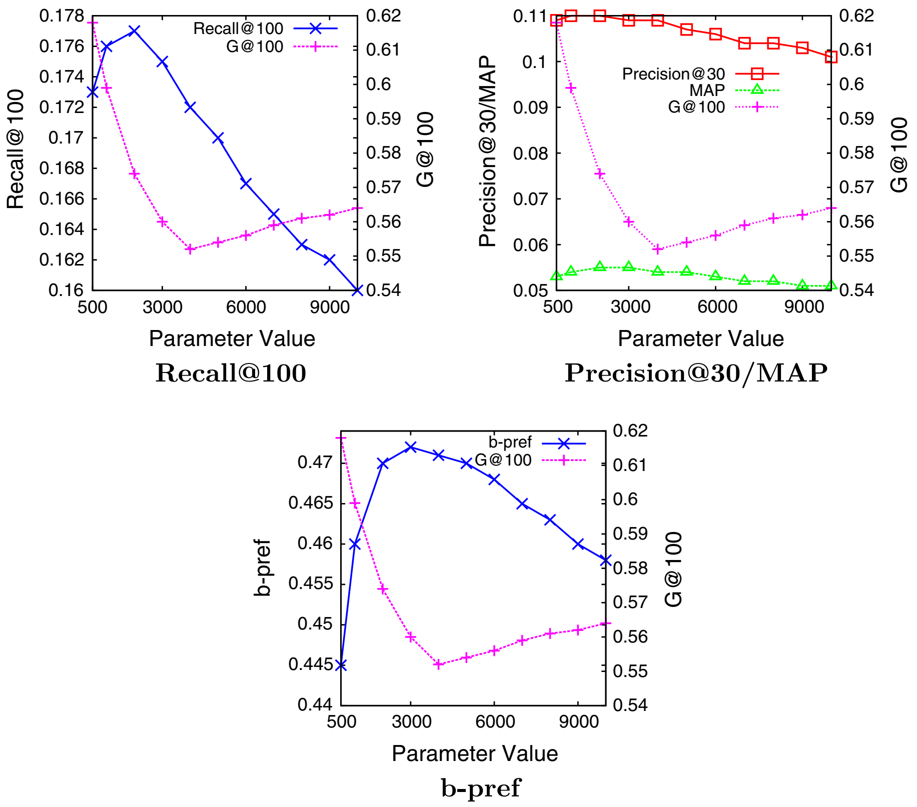

Fig. 6 Graphical relationship between the retrieval bias and the effectiveness across various parameter (μ) values of *DirS*

GP-evolved solutions is checked with the help of effectiveness measures (precision, MAP, recall, etc.). These studies depend on relevance judgments (ground truth) being available. By relying on the $\hat{r}(d)$ (retrievability scores of documents), our approach does not require any relevance judgments. An effective retrieval model evolves by applying genetic operations, such as crossover and mutation on the individuals of populations over a series of generations. In each generation, retrieval bias is used as a fitness function for evaluating the fitness of individuals in the population. After convergence of the GP process, analysis of the relationship between retrievability and effectiveness measures reveals that those individuals (heuristically modified retrieval models) that perform well on the retrievability measure have also good performance on the effectiveness measures.

The learning process is formalized as follows. Given a query collection Q and a document collection D, a feature extractor produces a vector of features that describe the match between the relevant documents and the queries in Q. The range of features that are used in this study includes both the classical IR features as well as recently developed proximity-based features. During the evaluation process, GP tries to optimize a retrieval model $\widehat{f}$ such that the access of $\widehat{f}$ over all documents of the collection can be maximized (increasing retrievability). Once all the generations finish their processing, the individual having the minimum retrieval bias is returned as an output.

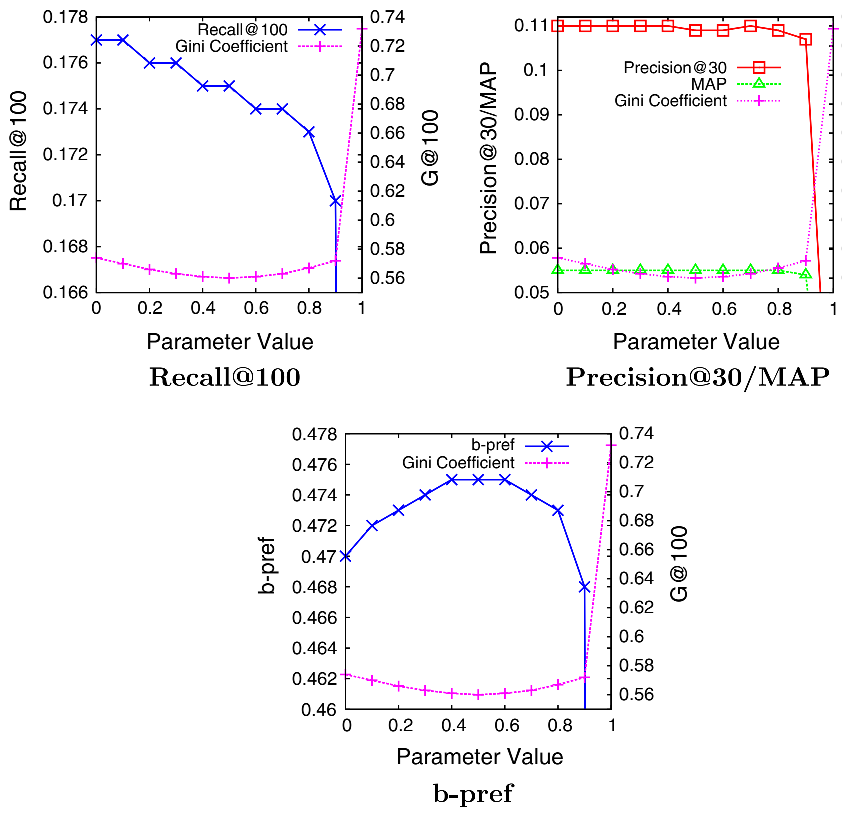

Fig. 7 Graphical relationship between the retrieval bias and the effectiveness across various parameter (λ) values of *TwoStage*

6.1 Genetic programming framework

Genetic programming is a branch of evolutionary computing. It helps to solve exhaustive search space problems without requiring the user to specify the structure of the solution in advance. There are two main steps in genetic programming: (a) initial population creation and (b) recombination with the existing population to evolve better solutions. Generally, the initial population (generation) is created randomly, and it is modeled in the form of trees. Each tree represents a solution, structured by several nodes. Nodes can be either operators (functions) or operands (terminals). From the initial population, recombination occurs to evolve better solutions (next generation's population). This is performed by either a crossover or a mutation. In order to produce better populations, it is important to select better solutions from the current population in a larger percentage. This selection is done by a fitness function that measures how well an individual performs in its environment. The process of recombination iterates until a predefined number of generations has been created or no further improvements can be observed. Some important parameters in GP are as follows: (a) the population size, (b) the number of generations, (c) the depth of tree, (d) the function set and (e) the terminal set.

The proposed GP-based learning framework is summarized as follows. An individual I of the current population represents a retrieval model. It is expressed as a functional expression of three components: Sv (retrieval features), Sc (constants) and Sop (operators). Sv is a set of retrieval features. Sc is a set of predefined real numbers ranging from 0 to 1. Sop is a set of arithmetic operators ($+$, $/$, $*$). In the implementation, I is represented as a binary tree structure, in which an internal node is an arithmetic operator and a leaf node is a ranking feature. In experiments, we explore the maximum depth of individuals up to 6 levels. Furthermore, we perform experiments with up to 100 generations and with 50 individuals per generation.

The evolution process works as follows. Individuals of the initial population are produced randomly by a ramped half-and-half method [27]. In this method, the individuals are produced randomly. However, it ensures that half of the individuals must not have all the branches of the maximum tree depth. For the reproduction of new individuals for the next generation, the top 10 % individuals of the current generation that exhibit minimum retrieval bias are moved to the next generation without any modification. This is done for the survival of the fittest individuals. Next, the remaining population is produced 70 % by crossover and 30 % by mutation. Parents for crossover are selected with the 5-tournament selection approach. This removes any kind of bias in the parents' selection process. The 5-tournament selection approach randomly selects a few individuals from the previous generation and returns one individual (for the mutation) or two individuals (for the crossover). Crossover is applied on the parents by simply switching their sub-trees to each other. The sub-trees for the crossover are also selected randomly. In this process, two new individuals are produced. In mutation, a mutant is created by randomly choosing an internal node of the selected individual, and then, its whole sub-tree is replaced with a randomly generated tree. After the evolution ends its processing, the output set O representing the best individuals is returned as a candidate solutions.

6.2 Retrieval features

Formally, a feature f_k is a function that takes a query and the document $f_k(d, q)$, and outputs documents' relevance score. The features that we use in the GP-based learning include a set of low-level features (Sect. 4.4) and the term proximity-based features (Sect. 4.5) proposed recently in [14,18,42,49].

Before starting the processing of GP, a query-based normalization on all features is performed in order to normalize all feature values into a range of [0, 1]. For a query q, the normalized value of $f_k(d, q)$ is calculated by Eq. 17, where $max(f_k(d, q))$ and $min(f_k(d, q))$ are the maximum and minimum values of $f_k(d, q)$, respectively, for all $d \in D$ for feature f_k.

$$f_k(d, q) = \frac{f_k(d, q) - min(f_k(d, q))}{max(f_k(d, q)) - min(f_k(d, q))} \tag{17}$$

6.3 Experiments

We evaluate two correlations (using Spearman's rank correlation coefficient) in order to determine whether the GP approach presented above can be utilized effectively for evolving retrieval models on the basis of retrievability measure without relying on the relevance judgments.[3]

- First, we analyze the relationship between the retrievability and the effectiveness measures over the fittest individual of each generation. This allows us to verify that the hypothesis

[3] The complete log of genetic programming and optimized retrieval models up to 100 generation is available at http://www.ifs.tuwien.ac.at/~bashir/Relationhip_Retrievability_Effectiveness.htm.

behind our GP-based approach is actually holding, as well as indicates any improvement over retrievability and effectiveness.

- Second, we analyze the relationship between the retrievability and the effectiveness over the average fitness of each generation. This ensures that the correlation hypothesis between the retrievability and the effectiveness holds in general.

Figure 8 shows the improvement in decreasing retrieval bias and increasing effectiveness that is gained by the fittest individual of each generation. As expected, retrieval bias decreases in a nonstrictly monotonic manner as the generations evolve. Figure 9 shows how the average fitness improves within each generation, as well as the associated gain in effectiveness measures. Within the first few generations, a large number of individuals yield only nonsensical weights. This gives poor average retrieval bias and effectiveness. These individuals quickly die out, resulting in the dramatic improvements on average fitness for the first few generations. Once the system stabilizes, average fitness rises very slowly over the course of a large number of generations. Over the last few generations, we can observe a considerable decrease in the retrieval bias and increase in the effectiveness. However, suddenly generations having low retrieval bias start hurting the effectiveness. This happens due to the relevance bias in the form of providing high level of equal access to all documents. This can be also observed from the Fig. 8 results, where the fittest individuals from generations 55–75 have lowest retrieval bias and higher effectiveness than the fittest individuals from generations 85–100. Overall, generations with high average Gini coefficients (i.e., high retrieval bias) show low effectiveness, while generations with low average Gini coefficients show high effectiveness. This is a strong indicator that, at least for the recall-oriented applications, a strong retrieval bias (rendering many documents virtually unfindable or hardly findable) has a significant impact on the effectiveness. Table 6 shows the correlation between the effectiveness measures and two retrievability scoring functions on the basis of average fittest for each generation. The fittest solution that evolved using $r(d)$ as a retrievability scoring function has effectiveness scores ($Recall@100 = 0.146$), ($Precision@30 = 0.096$), ($MAP = 0.045$) and ($b - pref = 0.417$), while with $\hat{r}(d)$ it has effectiveness scores ($Recall@100 = 0.157$), ($Precision@30 = 0.107$), ($MAP = 0.052$) and ($b - pref = 0.450$). Tables 7 and 8 list the three best retrieval models that evolved with genetic programming guided by $\hat{r}(d)$ and $r(d)$. The effectiveness of these retrieval models is given in Tables 9 and 10.

If we compare the effectiveness of the fittest GP-evolved solution with the term proximity-based retrieval models of Sect. 4.5 and the low level features of IR (Sect. 4.4), then the previous effectiveness with the lowest retrieval bias model (*PairCoOccurrrence*) of Table 2 results was ($Recall@100 = 0.139$), ($Precision@30 = 0.068$), ($MAP = 0.038$), ($b - pref = 0.483$). The GP improves this to ($Recall@100 = 0.157$), ($Precision@30 = 0.107$), ($MAP = 0.052$) and ($b - pref = 0.450$) with the fittest individual (see Table 9) with ($Recall@100 = +13\,\%$), ($Precision@30 = +57\,\%$), ($MAP = +37\,\%$) and ($b - pref = -7\,\%$) improvement. Note that this improvement happened without any use of ground truth relevance assessments.

7 Conclusion

Retrievability measures, to what extent a retrieval model provides theoretically equal access to all documents, i.e., returns all documents with equal likelihood if all possible queries are posed against a specific document collection. The studies presented in this paper reveal that at least for recall-oriented application domains, where users are willing to proceed down to 50, 100

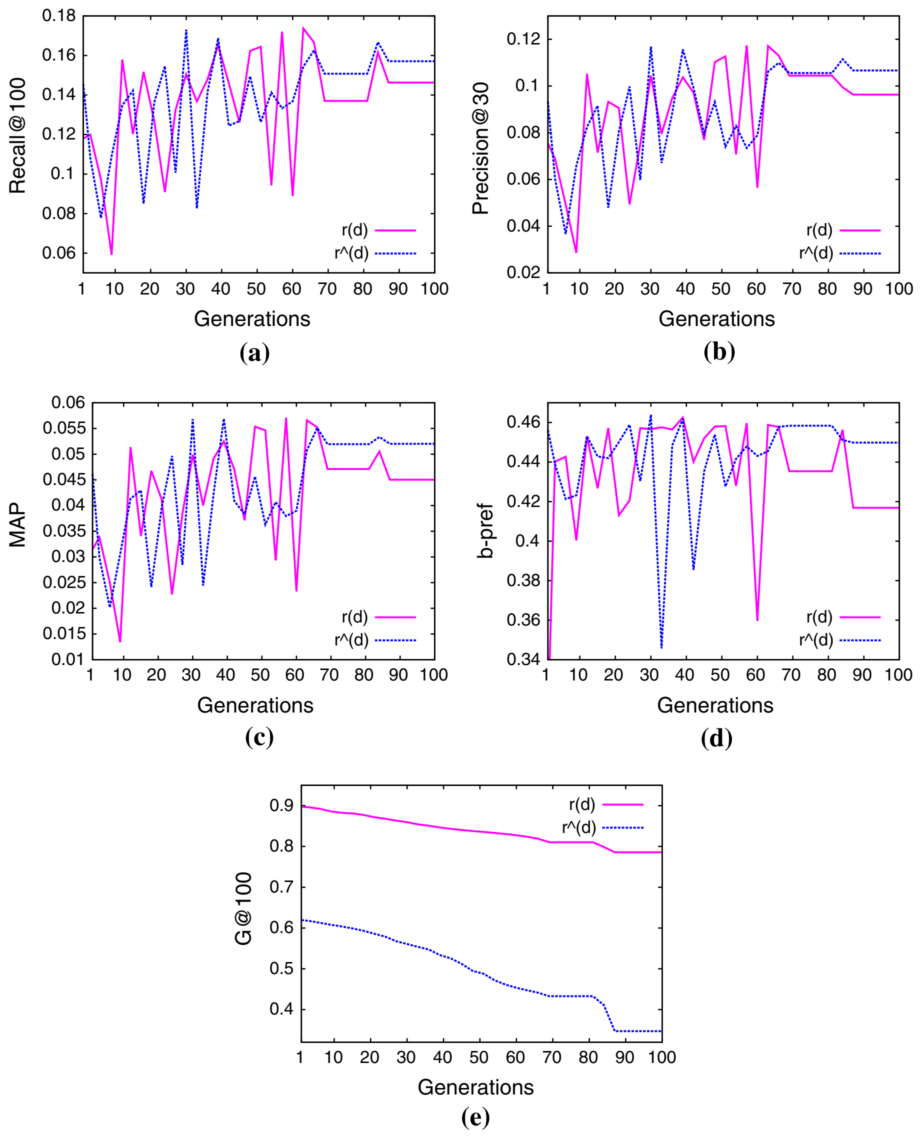

Fig. 8 Improvement gained over retrieval bias and effectiveness measures (Recall@100, Precision@30, MAP and b-pref) by the fittest individuals of each generation. **a**–**d** Show improvement gained over the effectiveness measures. **e** Shows improvement gained over the retrieval bias by the fittest individual. **a** Recall@100, **b** Precison@30, **c** MAP, **d** b-pref and **e** Retrieval Bias ($G@100$)

or 250 items in a ranked search result list, there is a high correlation between the effectiveness of a retrieval model measured in precision, recall, or MAP and retrievability. This indicates that retrieval models may be tuned using retrievability as a guiding measure, rather than more conventional approaches of tuning through effectiveness measures. The advantage of relying on retrievability is that it does not require any ground truth to be provided for tuning. Thus, a retrieval model's parameters may be tuned toward a given document collection and

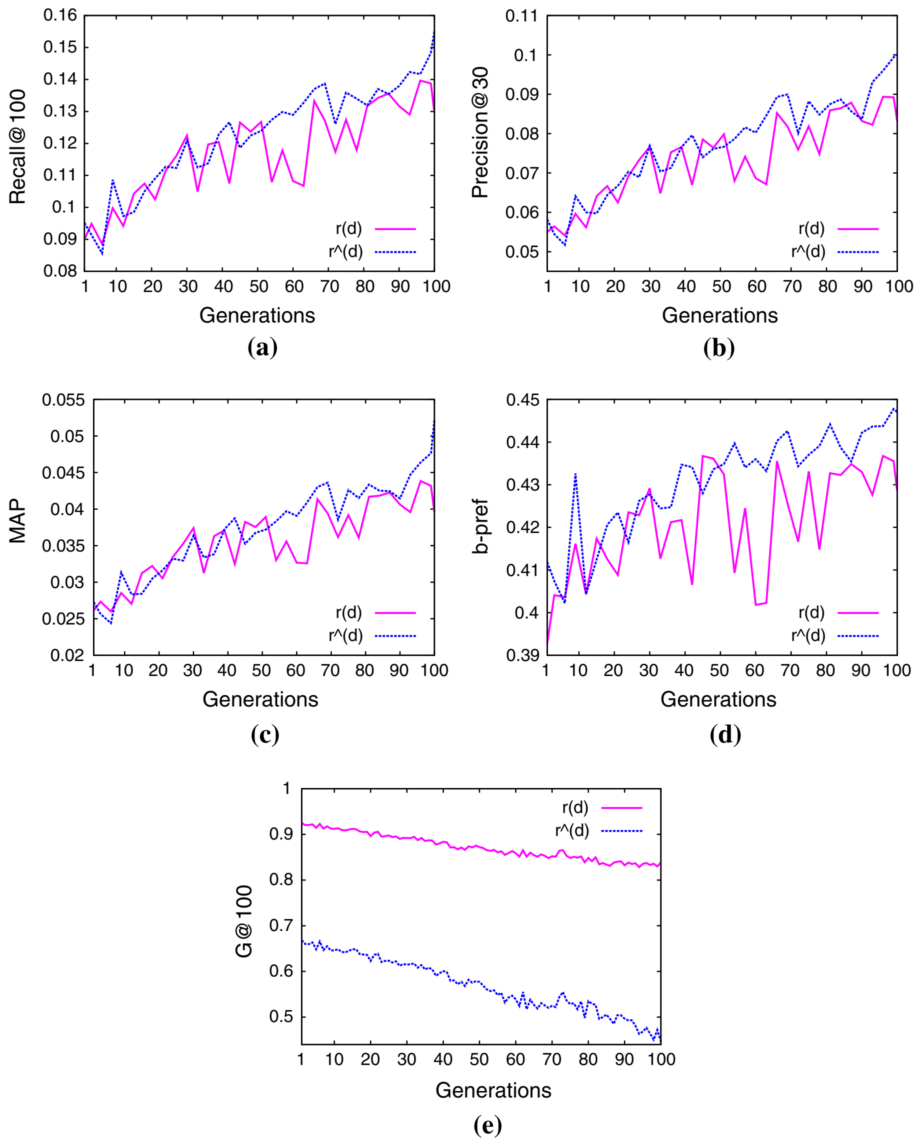

Fig. 9 Improvement gained over retrieval bias and effectiveness measures (Recall@100, Precision@30, MAP and b-pref) on the basis of average (retrieval bias and effectiveness) scores of all the individuals in each generation as the generations progress. **a**–**d** Show improvement gained over the effectiveness measures. **e** Shows improvement gained over the retrieval bias by the fittest individual. **a** Recall@100, **b** Precison@30, **c** MAP, **d** b-pref, **e** Retrieval Bias (*G*@100)

its characteristics (such as spread in the vocabulary richness and document length) without having a dedicated set of labeled ground truth available in each specific setting. Experiments conducted on the TREC Chemical Retrieval Track 2009 dataset yield the following findings.

- When analyzing the relationship between retrievability and effectiveness by ranking all retrieval models independently on both measures, results indicate a significant correlation

Table 6 The correlation between the retrieval bias and the effectiveness on the basis of average fittest individual of each generation with $r(d)$ and $\hat{r}(d)$

Effectiveness measures	$G@100 with r(d)$	$G@100 with \hat{r}(d)$
Recall@100	−0.76	−0.83
Precision@30	−0.77	−0.82
MAP	−0.75	−0.85
b-pref	−0.63	−0.79

$G@100$ refers to Gini coefficient (retrieval bias) with the rank cutoff factor 100. High negative correlation indicates that the retrieval models having less retrieval bias generate high effectiveness. The correlation values are computed with the Spearman's rank correlation coefficient

Table 7 A list of three best retrieval models evolved with the genetic programming, and when $\hat{r}(d)$ is used for scoring document retrievability

Number	Retrieval model
1	$\widehat{f1}(d,q) = ((\lvert d\rvert + (\lvert d\rvert * 0.6)) + ((\frac{ntf(q,d)}{(tf(q,d)+f2(q,d))} + f4(q,d)) + \lvert d\rvert))$
2	$\widehat{f2}(d,q) = ((\lvert d\rvert + \frac{\frac{\lvert d\rvert}{f2(q,d)}}{\frac{(0.8+0.2)}{(f4(q,d)+f4(q,d))}}) + (((ntf(q,d) * 0.4) + f4(q,d)) + \lvert d\rvert))$
3	$\widehat{f3}(d,q) = ((\lvert d\rvert + \frac{\frac{\lvert d\rvert}{f2(q,d)}}{\frac{(0.9+tf(q,d))}{(\lvert T_d\rvert+sdf(q,d))}}) + (((ntf(q,d) * 0.4) + f4(q,d)) + \lvert d\rvert))$

Table 8 A list of three best retrieval models evolved with the genetic programming, and $r(d)$ is used for scoring document retrievability

Number	Retrieval model
1	$\widehat{f1}(d,q) = (((f4(q,d) * f4(q,d)) + ((\lvert d\rvert + 0.8) + (ntf(q,d) * f4(q,d))))+ (\frac{\frac{(scf(q,d)+tf(q,d))}{ntf(q,d)}}{\lvert d\rvert} + ((\lvert d\rvert * \lvert d\rvert) * (ntf(q,d) * f4(q,d)))))$
2	$\widehat{f2}(d,q) = (((f4(q,d) * f4(q,d)) + ((\lvert d\rvert + 0.2) + (ntf(q,d) * f4(q,d))))+ (\frac{((\lvert d\rvert * f8(q,d))+sdf(q,d))}{f4(q,d)} + ((\lvert d\rvert * \lvert d\rvert) * (ntf(q,d) * f4(q,d)))))$
3	$\widehat{f3}(d,q) = (((f4(q,d) * f4(q,d)) + ((\lvert d\rvert + 0.2) + (ntf(q,d) * f4(q,d))))+ (\frac{((scf(q,d)*0.4)+sdf(q,d))}{f4(q,d)} + ((\lvert d\rvert * \lvert d\rvert) * (ntf(q,d) * f4(q,d)))))$

between these two ranking. Although the correlation was not perfect, retrieval models having low retrieval bias are consistently ranked in at least the top half of the ranking. This reveals that on the basis of retrievability measure, it is possible to automatically rank retrieval models.

- After the correlation analysis, we made two attempts to optimize a retrieval model's effectiveness on the basis of retrievability. These include changing the parameter values of retrieval models over retrievability and partitioning the collection based on low and high retrievability of documents. The analysis on changing the parameter values is promising, and analyses on the partitioning the collection reveal interesting findings for understanding the extent of this relationship. It seems to depend upon the bias in the relevance judgments. If the relevance judgments will be biased toward either long or short documents, then

Table 9 Effectiveness of three best retrieval models evolved with the genetic programming and $\hat{r}(d)$

Retrieval model	G@100	Recall@100	Precision@30	MAP	b-pref
$\widehat{f1}(d,q)$	0.347	0.157	0.107	0.052	0.450
$\widehat{f2}(d,q)$	0.359	0.127	0.089	0.039	0.419
$\widehat{f3}(d,q)$	0.365	0.150	0.103	0.050	0.446

G refers to Gini coefficient

Table 10 Effectiveness of three best retrieval models evolved with the genetic programming and $r(d)$

Retrieval model	G@100	Recall@100	Precision@30	MAP	b-pref
$\widehat{f1}(d,q)$	0.786	0.146	0.096	0.045	0.417
$\widehat{f2}(d,q)$	0.786	0.133	0.086	0.039	0.420
$\widehat{f3}(d,q)$	0.787	0.133	0.080	0.038	0.422

G refers to Gini coefficient

decreasing the retrieval bias of a retrieval model provides benefit in terms of effectiveness up to only a certain level. This level reflects the relationship between the probability of the retrieval likehood of the retrieving strategy and the probability of relevance. After this threshold, decreasing the retrieval bias starts penalizing the effectiveness of the retrieval model due to providing equal access to all documents of the collection.

- In subsequent experiments, we further show that this principle may even be applied to learn completely new retrieval models using a genetic programming approach, where the fitness function is guided solely by the evolving model's retrieval bias. This offers a promising approach for fine-tuning and optimizing retrieval models for a specific document collection and its characteristics without having to invest enormous amounts of effort into manually labeling the relevant documents of a collection, assessing their relevance.

Future work will focus on obtaining a better understanding of the specific characteristics of a document collection, its representation in feature space and the space spanned by the query representation. This will provide a clearer picture on a formal level of how a retrieval model may be tuned and specifically in which situations a high retrieval bias may be expected. We further want to investigate, to what extent similar approaches can be used to tune retrieval models for precision-oriented settings and for other TREC datasets and to what extent unbiased access toward all documents in a collection with respect to the features selected to represent them (that is when certain document characteristics such as unequal document lengths lead to unexpected biases rather than using this as an explicit feature favoring shorter results as more precise) is a desirable and beneficial characteristic of a retrieval model across a range of application domains.

References

1. Amitay E, Carmel D, Lempel R, Soffer A (2004) Scaling ir-system evaluation using term relevance sets. In: SIGIR '04: proceedings of the 27th annual international ACM SIGIR conference on research and development in, information retrieval, pp 10–17
2. Aslam JA, Savell R (2003) On the effectiveness of evaluating retrieval systems in the absence of relevance judgments. In: SIGIR'03: proceedings of the 26th international ACM SIGIR conference on research and development in, information retrieval, pp 361–362
3. Azzopardi L, Bache R (2010) On the relationship between effectiveness and accessibility. In: SIGIR '10: proceeding of the 33rd annual international ACM SIGIR conference on research and development in information retrieval. Geneva, Switzerland, pp 889–890

4. Azzopardi L, Owens C (2009) Search engine predilection towards news media providers. In: SIGIR '09: proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval. Boston, MA, USA, pp 774–775
5. Azzopardi L, Vinay V (2008) Retrievability: an evaluation measure for higher order information access tasks. In: CIKM '08: proceeding of the 17th ACM conference on information and knowledge management. Napa Valley, CA, USA, pp 561–570
6. Baccini A, Déjean S, Lafage L, Mothe J (2012) How many performance measures to evaluate information retrieval systems? In, Knowledge and Information Systems, volume 30, pp. 693–713. Springer
7. Bache R, Azzopardi L (2010) Improving access to large patent corpora. In Transactions on Large-Scale Data- and Knowledge-Centered Systems II, volume 2, pages 103–121. Springer
8. Bashir S, Rauber A (2009a) Analyzing document retrievability in patent retrieval settings. DEXA '09: Proceedings of the 20th International Conference on Database and Expert Systems Applications (Springer). Linz, Austria, pp 753–760
9. Bashir S, Rauber A (2009b) Improving retrievability of patents with cluster-based pseudo-relevance feedback documents selection. In CIKM '09: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pages 1863–1866, Hong Kong, China, November 2–6
10. Bashir S, Rauber A (2010a) Improving retrievability and recall by automatic corpus partitioning. In: Transactions on large-scale data- and knowledge-centered systems II, vol 2. Springer, pp 122–140
11. Bashir S, Rauber A (2010b) Improving retrievability of patents in prior-art search. In: ECIR '10: 32nd European conference on information retrieval research (Springer). Milton Keynes, UK. Springer, pp 457–470, March 28–31
12. Bashir S, Rauber A (2011) On the relationship between query characteristics and ir functions retrieval bias. J Am Soc Inf Sci Technol 62(8):1512–1532
13. Callan J, Connell M (2001) Query-based sampling of text databases. ACM Trans Inf Syst (TOIS) J 19(2):97–130
14. Cao G, Nie J-Y, Gao J, Robertson S (2008) Selecting good expansion terms for pseudo-relevance feedback. In: SIGIR '08: proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval. ACM, New York, NY, USA, pp 243–250
15. Chen H (1995) Machine learning for information retrieval: neural networks, symbolic learning, and genetic algorithms. J Am Soc Inf Sci Technol 46(3):194–216
16. Cronen-Townsend S, Zhou Y, Croft WB (2002) Predicting query performance. In: SIGIR '02: proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval, August 11–15. Tampere, Finland, pp 299–306
17. Cummins R, O'Riordan C (2005) Evolving general term-weighting schemes for information retrieval: tests on larger collections. Artif Intell Rev 24(3–4):277–299
18. Cummins R, O'Riordan C (2009) Learning in a pairwise term-term proximity framework for information retrieval. In: SIGIR '09: proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval. ACM, New York, NY, USA, pp 251–258
19. Diaz-Aviles E, Nejdl W, Lars S-T (2009) Swarming to rank for information retrieval. In: GECCO '09, proceedings of the 11th annual conference on genetic and evolutionary computation. ACM, New York, NY, USA, pp 9–16
20. Fan W, Fox EA, Pathak P, Wu H (2004) The effects of fitness functions on genetic programming-based ranking discovery for web search: research articles. J Am Soc Inf Sci Technol 55(7):628–636
21. Fujii A, Iwayama M, Kando N (2007) Introduction to the special issue on patent processing. Inf Process Manag J 43(5):1149–1153
22. Gastwirth JL (1972) The estimation of the Lorenz curve and Gini index. Rev Econ Stat 54(3):306–416
23. Hauff C, Hiemstra D, de Jong F, Azzopardi L (2009) Relying on topic subsets for system ranking estimation. In: CIKM '09: proceeding of the 18th ACM conference on information and knowledge management, pp 1859–1862
24. He B, Ounis I (2006) Query performance prediction. Inf Syst J 31(7):585–594
25. Itoh H (2004) Patent retrieval experiments at ricoh. In: Proceedings of NTCIR '04: NTCIR-4 workshop meeting
26. Kamps J (2005) Web-centric language models. In: CIKM'05: proceeding of the 14th ACM conference on information and knowledge management. ACM
27. Koza JR (1992) A genetic approach to the truck backer upper problem and the inter-twined spiral problem. In: Proceedings of IJCNN international joint conference on neural networks, vol IV. IEEE Press, pp 310–318
28. Kraaij W, Westerveld T (2000) Tno/ut *at trec-9: How different are web documents? In Proceedings of TREC-9, the 9th text retrieval conference
29. Lawrence S, Giles CL (1999) Accessibility of information on the web. Nature 400:107–109

30. Losada DE, Azzopardi L (2008) An analysis on document length retrieval trends in language modeling smoothing. Inf Retr J 11(2):109–138
31. Lupu M, Huang J, Zhu J, Tait J (2009) TREC-CHEM: large scale chemical information retrieval evaluation at TREC. ACM SIGIR Forum 43(2):63–70
32. Mase H, Matsubayashi T, Ogawa Y, Iwayama M, Oshio T (2005) Proposal of two-stage patent retrieval method considering the claim structure. ACM Trans Asian Lang Inf Process (TALIP) 4(2):190–206
33. Mowshowitz A, Kawaguchi A (2002) Bias on the web. Commun ACM 45(9):56–60
34. Nuray R, Can F (2006) Automatic ranking of information retrieval systems using data fusion. Inf Process Manag J 42(3):595–614
35. Cordon O, Herrera-Viedma E (2003) A review on the application of evolutionary computation to information retrieval. Int J Approx Reason 34(2–3):241–264
36. Robertson SE, Walker S (1994) Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In: SIGIR '94: proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval. Dublin, Ireland, pp 232–241
37. Shinmori A, Okumura M, Marukawa Y, Iwayama M (2003) Patent claim processing for readability: structure analysis and term explanation. In: Proceedings of the ACL-2003 workshop on patent corpus processing, vol 20, pp 56–65
38. Singhal A (1997) At&t at trec-6. In: The 6th text retrieval conference (TREC6), pp 227–232
39. Singhal A, Buckley C, Mitra M (1996) Pivoted document length normalization. In: SIGIR '96: proceedings of the 19th annual international ACM SIGIR conference on research and development in, information retrieval. ACM, pp 21–29
40. Soboroff I, Nicholas C, Cahan P (2001) Ranking retrieval systems without relevance judgments. In: SIGIR '01: proceedings of the 24th annual international ACM SIGIR conference on research and development in, information retrieval, pp 66–73
41. Spoerri A (2007) Using the structure of overlap between search results to rank retrieval systems without relevance judgments. Inf Process Manag J 43(4):1059–1070
42. Tao T, Zhai C (2007) An exploration of proximity measures in information retrieval. In: SIGIR '07: proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval. ACM, New York, NY, USA, pp 295–302
43. Vaughan L, Thelwall M (2004) Search engine coverage bias: evidence and possible causes. Inf Process Manag J 40(4):693–707
44. Verberne S, van Halteren H, Theijssen D, Raaijmakers S, Boves L (2011) Learning to rank for why-question answering. Inf Retr 14:107–132
45. Vrajitoru D (1998) Crossover improvement for the genetic algorithm in information retrieval. Inf Process Manag J 34(4):405–415
46. Lauw WH, Lim E-P, Wang K (2006) Bias and controversy: beyond the statistical deviation. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining. Philadelphia, PA, USA, pp 625–630
47. Wu S, Crestani F (2003) Methods for ranking information retrieval systems without relevance judgments. In: SAC '03: proceedings of the 2003 ACM symposium on applied, computing, pp 811–816
48. Zhai C (2002) Risk minimization and language modeling in text retrieval. PhD Thesis, Carnegie Mellon University
49. Zhao J, Yun Y (2009) A proximity language model for information retrieval. In: SIGIR '09: proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval. ACM, New York, NY, USA, pp 291–298
50. Zhao Y, Scholer F, Tsegay Y (2008) Effective pre-retrieval query performance prediction using similarity and variability evidence. In: ECIR'08: proceedings of the 30th European conference on advances in information retrieval. Glasgow, UK, pp 52–64

Shariq Bashir is currently working as an Assistant Professor at National University of Computer and Emerging Sciences (NU-FAST), Islamabad. He received his PhD degree in Computer Sciences from Vienna University of Technology in 2011 and BS and MS degrees in Computer Sciences from National University of Computer and Emerging Sciences (NU-FAST), Islamabad, and Punjab University, Lahore, in 2003 and 2005, respectively. From 2006 to 2007, he worked as a Lecturer in the Faculty of Computer Sciences at National University of Computer and Emerging Sciences (NU-FAST), Islamabad. Dr. Bashir has published around 25 research papers in leading international conferences and journals of information retrieval and data mining. His current research interests are analyzing retrieval systems performance using retrievability measurements particularly for patent retrieval domain, retrieval bias analysis, automatic analysis of retrieval systems and query performance prediction.

Andreas Rauber is Associate Professor at the Department of Software Technology and Interactive Systems (ifs) at the Vienna University of Technology (TU-Wien). He furthermore is president of AARIT, the Austrian Association for Research in IT and a Honorary Research Fellow in the Department of Humanities Advanced Technology and Information Institute (HATII), University of Glasgow. Dr. Rauber received his MSc and PhD in Computer Science from the Vienna University of Technology in 1997 and 2000, respectively. He has published numerous papers in refereed journals and international conferences and served as PC member and reviewer for several major journals, conferences and workshops. Dr. Rauber research interests cover the broad scope of digital libraries and information spaces, including specifically text and music information retrieval and organization, information visualization, as well as data analysis, neural computation and digital preservation.