An Information-Theoretic Account of Static Index Pruning

Ruey-Cheng Chen
National Taiwan University
1 Roosevelt Rd. Sec. 4
Taipei 106, Taiwan
rueycheng@turing.csie.ntu.edu.tw

Chia-Jung Lee
University of Massachusetts
140 Governors Drive
Amherst, MA 01003-9264
cjlee@cs.umass.edu

ABSTRACT

In this paper, we recast static index pruning as a model induction problem under the framework of Kullback's principle of minimum cross-entropy. We show that static index pruning has an approximate analytical solution in the form of convex integer program. Further analysis on computation feasibility suggests that one of its surrogate model can be solved efficiently. This result has led to the rediscovery of *uniform pruning*, a simple yet powerful pruning method proposed in 2001 and later easily ignored by many of us. To empirically verify this result, we conducted experiments under a new design in which prune ratio is strictly controlled. Our result on standard ad-hoc retrieval benchmarks has confirmed that uniform pruning is robust to high prune ratio and its performance is currently state of the art.

Categories and Subject Descriptors

H.1.1 [**Systems and Information Theory**]: Information theory; H.3.1 [**Content Analysis and Indexing**]: Indexing methods; H.3.4 [**Systems and Software**]: Performance evaluation (efficiency and effectiveness)

Keywords

Static index pruning; principle of minimum cross-entropy; model induction; uniform pruning

1. INTRODUCTION

Kullback discussed one famous problem in his seminal work [14] about inducing a probability measure based on some previous measurement. When one has some initial hypothesis about a system and seeks to update this measurement incrementally, she needs to choose a new hypothesis from a set of feasible measures that best approximates her current belief. Here, the difficulty lies in defining the notion of closeness in the probability space. While at the time this was an important issue in everyday probabilistic modeling, a genuine solution had yet to come.

SIGIR'13, July 28–August 1, 2013, Dublin, Ireland.

To answer the question he had raised, Kullback introduced a method called "minimum discrimination information," or in the recent literature known as the *principle of minimum cross-entropy*. This approach has later become one of the most influential inductive principles in statistics, and also has benefited numerous fields, including some subareas in information retrieval [15,20]. Kullback's solution was simple and elegant: One shall choose a measure that most closely resembles the previous measurement in terms of Kullback-Leibler divergence. Specifically, this is equivalent to solving the following optimization problem, given some prior measure p and a set of feasible measures $\mathcal{F}$:

$$\begin{array}{ll} \text{minimize} & \mathrm{D}(q||p) \\ \text{subject to} & q \in \mathcal{F}. \end{array} \tag{1}$$

In this paper, we apply this induction framework to a classic problem in information retrieval, called *static index pruning*. Static index pruning is a task that reduces the index size for improving disk usage and query throughput [1]. Size reduction is done by removing index entries. Generally, the aim in static index pruning is to find a subset of index entries that best approximates the full index in terms of retrieval performance. This aspect, as we will show later, is closely related to model induction.

One key assumption in this paper is that an inverted index is a nonparametric, conditional distribution of document D given term T, i.e., $p(D|T)$. This follows directly from Chen et al.'s definition [10], which allows us to measure the resemblance between two versions of inverted indexes the way we do probability distributions. Here, the following definitions put static index pruning into the framework of Equation (1):

- The prior distribution p is defined as the full (unpruned) inverted index.
- The set of feasible hypotheses $\mathcal{F}$ contains all the possible pruned indexes of p that have reached some given prune ratio ρ. In other words, each element $q \in \mathcal{F}$ is a pruned version of the original inverted index p.

This conception marks the very beginning of our quest for developing an efficient solution of static index pruning. Through analysis, we first show that static index pruning is essentially a combinatorial optimization problem. Nevertheless, in Section 3, we manage to obtain a weaker analytical solution that is practically operable by trading off some mathematical rigor. We found that, under appropriate assumptions, static index pruning reduces to a convex integer program. But this is not a good solution in general, since the number of variables in the convex program is linear to the number of postings in the inverted index, which may easily exceed a few millions on any medium-sized text collection. That means this solution does not scale at all, even with the latest super-efficient convex solver.

We further attacked this problem using an alternative approach, called surrogate modeling. We created a surrogate problem that is easier to solve. As we will show in later sections, this analytical solution has pointed us to a general version of a simple pruning method called *uniform pruning*. Sharp-eyed readers might notice that uniform pruning is by no means a new invention. Uniform pruning was originally introduced to static index pruning in Carmel et al.'s paper as a baseline approach [9]. In a preliminary experiment, Carmel et al. compared this method with their term-based pruning method. Using TF-IDF as the score function, they found that, even though term-based method performed slightly better, in general the performance for both approaches was roughly comparable. While this was indeed a very interesting finding, the exploration was discontinued as they went ahead to study other important issues.

To the best of our knowledge, since then uniform pruning has not been studied in any follow-up work. It is easy to see why this has been the case. The lack of control on one experiment variable, prune ratio, has made the performance result difficult to interpret. When we make comparisons between methods, this variable needs to be strictly controlled so that the comparisons make sense. Nevertheless, very few in the previous work adopted this design. As a result, there was no obvious way to conduct any form of significance testing to static index pruning. Without serious scrutiny—by which we mean significance assessment—it is only reasonable to dismiss uniform pruning, for that it seemed like an ad-hoc and maybe inferior approach.

In our study, the rediscovery of uniform pruning has gained us a second chance to rethink this issue. Our answer was a redesigned empirical study, in which prune ratio for each experimental method is strictly controlled to minimize the experimental error, and the performance is analyzed using multi-way repeated-measure analysis of variance. As we will shortly cover, the experiment result suggests that uniform pruning with Dirichlet smoothing significantly outperformed the other term-based methods under diverse settings.

The rest of the paper is structured as follows. Section 2 covers an overview to static index pruning and the relevant research. In Section 3, we motivate static index pruning in the minimum cross-entropy framework and show that the analytical solution leads to the uniform pruning method. An empirical study is given in Section 4. We put the theoretical and empirical evidence together and discuss the implication in Section 5. Section 6 delivers the concluding remarks.

2. RELATED WORK

In the coming subsections, we briefly review the literature and discuss the recent development of static index pruning. Following an overview, some notable pruning methods will be treated in slightly more details. Note that this is only aimed at providing enough background knowledge for the reader. A complete coverage is not attempted here.

2.1 Overview

The idea of static index pruning first appeared in the groundbreaking work of Carmel et al. [9] and has since garnered much attention for its implication to Web-scale retrieval [8,11]. Static index pruning is all about reducing index size—by removing index entries from the inverted index. This technique was proposed to mitigate the efficiency issue caused by operating a large index, for that a smaller index loads faster, occupies less disk space, and has better query throughput. But since only partial term-document mapping is preserved, a loss in retrieval performance is inevitable.

Much effort has been driven towards developing importance measures of individual index entries, so that one can easily prioritize index entries on their way out of the index. Many such measures have been proposed and tested in various retrieval settings. One simple example is *impact*, the contribution of a term-document pair to the final retrieval score [8,9]. Other approaches in this line include probability ranking principle (PRP) [7], two-sample two proportion (2P2N) [19], and information preservation (IP) [10]. Some measures assess only term importance [6], so the corresponding pruning algorithms can only choose between keeping the entire term posting list or not at all. Some others assess only documents importance [21].

2.2 Methodologies

Term-based pruning (or *term-centric pruning*) is proposed by Carmel et al. [9]. It was so named because it attempts to reduce the posting list for each term in the index. The basic idea is to compute a cutting threshold for each term, and throw away those entries with smaller impact values. Since the cutting threshold depends on some order statistics (i.e., the k-th largest impact value) about the posting list, term-based pruning is less efficient than the other methods.

In contrast to the aforementioned term-centric approach, *document-centric pruning* seeks to reduce the posting list for each document. Büttcher and Clarke [8] considered the contribution for term t to the Kullback-Leibler divergence $\mathrm{D}(d||C)$ between document d and the collection model C. This quantity is used to measure the importance of a posting. Analogously, for each document, a cutting threshold has to be determined based on some order statistics.

There are also other pruning strategies that focus on removing an entire term posting list (whole-term) or an entire document (whole-document) all at once. Blanco and Barreiro [6] presented four term-importance measures, including inverse document frequency (idf), residual inverse document frequency (ridf), and two others based on term discriminative value (TDM). They adopted a whole-term pruning strategy. Analogously, Zheng and Cox [21] proposed an entropy-based measure in a whole-document pruning setting. Both parties have reported comparable performance to term-based pruning on some standard benchmark.

Blanco and Barreiro [7] developed a decision criterion based on the probability ranking principle [18]. The idea is to take every term in the index as a single-word query and calculate the odd-ratio of relevance $p(r|t,d)/p(\overline{r}|t,d)$. This quantity is used in prioritizing all the term-document pair. Since there is only one cutting threshold determined globally, the implementation is relatively easy and efficient.

Thota and Carterette [19] used a statistical procedure, called *two-sample two-proportion* (2P2N), to determine if the occurrence of term t within document d is significantly different from its occurrence within the whole collection. Chen et al. [10] developed a method called *information preservation*. They suggest using the innermost summand of the conditional entropy $H(D|T)$ to measure predictive power contributed by individual term-document pairs to the index model. This quantity is claimed easier to compute than the probability ranking principle.

Altingovde et al. [2] proposed an interesting *query-view* technique that works orthogonally with the aforementioned measure-based approaches. The general idea is to count the number of time a document falls within the top-k window of any given query collected from the query log. The count collected from a query is then evenly distributed to individual query terms. Thus the larger this number, the greater importance the posting is. The query view algorithm would later use this information to prune the entries.

Our work in this paper departs from the previous effort in three major ways. First, our approach is model-based, meaning that we infer a pruned model as a whole rather than partially. This is a novel approach in contrast to all the previous methods. Second, other information-theoretic approaches, such as Zheng and Cox [21] and Chen et al. [10], focused on minimizing the loss of information, while ours focused on minimizing the divergence from the full index. These are entirely different concepts in information theory. Three, our result on the uniform pruning method is more general than Carmel et al.'s description because we considered the query model $p(t)$. Our take of uniform pruning is a weighted version, which may be useful when such a query model (e.g., session logs) is available.

3. MINIMUM CROSS-ENTROPY AND STATIC INDEX PRUNING

3.1 Problem Definition

Let us first develop some notation for describing an inverted index. Let T denote the set of terms and D denote the set of documents. We define an index entry (*posting*) as a 3-tuple of the form (t, d, n), where $t \in T$, $d \in D$, and $n \in \mathbf{N}_+$ (i.e., n is a positive integer.) This means that "term t appears n times in document d." We further consider an inverted index as a probabilistic model $p(D|T; \theta)$ that takes a set of index entries θ as parameters. This model is therefore *nonparametric* because the number of its parameters is not fixed. For brevity, in this paper we sometimes abuse the notation and use one symbol, e.g., θ, to represent both a distribution and its parameters.

In static index pruning, one seeks to induce a pruned model θ from a full model θ_0 such that the following two constraints are satisfied: (i) θ is a subset of the full model θ_0, and (ii) the size of θ is $1 - \rho$ times the size of θ_0. Here, $0 < \rho < 1$ denotes the *prune ratio*. Note that these constraints only specify what we need as the output from static index pruning, not how pruning shall be done. As there are exponentially many ways to prune an index down to a given ratio, it is natural to ask how does one engineer this decision to avoid excessive performance loss.

Now, to illustrate this point, let us assume the existence of a function $g(\theta)$ that measures the retrieval performance of model θ. With this hypothetical construct, we formally define static index pruning as follows.

$$\begin{array}{ll} \text{maximize} & g(\theta) \\ \text{subject to} & \theta \subseteq \theta_0 \\ & |\theta|/|\theta_0| \text{ reaches } 1-\rho. \end{array} \tag{2}$$

It is not difficult to envision static index pruning being formulated this way, as a constrained optimization problem. For now, we shall focus on estimation of this hypothetical function. The conventional approach, as discussed in Section 2.2, is to devise an importance measure to take the role of $g(\cdot)$, which is expected to capture certain properties of an index relevant to retrieval performance. Yet one caveat is that sometimes we risk being arbitrary: The importance measure may only be empirically tested and does not necessarily come with any theoretical guarantee.

One simple idea that we had failed to see casted away all these doubts. We noticed the similarity between this formulation and Kullback's famous induction framework. As we replace $g(\cdot)$ in Equation (2) with the negative Kullback-Leibler divergence (KL divergence), the static index pruning problem reduces to a model induction problem, written in a minimization form:

$$\begin{array}{ll} \text{minimize} & \mathrm{D}(\theta||\theta_0) \\ \text{subject to} & \theta \subseteq \theta_0 \\ & |\theta|/|\theta_0| \text{ reaches } 1-\rho. \end{array} \tag{3}$$

In the following subsections, we shall develop a procedure to practically solve this optimization problem. For brevity, we write $p(\cdot)$ and $p_0(\cdot)$, respectively, to denote the models parametrized by inverted indexes θ and θ_0. The probability measures that we consider here are conditional distributions of D given T. To make this explicit, we define:

$$\mathrm{D}(\theta||\theta_0) \equiv \mathrm{D}(p(D|T)||p_0(D|T)). \tag{4}$$

3.2 Assumptions

Before diving into the full analysis, we need to make explicit two important assumptions.

Assumption 1 (Query and Index Models). *We can separate a joint distribution of D and T into a product of two models: (1) a distribution of T, called the query model, and (2) a conditional distribution of D given T, called the index model. We assume there is only one query model $q(t)$ and it is independent of the index model in use. In other words, we have:*

$$p(d,t) = p(d|t)q(t), \qquad p_0(d,t) = p_0(d|t)q(t).$$

Sometimes, we simply write $p(t)$ or $p_0(t)$ to denote the query model when the meaning is clear in the context.

Assumption 1 simply states that the query model $q(t)$ (or $p(t)$) has to be estimated from somewhere else. It makes little sense to infer a query model from the index.

Assumption 2 (Normalization Factor). *Let $p(t|d)$ and $p_0(t|d)$ be the conditional distributions of T given D for the induced and the original models, respectively. Let $\mathbb{I}_{t,d}$ be a binary variable that indicates whether an index entry (t, d, n) (for some $n \in \mathbb{N}_+$) in the original model is retained in the induced model. We have $p(t|d) \equiv \mathbb{I}_{t,d}p_0(t|d)/Z_d$, where Z_d is the normalization factor for document d.*

In Assumption 2, we introduce a normalization factor Z_d for each document d. As we shall address later, setting an appropriate value for Z_d is the key step in the subsequent analysis. To correctly normalize $p(t|d)$, we need to set:

$$Z_d = \sum_t \mathbb{I}_{t,d} p_0(t|d).$$

But this would make the resulting formula intractable, since the value of $\mathbb{I}_{t,d}$ depends on other variables in the same document, i.e., $\mathbb{I}_{\cdot,d}$. To deal with this issue, we suggest setting $Z_d = k$ for all $d \in D$, where $k > 0$ is some constant. Using this normalization trick results in weak inference and inevitably sacrifices mathematical rigors. We want to emphasize that this is a necessary compromise, without which the following analysis would not have been possible.

3.3 Analysis

Now, we shall go ahead and analyze the objective function. First of all, let us write out the objective in full:

$$\mathrm{D}(p(D|T)||p_0(D|T)) \equiv \sum_{t,d} p(d,t) \log \frac{p(d|t)}{p_0(d|t)}. \tag{5}$$

We use Assumption 1 to dissect the joint distribution $p(d,t)$ into the product of the query model $p(t)$ and the index model $p(d|t)$. Applying Bayes Theorem to $p(d|t)$ and $p_0(d|t)$ and assuming uniform $p(d)$ and $p_0(d)$, we have the objective organized as follows:

$$\sum_t p(t) \sum_d \frac{p(t|d)}{\sum_{d'} p(t|d')} \log \frac{p(t|d)}{p_0(t|d)} \frac{\sum_{d'} p_0(t|d')}{\sum_{d'} p(t|d')}. \tag{6}$$

Observe that, since in this optimization framework we look for a subset of θ_0, we are essentially dealing with a combinatorial problem ("assignment problem"). Each index entry $(t, d, n) \in \theta_0$ either stays within the induced model θ or gets removed. This combinatorial nature is best characterized via the indicator variables $\mathbb{I}_{\cdot,\cdot}$ in Assumption 2.

Let us now replace all the occurrences of $p(t|d)$. Note that, under the setting $Z_d = k$ (suggested), all the normalization factors cancel out. We have:

$$\sum_t p(t) \sum_d \frac{\mathbb{I}_{t,d} p_0(t|d)}{\sum_{d'} \mathbb{I}_{t,d'} p_0(t|d')} \log \mathbb{I}_{t,d} \frac{\sum_{d'} p_0(t|d')}{\sum_{d'} \mathbb{I}_{t,d'} p_0(t|d')}. \tag{7}$$

As we separate the support of the inner summation over d into two subsets according to whether $\mathbb{I}_{t,d}$ is switched on, i.e., one over $\{d|\mathbb{I}_{t,d} = 1\}$ and the other over $\{d|\mathbb{I}_{t,d} = 0\}$, the latter sub-summation disappears since $0 \log 0 = 0$. The resulting equation becomes:

$$\sum_t p(t) \sum_{d:\mathbb{I}_{t,d}=1} \frac{p_0(t|d)}{\sum_{d'} \mathbb{I}_{t,d'} p_0(t|d')} \log \frac{\sum_{d'} p_0(t|d')}{\sum_{d'} \mathbb{I}_{t,d'} p_0(t|d')}. \tag{8}$$

Notice that the innermost logarithm does not depend on d anymore. We can therefore move that entire term out of the inner summation. From there, we have the inner summation over d canceled out. The equation is now written as:

$$\sum_t p(t) \log \frac{\sum_{d'} p_0(t|d')}{\sum_{d'} \mathbb{I}_{t,d'} p_0(t|d')}. \tag{9}$$

We can get rid of the numerator, i.e., $\sum_{d'} p_0(t|d')$, in the logarithm when minimizing this equation, because the numerator does not depend any combinatorial choice we make. Once again, we rewrite it as a maximization problem by taking the negation. The final form of static index pruning is expressed as the following:

$$\begin{array}{ll} \text{maximize} & \sum_t p_0(t) \log \sum_d \mathbb{I}_{t,d} p_0(t|d) \\ \text{subject to} & \mathbb{I}_{t,d} \text{ is binary, for all } (t,d,\cdot) \in \theta_0, \\ & \sum_{t,d} \mathbb{I}_{t,d} = (1-\rho)|\theta_0|. \end{array} \tag{10}$$

Input: a global threshold ϵ
begin
  for $t \in T$ **do**
    for $d \in postings(t)$ **do**
      compute $A(t,d) = p(t)p(t|d)$
      if $A(t,d) < \epsilon$ **then**
        remove d from $postings(t)$
      end
    end
  end
end

Algorithm 1: The weighted uniform pruning algorithm.

Equation (10) is in general ill-posed even though it can be solved with a convex integer program solver. This is because the number of index entries can easily exceed a few millions in any production retrieval system. Solving this exactly is only possible for very small test collections. To tackle this issue, we resort to a technique called *surrogate modeling* (or optimization transfer), which approximates the original objective using a majorization/minorization function that is analytically or numerically efficient to compute. See Lange et al. [16] for a comprehensive treatment.

To the best of our knowledge, there are two major approaches for inducing such surrogate models: taking the first-order Taylor approximation, or using the Jensen's inequality. In this paper, we stick with the second approach[1]. Recall that Jensen's inequality states that the following properties hold for any convex (or concave) function f:

$$\mathbb{E}f(X) \geq f(\mathbb{E}X) \quad (f \text{ is convex})$$
$$\mathbb{E}f(X) \leq f(\mathbb{E}X) \quad (f \text{ is concave}).$$

Let f be the logarithmic function. The original objective in our problem (Equation 10) now corresponds to the left-hand side $\mathbb{E}f(X)$. Since the logarithmic function is concave, we have the surrogate model $f(\mathbb{E}X)$ an upper bound of the original objective:

$$\text{maximize} \quad \log \sum_{t,d} \mathbb{I}_{t,d} p_0(t) p_0(t|d), \tag{11}$$

or equivalently:

$$\text{maximize} \quad \sum_{t,d} \mathbb{I}_{t,d} p_0(t) p_0(t|d). \tag{12}$$

This surrogate model has a simple analytical solution: Sort the index entries according to weighted query likelihood, i.e., $p(t)p(t|d)$, and keep only the top $(1-\rho)N$ entries. It can be shown that a simple maneuver such as Algorithm 1, called *weighted uniform pruning*, guarantees to maximize the objective. This corresponds to a weighted version of Carmel et al.'s uniform pruning method. This algorithm would fall back to the unweighted form when we supply a uniform $p(t)$ and a plug-in estimate of the query likelihood. Note that the plug-in approach is valid only when the score function is proportional to the true likelihood.

For simplicity, we take a very loose definition of query likelihood in this paper so as to cover the well-known BM25 function. As we shall present shortly, the empirical result

[1] In our case, the first-order Taylor expansion leads to an even more sophisticated objective.

shows that the performance of BM25 is no worse than that of a rigorously defined language model (with Jelinek-Mercer smoothing). What is left unsettled is how to estimate ϵ given a target prune ratio ρ. This issue is treated in Section 4.2.

4. EXPERIMENT

Thus far, we have established the theoretical ground for uniform pruning. Our next quest is to find empirical evidence that supports this result. In the coming subsections, we shall briefly describe the experiment settings and present the experimental result in greater detail.

4.1 Setup

We used three test collections in this experiment: TREC disks 4 & 5, WT2G, and WT10G. The first two collections are tested against topics 401-450 and the latter against topics 451-550. For each topic, we tested both short (title) and long (title + description) queries. Details about the benchmark are summarized in Table 1. All three collections were indexed using the Indri toolkit[2]. To preprocess the documents, we applied the porter stemmer and used the standard 411 InQuery stoplist. No additional text processing is done to the test collections.

According to how index traversal is preferred, a pruning method can be either term-centric or document-centric. Since different traversal strategies rely on different index creation procedures, it is difficult to have both sets implemented in one place. For simplicity, in this experiment we focused only on term-centric methods. Specifically, we tested the following methods:

1. Uniform pruning (UP) [9]: This method is the subject of this experiment. In this experiment, we tested three variations of uniform pruning, each using a different score function. These functions are BM25 (UP-bm25), language model using Dirichlet smoothing (UP-dir), and language model using Jelinek-Mercer smoothing (UP-jm). For BM25, we used the standard setting provided by Indri. For language models, we set $\mu = 2500$ in Dirichlet smoothing and $\lambda = 0.6$ for the Jelinek-Mercer smoother.

2. Top-k term-centric pruning (TCP) [9]: We set $k = 10$ as suggested to maximize the top-10 precision and used BM25 as the score function. Note that other score functions such as language models may also apply to this pruning method. Here, we simply comply with the previous work.

3. Probability ranking principle (PRP) [7]:
$$\frac{p(r|t,d)}{p(\overline{r}|t,d)} \equiv \frac{p(t|D)p(r|D)}{p(t|\overline{r})(1-p(r|D))}.$$
As suggested, we use the following equations to estimate these probabilities:
$$p(t|D) = (1-\lambda)p_{\mathrm{ML}}(t|D) + \lambda p(t|C), \qquad (13)$$
$$p(r|D) = \frac{1}{2} + \frac{1}{10}\tanh\frac{dl - \overline{X_d}}{S_d}, \qquad (14)$$
$$p(t|\overline{r}) = p(t|C). \qquad (15)$$
Note that dl is the document length. X_d and S_d respectively are the sample mean and sample standard deviation of document length. For query likelihood, we set $\lambda = 0.6$.

4. Information preservation, with uniform document prior (IP-u) [10]:
$$-\frac{p(t|d)p(d)}{\sum_{d'} p(t|d')p(d')} \log \frac{p(t|d)p(d)}{\sum_{d'} p(t|d')p(d')}.$$
In this formula, the query likelihood $p(t|d)$ is estimated using Jelinek-Mercer smoothing. Here, We set $\lambda = 0.6$ and assumed uniform document prior $p(d)$.

[2]http://www.lemurproject.org/indri.php

Collection	# Documents	Query Topics
Disks 4 & 5	528k	401-450
WT2G	247k	401-450
WT10G	1692k	451-550

Table 1: Test collections and the corresponding query topics.

We are aware that document-length update may improve the TCP and PRP retrieval performance [5,7]. Nevertheless, in this study we did not implement this feature. This shall be addressed in the future work.

4.2 Prune Ratio

In this experiment, we settled on 9 fixed prune levels at $\rho = 0.1, 0.2, \ldots, 0.9$. To control the prune ratio, comparison is only allowed between experimental runs at the same prune level. In each reference method, the true prune ratio depends on some parameter (e.g., ϵ in TCP and PRP), which we called the *threshold parameter*. To reduce the index down to the right size, we employed two different approaches to determine this cutting threshold:

1. Sample percentile: Collect the prune scores on top of a sample of index entries and use the percentile estimates to determine the right cutting threshold. This is mostly useful when the prune score is globally determined. Here, we used Definition 8 from Hyndman and Fan [13] to estimate percentiles.

2. Bisection: Take an interval of feasible parameter values $[a, b]$, and test-prune using the median value $(a + b)/2$. Return the current median if the test-prune reached the expected ratio; otherwise shrink the interval in half and repeat. This method is useful when the prune score for each index entry depends on the others in the same posting list, as in TCP.

Bisection requires several test-prune runs into the entire index and is therefore more time-consuming. Sample percentile needs only one pass through the index, but the resulting prune ratio can be less precise than that with the values learned using bisection. In this paper, we applied bisection to TCP to learn the parameter ϵ, and applied sample percentile to the rest of methods. Specifically, we used a sample size of 10% of the entire index. For either case, the prune ratio error is controlled to within ±0.2%.

MAP (t)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	160	157	152	145	139	134	127	118	095
UP-bm25	159	158	154	148	144	140	135	127	102
UP-dir	160	157	153	149	145	143	142	137	120
UP-jm	160	158	153	146	140	133	126	110	085
PRP	156	152	148	142	133	123	110	088	045
IP-u	156	153	148	140	135	123	107	092	046

MAP (td)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	203	197	187	177	163	147	142	130	091
UP-bm25	204	189	177	170	160	158	141	141	116
UP-dir	204	199	191	183	177	175	174	164	136
UP-jm	204	199	185	179	164	150	137	103	074
PRP	186	174	160	150	136	123	112	090	050
IP-u	189	171	165	149	143	124	116	095	051

P10 (t)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	260	256	252	246	245	236	209	201	174
UP-bm25	259	258	254	239	226	225	204	192	168
UP-dir	261	257	254	248	249	242	241	236	222
UP-jm	261	259	253	249	243	237	225	190	168
PRP	256	250	239	224	201	182	157	125	107
IP-u	257	248	243	214	202	193	164	133	106

P10 (td)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	347	347	339	332	315	290	279	267	231
UP-bm25	351	339	336	309	295	287	264	248	209
UP-dir	347	350	341	338	327	317	316	311	291
UP-jm	347	350	341	335	319	299	267	229	191
PRP	344	324	300	281	249	222	181	169	127
IP-u	340	329	310	280	253	233	186	175	122

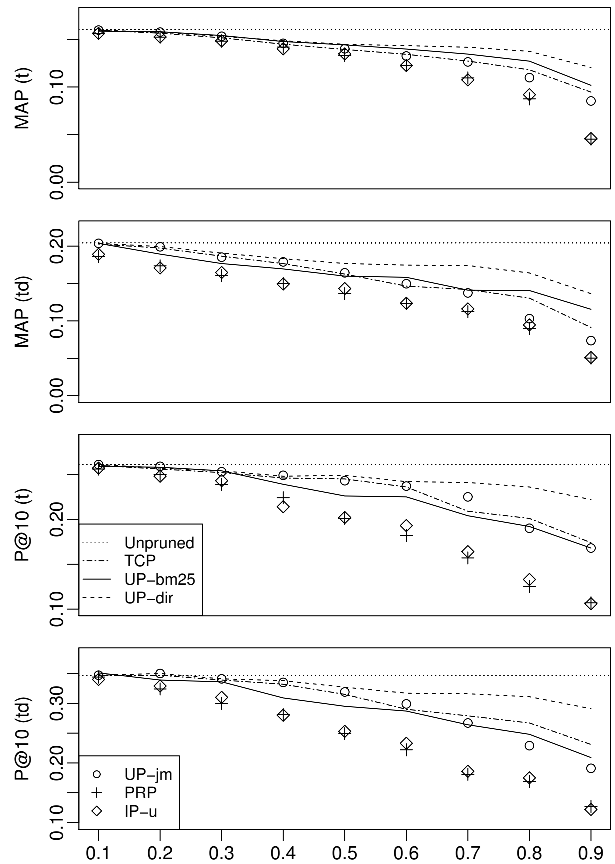

Figure 1: The overall performance result on WT10g. All the measurements are rounded to the 3rd decimal place, with preceding zero and decimal point removed. The best raw performance at each prune level is underlined. The unpruned baseline has achieved 0.160/0.204 (t/td) in MAP and 0.261/0.347 (t/td) in P@10.

4.3 Retrieval Performance

We followed Blanco and Barreiro [7] for using BM25 as the post-pruning retrieval method. Retrieval performance is measured in mean average precision (MAP) and precision-at-10 (P@10). The result on the largest set WT10G is summarized in Figure 1 (see Figures 2 and 3 at the very end of this paper for results on the smaller sets). Each figure has four sets of measure-to-query-type combinations, and the result for each combination is given both as a table on the left and a plot on the right. These combinations from top to bottom, respectively, are MAP-t, MAP-td, P@10-t, and P@10-td. Table columns and x-axes in the plots indicate prune levels, from 0.1 to 0.9 (10% to 90%). Rows and curves indicate pruning methods.

Our result shows that, at small prune levels (≤ 0.5), all these methods differ little in performance, and the difference at larger prune levels seems more evident. Both PRP and IP-u, whose performance was nearly identical, have consistently achieved the bottom performance in all settings. In general, the performance for the UP family and TCP is comparable, though UP-dir performed slightly better than the other. We noticed that the performance of UP-dir is also robust to high prune ratio. On WT10G, when tested under an extreme setting with 90% prune ratio, UP-dir still managed to retain 75% of the baseline MAP for short queries, and 66.7% for long queries. Of the baseline P@10, UP-dir retained 85.1% for short queries and 83.9% for long queries. Under a less aggressive setting such as 50% prune ratio, UP-dir have done even better by retaining 90.6% and 86.8% of baseline MAP, and 95.4% and 94.2% of baseline P@10, respectively for short and long queries.

4.4 Significance Tests

We further conducted an analysis of variance (ANOVA) to check if the performance difference is significant. Due to the unbalanced size of measurement, we tested each corpus independently. Here, we assume a fixed-effect, 4-way no interaction, repeated measure design, expressed as:

$$Y_{i,j,k,l} = a_i + b_j + c_k + d_l + \epsilon_{i,j,k,l},$$

where $Y_{i,j,k,l}$ is the measured performance, a_i is the query-type effect, b_j the prune-level effect, c_k the method effect, and d_l the topic effect, and $\epsilon_{i,j,k,l}$ denotes the error.

The result is covered in Table 2. Each row indicates a measure-effect combination and each column a test corpus. Test statistics, such as degrees of freedom (DF) and F-values (F), are given for every test case. We used partial eta-square (η_p^2) to measure the effect size [17]. We first ran an omnibus test to see if any main effect is significant. Of all three collections, all the main effects were tested significant for

Response	Main Effect	Disks 4 & 5 DF	F	η_p^2	WT2G DF	F	η_p^2	WT10G DF	F	η_p^2
MAP	Query Type	F(1, 5336)	74.10	.01	F(1, 5336)	42.57	.01	F(1, 10686)	192.25	.02
	Prune Ratio	F(8, 5336)	240.30	.26	F(8, 5336)	306.17	.31	F(8, 10686)	193.26	.13
	Method	F(5, 5336)	11.00	.01	F(5, 5336)	40.20	.04	F(5, 10686)	61.47	.03
	Query Topic	F(49, 5336)	885.35	.89	F(49, 5336)	335.89	.76	F(49, 10686)	422.46	.80
P@10	Query Type	F(1, 5336)	66.16	.01	F(1, 5336)	10.89	.00	F(1, 10686)	622.34	.06
	Prune Ratio	F(8, 5336)	105.00	.14	F(8, 5336)	133.98	.17	F(8, 10686)	122.43	.08
	Method	F(5, 5336)	20.34	.02	F(5, 5336)	44.06	.04	F(5, 10686)	71.01	.03
	Query Topic	F(49, 5336)	484.06	.82	F(49, 5336)	296.88	.73	F(49, 10686)	226.31	.68

Table 2: The 4-way no-interaction ANOVA result. Each cell indicates a combination of performance measure (row) and test collection (column). Degrees of freedom and F-values are given for testing all the main effects. Effect size is given in η_p^2. In our experiment, all the main effects are significant for $p < 0.001$.

	Disks 4 & 5 Method	Mean	Group	WT2G Method	Mean	Group	WT10G Method	Mean	Group
MAP	UP-bm25	.204	a..	UP-dir	.223	a...	UP-dir	.162	a..
	UP-dir	.200	a..	UP-bm25	.211	.b..	UP-bm25	.151	.b.
	TCP	.196	ab.	TCP	.204	.b..	TCP	.148	.b.
	UP-jm	.191	.bc	UP-jm	.192	..c.	UP-jm	.145	.b.
	PRP	.187	..c	IP-u	.181	...d	IP-u	.129	..c
	IP-u	.187	..c	PRP	.179	...d	PRP	.127	..c
P@10	UP-dir	.433	a..	UP-dir	.404	a...	UP-dir	.286	a..
	TCP	.433	a..	TCP	.385	ab..	TCP	.268	.b.
	UP-jm	.424	a..	UP-jm	.367	.bc.	UP-jm	.265	.b.
	UP-bm25	.417	a..	UP-bm25	.359	..c.	UP-bm25	.259	.b.
	PRP	.392	.b.	IP-u	.322	...d	IP-u	.222	..c
	IP-u	.389	.b.	PRP	.319	...d	PRP	.219	..c

Table 3: The overall result for Tukey's HSD test. For each combination of performance measure (row) and test collection (column), pruning methods are ordered in descending mean and tested for group difference. Methods that differ significantly do not share the same group label.

$p < 0.001$. Further analysis shows that query type and prune method have relatively small effect sizes, suggesting that query topic and prune ratio have much greater influence on the retrieval performance than the others do.

Post-hoc tests are then called for to examine the difference caused by different factor values. Since our experimental setting involves multiple comparison, we employed Tukey's honest significance difference (HSD) to control the overall Type I error [12]. Note that since Tukey's HSD is a one-way test, only one effect is tested in each run. In the following paragraphs, we summarize the HSD results for all the main effects. Here, since our focus is on the method effect, we shall briefly cover the other three for the sake of completeness.

Method Effect. Table 3 summarizes this HSD result for the method effect. For each measure-corpus combination, we assigned group labels, e.g., "a" to "d", to individual methods based on the pairwise differences in their means. The difference between two methods is significant if and only if they share no common group label.

The result is briefly summarized as follows. First, the UP family and TCP consistently achieved top performance in both MAP and P@10 across different test settings. In the leading group, UP-dir delivers slightly better performance than the others. This is even more pronounced under the Web settings, in which UP-dir significantly outperformed the rest of methods in MAP (on both corpora) and in P@10 (on WT10G only). Second, the performance for the rest of UP family and TCP is in general comparable. Take UP-bm25 and TCP. The performance difference between the two is subtle: UP-bm25 was shown superior in MAP on Disks 4 & 5 but inferior in P@10 on WT2G. Third, PRP and IP-u are inferior to all the other methods. This result is consistent with our analysis on the raw performance measurements.

Query Type Effect. Long queries (td) achieve better performance than short queries (t), which is expected because short queries are less precise than longer ones. This difference is confirmed on all three test collections, and appears more evident in the largest set WT10G.

Prune Ratio Effect. Small prune levels do better than large ones in both metrics, which is also expected since more aggressive pruning results in less information in the index. According to the pairwise comparison made within the HSD test, this result is generally true except for a few small pairs such as 0.1-against-0.2. Specifically, WT10G has many such insignificant small pairs, suggesting that retrieval on larger Web collections is less sensitive to information loss.

Topic Effect. The result is difficult to interpret due to the size of topic pairs, e.g., topics 451-551 on WT10G has produced 4950 such pairwise comparisons. In general, only a small number of queries have significantly deviated from the average performance, meaning that most queries are designed to be about equally difficult.

5. DISCUSSION

The experiment result for uniform pruning is generally in line with our understanding to impact, much of this was contributed by the previous work in index compression and dynamic pruning. Since many ideas come from the same outlet in the indexing pruning community, it is no surprise that uniform pruning is related to many existing impact-based methods. For example, Anh et al. [3] concluded that impact-sorted indexes combined with early termination heuristics can best optimize retrieval system performance. This technique is conceptually equivalent to uniform pruning. Further work in this line investigated impact-based pruning, an application of impact-sorting to dynamic query pruning [4]. And again, this is a dynamic version of uniform pruning. Adding to these results, our analysis shows that impact-based methods are good approximate solutions to the proposed model induction problem.

One further question that invites curious eyes is why Dirichlet smoothing worked so well with uniform pruning that it significantly outperformed all the other variations on our Web benchmark WT2G and WT10G. So far the answer is still unclear to us. Here, let us discuss a few possibilities:

- BM25 might be a poor approximation to the probability $p(t|d)$ since the framework presented in this paper was tailored specifically for language models. While this may explain why BM25 was inferior to Dirichlet smoothing in our experiments, it does not tell us why the performance for Jelinek-Mercer smoother and for BM25 were comparable.

- Another possibility is that, since parameter optimization is lacking in our experiment, we might have failed in producing the most competitive result for BM25 and Jelinek-Mercer smoother. If this theory is true, score functions will need task-specific fine-tuning in their further use. But for that to make sense, one needs to point out in what major way the role of a score function in index pruning departs from that in ordinary ad-hoc retrieval. This may point to an interesting direction for future work, but based on the evidence collected so far this claim is difficult to verify.

With the argument given in Section 3 about the convex integer program, one may argue that it is important to prevent depleting any term posting since doing so would take the objective in Equation 10 to minus infinity. In other words, an additional constraint, called "no depletion", shall be added into the index pruning guideline. This is because, even though we do not attempt to solve the convex program, the constraint still needs to be enforced to guarantee that information loss is bounded. In this respect, it is necessary to adopt a top-k preserving strategy (i.e., skip any term posting that has less than k entries), such as the one in TCP, to avoid depleting term postings.

6. CONCLUSION

In this paper, we review the problem of static index pruning from a brand new perspective. Given the appropriate assumptions, we show that this problem can essentially be tackled within a model induction framework, using the principle of minimum cross-entropy. The theory guarantees that the induced model best approximates the full model in terms of probability divergence. We show that static index pruning can be written as a convex integer program. Yet exact inference, though possible as it might be, is generally computationally infeasible for large collections. So we further propose a surrogate model to address the computation issue, and show that uniform pruning is indeed an optimal solution to the formalism. To verify the correctness of our result, we conducted an extensive empirical study. The experiment was redesigned to take two factors, variable control and significance testing, into consideration. This setup has helped us reduce possible experimental bias or error.

Our result confirms that, when paired with the Dirichlet smoother, the performance of uniform pruning is state of the art. Significant improvement over the other methods were observed across diverse retrieval settings. Uniform pruning also exhibits an advantage in robustness with respect to large prune ratio. Specifically, our result on WT10G for short queries suggests that uniform pruning with the Dirichlet smoother retains at least 90% of the baseline performance at 50% prune ratio and 85% at 80% prune ratio. To the best of our knowledge, this is by far the best performance ever reported for static index pruning on the standard benchmark.

This research work has given rise to many technical issues, some have been addressed in Section 5 and some remain unsettled. It shall be interesting to see how uniform pruning responds to other test environments, such as different retrieval engines, corpora, or tasks. Document-length update and pseudo relevance feedback have been two landmark issues that we are ready to explore. Since we did not fine-tune the baseline performance, testing pruning methods against optimized, strong baseline shall provide more insight about this art. Besides all these possibilities, one promising direction is to extend the model induction idea to other type of structured data, such as lexicons or language models. Further investigation into the theory may shed us some light in the role that impact plays in different IR tasks.

7. ACKNOWLEDGMENT

We would like to thank Wei-Yen Day, Ting-Chu Lin, and the anonymous reviewers for their useful comments.

8. REFERENCES

[1] I. S. Altingovde, R. Ozcan, and O. Ulusoy. A practitioner's guide for static index pruning. In M. Boughanem, C. Berrut, J. Mothe, and C. Soule-Dupuy, editors, *Advances in Information Retrieval*, volume 5478 of *Lecture Notes in Computer Science*, chapter 65, pages 675–679. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2009.

[2] I. S. Altingovde, R. Ozcan, and O. Ulusoy. Static index pruning in web search engines: Combining term and document popularities with query views. *ACM Transactions on Information Systems*, 30(1), Mar. 2012.

[3] V. N. Anh, O. de Kretser, and A. Moffat. Vector-space ranking with effective early termination. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 35–42, New York, NY, USA, 2001. ACM.

[4] V. N. Anh and A. Moffat. Pruned query evaluation using pre-computed impacts. In *Proceedings of the*

MAP (t)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	225	221	213	206	198	186	172	149	109
UP-bm25	227	226	223	218	208	194	168	166	143
UP-dir	225	222	213	206	197	190	178	157	117
UP-jm	224	221	211	202	192	180	163	140	103
PRP	227	224	219	209	194	168	148	149	108
IP-u	227	225	220	211	195	173	147	147	108

MAP (td)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	251	246	239	232	219	201	184	162	117
UP-bm25	250	246	235	229	222	202	179	176	158
UP-dir	251	247	239	232	220	210	196	173	133
UP-jm	250	246	238	228	216	194	173	148	110
PRP	253	246	228	212	195	168	148	146	129
IP-u	252	240	230	215	195	173	145	138	125

P10 (t)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	438	434	434	430	428	426	424	384	326
UP-bm25	438	440	436	438	438	412	370	358	294
UP-dir	438	434	432	428	426	434	416	398	344
UP-jm	438	434	432	428	424	416	410	370	318
PRP	436	442	444	452	428	400	340	284	204
IP-u	438	440	440	450	430	406	338	278	190

P10 (td)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	476	468	478	474	468	460	462	432	358
UP-bm25	470	486	482	470	444	420	400	390	324
UP-dir	474	468	472	462	458	450	440	442	386
UP-jm	476	470	470	468	466	442	430	412	336
PRP	486	486	468	470	428	388	350	316	236
IP-u	474	478	474	464	436	400	340	292	226

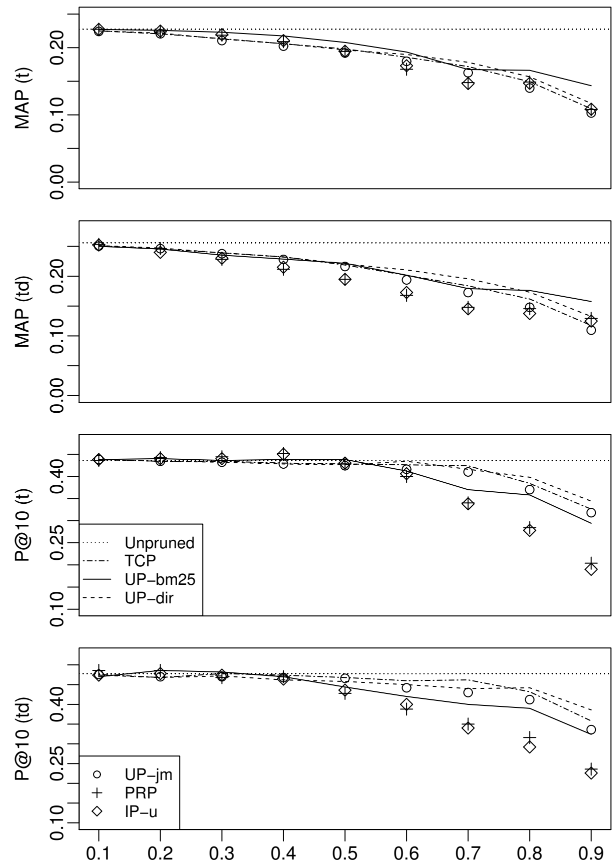

Figure 2: The performance result on Disk 4 & 5, with all measurements rounded to the 3rd decimal place, preceding zero and decimal point removed. The best raw performance at each prune level is underlined. The unpruned baseline has has achieved 0.228/0.256 (t/td) in MAP and 0.436/0.478 (t/td) in P@10.

29th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '06, pages 372–379, New York, NY, USA, 2006. ACM.

[5] R. Blanco and A. Barreiro. Boosting static pruning of inverted files. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 777–778, New York, NY, USA, 2007. ACM.

[6] R. Blanco and A. Barreiro. Static pruning of terms in inverted files. In G. Amati, C. Carpineto, and G. Romano, editors, *Advances in Information Retrieval*, volume 4425 of *Lecture Notes in Computer Science*, chapter 9, pages 64–75. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[7] R. Blanco and A. Barreiro. Probabilistic static pruning of inverted files. *ACM Transactions on Information Systems*, 28(1), Jan. 2010.

[8] S. Büttcher and C. L. A. Clarke. A document-centric approach to static index pruning in text retrieval systems. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, CIKM '06, pages 182–189, New York, NY, USA, 2006. ACM.

[9] D. Carmel, D. Cohen, R. Fagin, E. Farchi, M. Herscovici, Y. S. Maarek, and A. Soffer. Static index pruning for information retrieval systems. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 43–50, New York, NY, USA, 2001. ACM.

[10] R.-C. Chen, C.-J. Lee, C.-M. Tsai, and J. Hsiang. Information preservation in static index pruning. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, CIKM '12, pages 2487–2490, New York, NY, USA, 2012. ACM.

[11] E. S. de Moura, C. F. dos Santos, D. R. Fernandes, A. S. Silva, P. Calado, and M. A. Nascimento. Improving web search efficiency via a locality based static pruning method. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 235–244, New York, NY, USA, 2005. ACM.

[12] D. Hull. Using statistical testing in the evaluation of retrieval experiments. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '93, pages 329–338, New York, NY, USA, 1993. ACM.

MAP (t)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	248	242	233	222	208	195	176	143	090
UP-bm25	249	250	249	236	225	213	188	133	108
UP-dir	247	240	234	226	207	195	198	181	139
UP-jm	247	239	229	220	199	173	143	107	080
PRP	253	249	237	226	196	166	114	092	063
IP-u	250	247	240	225	205	164	126	081	074

MAP (td)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	288	278	258	241	222	206	183	147	096
UP-bm25	288	276	260	237	226	209	193	151	122
UP-dir	290	279	268	258	238	225	224	209	161
UP-jm	289	281	262	245	217	187	149	113	078
PRP	284	262	233	214	189	166	119	093	067
IP-u	284	263	236	216	190	161	124	087	079

P10 (t)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	416	410	408	398	400	380	374	338	268
UP-bm25	414	420	418	386	380	384	340	238	210
UP-dir	414	402	402	404	380	386	392	374	332
UP-jm	414	404	402	402	378	348	336	306	248
PRP	418	418	408	394	362	342	250	126	144
IP-u	410	408	400	390	380	338	268	138	126

P10 (td)	.1	.2	.3	.4	.5	.6	.7	.8	.9
TCP	464	446	428	410	410	388	356	328	302
UP-bm25	464	446	412	370	378	384	326	268	218
UP-dir	460	446	432	436	422	428	422	408	346
UP-jm	460	446	430	420	388	354	340	276	260
PRP	442	434	392	382	340	334	260	142	146
IP-u	450	430	402	392	348	324	258	158	170

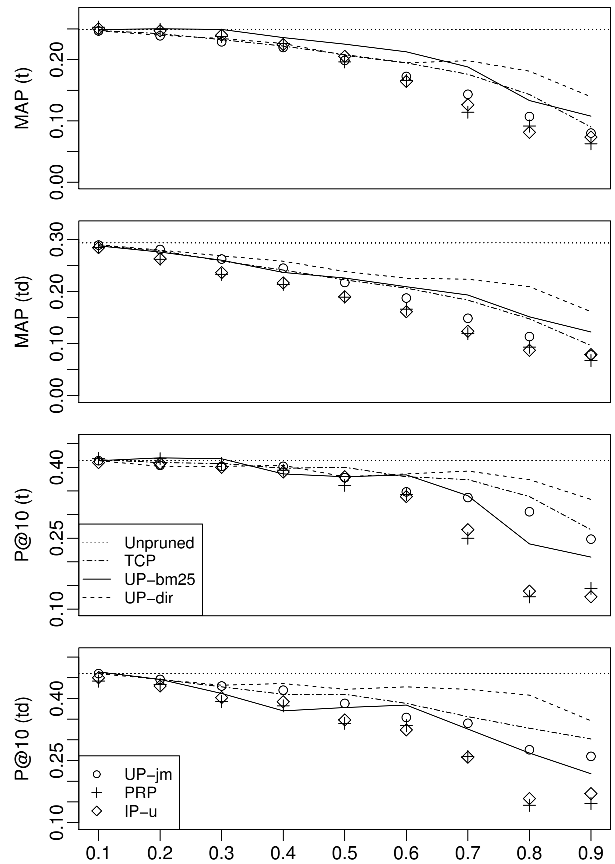

Figure 3: The performance result on WT2G. All measurements were rounded to the 3rd decimal place, and preceding zero and decimal point removed. The best raw performance at each prune level is underlined. The unpruned baseline has has achieved 0.249/0.293 (t/td) in MAP and 0.414/0.460 (t/td) in P@10.

[13] R. J. Hyndman and Y. Fan. Sample quantiles in statistical packages. *The American Statistician*, 50(4):361–365, Nov. 1996.

[14] S. Kullback. *Information Theory and Statistics*. Wiley, New York, 1959.

[15] J. D. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In W. B. Croft, D. J. Harper, D. H. Kraft, J. Zobel, W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *SIGIR*, SIGIR '01, pages 111–119, New York, NY, USA, 2001. ACM.

[16] K. Lange, D. R. Hunter, and I. Yang. Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, 9(1), 2000.

[17] D. C. Montgomery. *Design and analysis of experiments (6th ed.)*. Wiley, 2004.

[18] S. Robertson. The probability ranking principle in IR. In K. S. Jones and P. Willett, editors, *Reading in Information Retrieval*, chapter The probability ranking principle in IR, pages 281–286. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[19] S. Thota and B. Carterette. Within-Document Term-Based index pruning with statistical hypothesis testing. In P. Clough, C. Foley, C. Gurrin, G. Jones, W. Kraaij, H. Lee, and V. Mudoch, editors, *Advances in Information Retrieval*, volume 6611 of *Lecture Notes in Computer Science*, chapter 54, pages 543–554. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[20] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on Information and knowledge management*, CIKM '01, pages 403–410, New York, NY, USA, 2001. ACM.

[21] L. Zheng and I. J. Cox. Entropy-Based static index pruning. In M. Boughanem, C. Berrut, J. Mothe, and C. Soule-Dupuy, editors, *Advances in Information Retrieval*, volume 5478 of *Lecture Notes in Computer Science*, chapter 72, pages 713–718. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2009.