Direct Maximization of Average Precision by Hill-Climbing, with a Comparison to a Maximum Entropy Approach

William Morgan and **Warren Greiff** and **John Henderson**
The MITRE Corporation
202 Burlington Road MS K325
Bedford, MA 01730
{wmorgan, greiff, jhndrsn}@mitre.org

Abstract

We describe an algorithm for choosing term weights to maximize average precision. The algorithm performs successive exhaustive searches through single directions in weight space. It makes use of a novel technique for considering all possible values of average precision that arise in searching for a maximum in a given direction. We apply the algorithm and compare this algorithm to a maximum entropy approach.

1 Introduction

This paper presents an algorithm for searching term weight space by directly hill-climbing on average precision. Given a query and a topic—that is, given a set of terms, and a set of documents, some of which are marked "relevant"—the algorithm chooses weights that maximize the average precision of the document set when sorted by the sum of the weighted terms. We show that this algorithm, when used in the larger context of finding "optimal" queries, performs similar to a maximum entropy approach, which does not climb directly on average precision.

This work is part of a larger research program on the study of optimal queries. Optimal queries, for our purposes, are queries that best distinguish relevant from non-relevant documents for a corpus drawn from some larger (theoretical) population of documents. Although both performance on the training data and generalization ability are components of optimal queries, in this paper we focus only on the former.

2 Motivation

Our initial approach to the study of optimal queries employed a conditional maximum entropy model. This model exhibited some problematic behavior, which motivated the development of the weight search algorithm described here.

The maximum entropy model is used as follows. It is given a set of relevant and non-relevant documents and a vector of terms (the query). For any document, the model predicts the probability of relevance for that document based on the Okapi term frequency (*tf*) scores (Robertson and Walker, 1994) for the query terms within it. Queries are developed by starting with the best possible one-term query and adding individual terms from a candidate set chosen according to a mutual information criterion. As each term is added, the model coefficients are set to maximize the probability of the empirical data (the document set plus relevance judgments), as described in Section 4.

Treating the model coefficients as term weights yields a weighted query. This query produces a retrieval status value (RSV) for each document that is a monotonically increasing function of the probability of relevance, in accord with the probability ranking principle (Robertson, 1977). We can then calculate the average precision of the document set as ordered by these RSVs.

As each additional query term represents another degree of freedom, one would expect model performance to improve at each step. However, we noted that the addition of a new term would occasionally result in a decrease in average precision—despite the fact that the model could have chosen a zero weight for the newly added term. Figure 1 shows an example of this phenomenon for one TREC topic.

This is the result of what might be called "metric divergence". While we use average precision to evaluate the queries, the maximum entropy model maximizes the likelihood of the training data. These two metrics occasionally disagree in their evaluation of particular weight vectors. In particular, maximum entropy modeling may favor increasing the estimation of documents lower in the ranking at the expense of accuracy in the prediction of highly ranked documents. This can increase training data likelihood yet have a detrimental effect on average precision.

The metric divergence problem led us to consider an alternative approach for setting term weights which would

hill-climb on average precision directly. In particular, we were interested in evaluating the results produced by the maximum entropy approach—how much was the maximization of likelihood affecting the ultimate performance as measured by average precision? The algorithm described in the following section was developed to this end.

3 The Weight Search Algorithm

The general behavior of the weight search algorithm is similar to the maximum entropy modeling described in Section 2—given a document corpus and a term vector, it seeks to maximize average precision by choosing a weight vector that orders the documents optimally. Unlike the maximum entropy approach, the weight search algorithm hill-climbs directly on average precision.

The core of the algorithm is an exhaustive search of a single direction in weight space. Although each direction is continuous and unbounded, we show that the search can be performed with a finite amount of computation. This technique arises from a natural geometric interpretation of changes in document ordering and how they affect average precision.

At the top level, the algorithm operates by cycling through different directions in weight space, performing an exhaustive search for a maximum in each direction, until convergence is reached. Although a global maximum is found *in each direction*, the algorithm relies on a greedy assumption of unimodality and, as with the maximum entropy model, is not guaranteed to find a global maximum in the multi-dimensional space.

3.1 Framework

This section formalizes the notion of weight space and what it means to search for maximum average precision within it.

Queries in information retrieval can be treated as vectors of terms $t_1, t_2, \cdots, t_N$. Each term is, as the name suggests, an individual word or phrase that might occur in the document corpus. Every term t_i has a weight λ_i determining its "importance" relative to the other terms of the query. These weights form a weight vector $\boldsymbol{\lambda} = \langle \lambda_1 \ \lambda_2 \ \cdots \ \lambda_N \rangle$. Further, given a document corpus Δ, for each document $d_j \in \Delta$ we have a "value vector" $\boldsymbol{v}_j = \langle v_{j1} \ v_{j2} \ \cdots \ v_{jN} \rangle$, where each "value" $v_{ji} \in \Re$ gives some measure of term t_i within document d_j—typically the frequency of occurrence or a function thereof. In the case of the standard *tf-idf* formula, v_{ji} is the term frequency and λ_i the inverse document frequency.

If the document corpus and set of terms is held fixed, the average precision calculation can be considered a function $f : \Re^N \rightarrow [0, 1]$ mapping $\boldsymbol{\lambda}$ to a single average precision value. Finding the weight vectors in this context is then the familiar problem of finding maxima in an N-dimensional landscape.

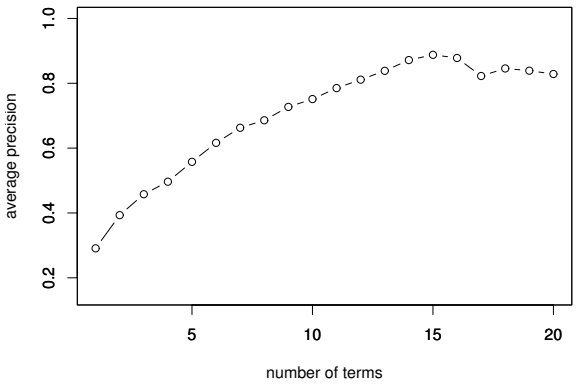

Figure 1: Average precision by query size as generated by the maximum entropy model for TREC topic 307.

3.2 Powell's algorithm

One general approach to this problem of searching a multi-dimensional space is to decompose the problem into a series of iterated searches along single directions within the space. Perhaps the most basic technique, credited to Powell, is simply a round-robin-style iteration along a set of unchanging direction vectors, until convergence is reached (Press et al., 1992, pp. 412-420). This is the approach used in this study.

Formally, the procedure is as follows. You are given a set of direction vectors $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \cdots, \boldsymbol{\omega}_N$ and a starting point $\boldsymbol{\pi}_0$. First move $\boldsymbol{\pi}_0$ to the maximum along $\boldsymbol{\omega}_1$ and call this $\boldsymbol{\pi}_1$, i.e. $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_0 + \mu_1 \boldsymbol{\omega}_1$ for some scalar μ_1. Next move $\boldsymbol{\pi}_1$ to the maximum along $\boldsymbol{\omega}_2$ and call this $\boldsymbol{\pi}_2$, and so on, until the final point $\boldsymbol{\pi}_N$. Finally, replace $\boldsymbol{\pi}_0$ with $\boldsymbol{\pi}_N$ and repeat the entire process, starting again with $\boldsymbol{\omega}_1$. Do this until some convergence criterion is met.

This procedure has no guaranteed rate of convergence, although more sophisticated versions of Powell's algorithm do. In practice this has not been a problem.

3.3 Exhaustively searching a single direction

Powell's algorithm can make use of any one-dimensional search technique. Rather than applying a completely general hill-climbing search, however, in the case where document scores are calculated by a linear equation on the terms, i.e.

$$\sigma_j = \sum_{i=1}^{N} \lambda_i v_{ji} = \boldsymbol{\lambda} \cdot \boldsymbol{v}_j$$

as they are in the *tf-idf* formula, we can exhaustively search in a single direction of the weight space in an efficient manner. This potentially yields better solutions and potentially converges more quickly than a general hill-climbing heuristic.

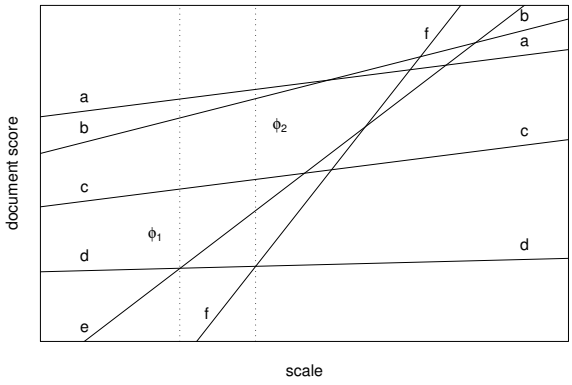

Figure 2: Sample plot of σ versus μ for a given direction.

The insight behind the algorithm is as follows. Given a direction $\boldsymbol{\omega}$ in weight space and a starting point $\boldsymbol{\pi}$, the score of each document is a linear function of the scale μ along $\boldsymbol{\omega}$ from $\boldsymbol{\pi}$:

$$\begin{aligned}\sigma_j &= \boldsymbol{\lambda} \cdot \boldsymbol{\upsilon}_j \\ &= (\boldsymbol{\pi} + \mu\boldsymbol{\omega}) \cdot \boldsymbol{\upsilon}_j \\ &= \boldsymbol{\pi} \cdot \boldsymbol{\upsilon}_j + \mu\,(\boldsymbol{\omega} \cdot \boldsymbol{\upsilon}_j)\,.\end{aligned}$$

i.e. document d_i's score, plotted against μ, is a line with slope $\boldsymbol{\omega} \cdot \boldsymbol{\upsilon}_i$ and y-intercept $\boldsymbol{\pi} \cdot \boldsymbol{\upsilon}_j$.

Consider the graph of lines for all documents, such as the example in Figure 2. Each vertical slice of the graph, at some point ϕ on the x axis, represents the order of the documents when $\mu = \phi$; specifically, the order of the documents is given by the order of the intersections of the lines with the vertical line at $x = \phi$.

Now consider the set of intersections of the document lines. Given two documents d_r and d_s, their intersection, if it exists, lies at point $\iota_{rs} = (\iota^x_{rs}, \iota^y_{rs})$ where

$$\iota^x_{rs} = \frac{\boldsymbol{\pi} \cdot (\boldsymbol{\upsilon}_s - \boldsymbol{\upsilon}_r)}{\boldsymbol{\omega} \cdot (\boldsymbol{\upsilon}_r - \boldsymbol{\upsilon}_s)}, \text{ and}$$

$$\iota^y_{rs} = \boldsymbol{\pi} \cdot \boldsymbol{\upsilon}_r + \iota^x_{rs}\,(\boldsymbol{\omega} \cdot \boldsymbol{\upsilon}_r)$$

(Note that this is undefined if $\boldsymbol{\omega} \cdot \boldsymbol{\upsilon}_r = \boldsymbol{\omega} \cdot \boldsymbol{\upsilon}_s$, i.e., if the document lines are parallel.)

Let Ψ be the set of all such document intersection points for a given direction, document set and term vector. Note that more than two lines may intersect at the same point, and that two intersections may share the same x component while having different y components.

Now consider the set Ψ^x, defined as the projection of Ψ onto the x axis, i.e. $\Psi^x = \{\phi \mid \exists\, \iota \in \Psi \text{ s.t. } \iota^x = \phi\}$. The points in Ψ^x represent precisely those values of μ where two or more documents are tied in score. Therefore, the document ordering changes at and only at these points of intersection; in other words, the points in Ψ^x partition the range of μ into at most $M(M-1)/2+1$ regions, where M is the total number of documents. Within a given region, document ordering is invariant and hence average precision is constant. As we can calculate the boundaries of, and the document ordering and average precision within, each region, we now have a way of finding the maximum across the entire space by evaluating a finite number of regions. Each of the $O(M^2)$ regions requires an $O(M \log M)$ sort, yielding a total computational bound of $O(M^3 \log M)$.

In fact, we can further reduce the computation by exploiting the fact that the change in document ordering between any two regions is known and is typically small. The weight search algorithm functions in this manner. It sorts the documents completely to determine the ordering in the left-most region. Then, it traverses the regions from left to right and updates the document ordering in each, which does not require a sort. Average precision can be incrementally updated based on the document ordering changes. This reduces the computational bound to $O(M^2 \log M)$, the requirement for the initial sort of the $O(M^2)$ intersection points.

4 Experiment Setup

In order to compare the results of the weight search algorithm to those of the maximum entropy model, we employed the same experiment setup. We ran on 15 topics, which were manually selected from the TREC 6, 7, and 8 collections (Voorhees and Harman, 2000), with the objective of creating a representative subset. The document sets were divided into randomly selected training, validation and test "splits", comprising 25%, 25%, and 50%, respectively, of the complete set.

For each query, a set of *candidate* terms was selected based on mutual information between (binary) term occurrence and document relevance. From this set, terms were chosen individually to be included in the query, and coefficients for all terms were calculated using L-BFGS, a quasi-Newton unconstrained optimization algorithm (Zhu et al., 1994).

For experimenting with the weight search algorithm, we investigated queries of length 1 through 20 for each topic, so each topic involved 20 experiments. The first term weight was fixed at 1.0. The single-term queries did not require a weight search, as the weight of a single term does not affect the average precision score. For the remaining 19 experiments for each topic, the direction vectors $\boldsymbol{\omega}$ were chosen such that the algorithm searched a single term weight at a time. For example, a query with

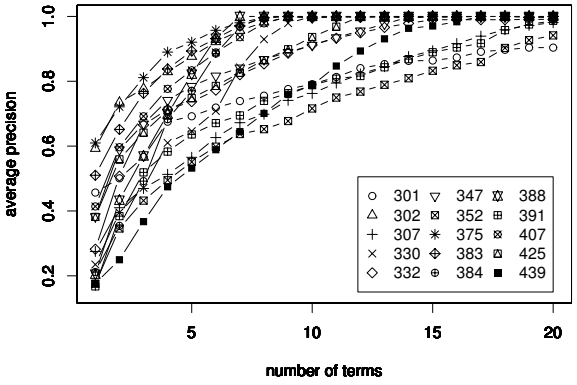

Figure 3: Average precision versus query size for the weight search algorithm. Each line represents a topic.

i terms used the $i-1$ directions

$$\begin{aligned} \boldsymbol{\omega}_{i,1} &= \langle 0\,1\,0\,0\,\cdots\,0\rangle, \\ \boldsymbol{\omega}_{i,2} &= \langle 0\,0\,1\,0\,\cdots\,0\rangle, \\ &\vdots \\ \boldsymbol{\omega}_{i,i-1} &= \langle 0\,0\,0\,0\,\cdots\,1\rangle. \end{aligned}$$

The two-term query for a topic started the search from the point $\boldsymbol{\pi}_{2,0} = \langle 1\ 0\rangle$, and each successive experiment for that topic was initialized with the starting point $\boldsymbol{\pi}_0$ equal to the final point in the previous iteration, concatenated with a 0. The "value vectors" $\boldsymbol{v}_j$ used in all experiments were Okapi *tf* scores.

5 Results

The average precision scores obtained by the maximum entropy and weight search algorithm experiments are listed in Table 1. The "Best AP" and "No. Terms" columns describe the query size at which average precision was best and the score at that point. These columns show that the maximum entropy approach performs just as well as the average precision hill-climber, and in some cases actually performs slightly better. This suggests that the metric divergence as seen in Figure 1 did not prohibit the maximum entropy approach from maximizing average precision in the course of maximizing likelihood.

The "5 term AP" column compares the performance of the algorithms on smaller queries. The weight search algorithm shows a slight advantage over the maximum entropy model on 10 of the 15 topics and equal performance on the others, but definitive conclusions are difficult at this stage.

Figure 3 shows the average precision achieved by the weight search algorithm, for all 20 query sizes and for all 15 topics. Unlike the maximum entropy results, the algorithm is guaranteed to yield monotonically non-decreasing scores.

Topic	5 term AP		Best AP		No. Terms	
	WS	ME	WS	ME	WS	ME
301	0.68	0.67	0.90	0.90	>20	>20
302	0.88	0.86	1.00	1.00	10	10
307	0.57	0.56	0.98	0.89	>20	>20
330	0.65	0.61	1.00	1.00	10	10
332	0.74	0.72	0.99	1.00	**>20**	**18**
347	0.78	0.78	1.00	1.00	**17**	**14**
352	0.55	0.55	0.94	0.93	>20	>20
375	0.92	0.92	1.00	1.00	9	9
383	0.89	0.89	1.00	1.00	9	9
384	0.77	0.73	1.00	1.00	8	8
388	0.82	0.80	1.00	1.00	7	7
391	0.64	0.63	0.99	0.98	>20	>20
407	0.83	0.83	1.00	1.00	9	9
425	0.75	0.73	1.00	1.00	12	12
439	0.53	0.51	1.00	1.00	**17**	**16**

Table 1: Average precision achieved for weight search (WS) and maximum entropy (ME) algorithms.

6 Conclusions

We developed an algorithm for exhaustively searching a continuous and unbounded direction in term weight space in $O(M^2 \log M)$ time. Initial results suggest that the maximum entropy approach performs as well as this algorithm, which hill-climbs directly on average precision, allaying our concerns that the metric divergence exhibited by the maximum entropy approach is a problem for studying optimal queries.

References

William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. 1992. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, second edition.

S. E. Robertson and S. Walker. 1994. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In W. Bruce Croft and C. J. van Rijsbergen, editors, *Proc. 17th SIGIR Conference on Information Retrieval*.

S. E. Robertson. 1977. The probability ranking principle in IR. *Journal of Documentation*, 33:294–304.

E. M. Voorhees and D. K. Harman. 2000. Overview of the eighth Text REtrieval Conference (TREC-8). In E. M. Voorhees and D. K. Harman, editors, *The Eighth Text REtreival Conference (TREC-8)*. NIST Special Publication 500-246.

C. Zhu, R. Byrd, P. Lu, and J. Nocedal. 1994. LBFGS-B: Fortran subroutines for large-scale bound constrained optimization. Technical Report NAM-11, EECS Department, Northwestern University.