The Weaver System for Document Retrieval

Adam Berger and **John Lafferty**
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
<aberger,lafferty>@cs.cmu.edu

Abstract

This paper introduces Weaver, a probabilistic document retrieval system under development at Carnegie Mellon University, and discusses its performance in the TREC-8 *ad hoc* evaluation. We begin by describing the architecture and philosophy of the Weaver system, which represents a departure from traditional approaches to retrieval. The central ingredient is a statistical model of how a user might distill or "translate" a given document into a query. The retrieval-as-translation approach is based on the noisy channel paradigm and statistical language modeling, and has much in common with other recently proposed models [12, 10]. After the initial high-level overview, the bulk of the paper contains a discussion of implementation details and the empirical performance of the Weaver retrieval system.

1 Introduction

This paper introduces the Weaver system for document retrieval, and discusses its performance on the TREC-8 ad hoc retrieval task. Weaver represents a significant departure from the traditional *tfidf*-based retrieval architecture, and its performance in TREC-8 suggests the promise of exploring new probabilistic approaches to retrieval.

The Weaver system is based on the use of statistical language modeling methods and the noisy channel paradigm from communication theory. At its core, however, is a model originally introduced in the context of statistical machine translation [5]. For this reason, we named the system after Warren Weaver, who nearly fifty years ago was the first to propose (albeit decades before computers were up to the task) that statistical techniques might be used to automate the process of translating text from one language into another.

This paper gives a brief overview of the guiding principles behind Weaver, and discusses several implementation details, including a description of how Weaver estimates the parameters of its statistical models from the TREC document collection. The following section contains an abbreviated discussion of the mathematical fundamentals behind Weaver; a more thorough treatment can be found in [1]. Section 3 provides details on the architecture of the Weaver system used in TREC-8. Section 4 contains information on the performance of Weaver in TREC-8, and some preliminary analysis of the results. We conclude in Section 5 by outlining some directions for future work.

2 The Probabilistic Framework

In formulating a query to a retrieval system, we imagine that a user begins with an information need. This information need is then represented as a fragment of an "ideal document"—a portion of the type of document that the user hopes to receive from the system. The user then translates or distills this ideal document fragment into a succinct query, selecting key terms and replacing some terms with related terms.

Summarizing the model of query generation,

1. The user has an information need $\Im$.
2. From this need, he generates an ideal document fragment $\mathbf{d}_\Im$.
3. He selects a set of key terms from $\mathbf{d}_\Im$, and generates a query $\mathbf{q}$ from this set.

One can view this imaginary process of query formulation as a corruption of the ideal document. In this setting, the task of a retrieval system is to find those documents most similar to $\mathbf{d}_\Im$. In other words, retrieval is the task of finding, among the documents comprising the collection, likely preimages of the user's query. Figure 1 depicts this model of retrieval in a block diagram.

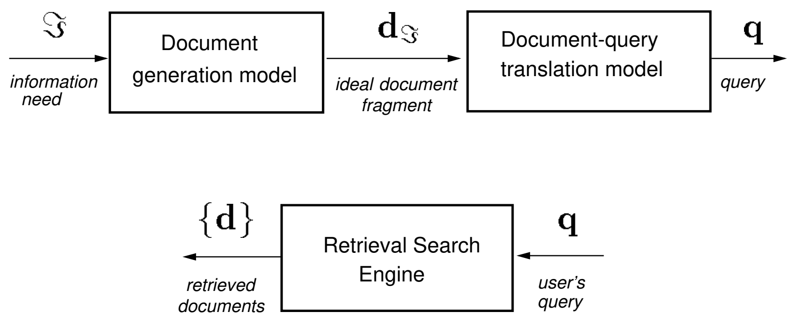

Figure 1. *Model of query generation and retrieval*

We have drawn Figure 1 in a way that suggests an information-theoretic perspective. One can view the information need $\Im$ as a signal that gets corrupted as the user $\mathcal{U}$ distills it into a query $\mathbf{q}$. That is, the query-formulation process represents a noisy channel, corrupting the information need just as a telephone cable corrupts the data transmitted by a modem. Given $\mathbf{q}$ and a model of the channel—how an information need gets corrupted into a query—the retrieval system's task is to identify those documents $\mathbf{d}$ that best satisfy the information need of the user.

More precisely, the retrieval system's task is to find the *a posteriori* most likely documents given the query; that is, those $\mathbf{d}$ for which $p(\mathbf{d} \mid \mathbf{q}, \mathcal{U})$ is highest. By Bayes' law,

$$p(\mathbf{d} \mid \mathbf{q}, \mathcal{U}) = \frac{p(\mathbf{q} \mid \mathbf{d}, \mathcal{U})\, p(\mathbf{d} \mid \mathcal{U})}{p(\mathbf{q} \mid \mathcal{U})}. \tag{1}$$

Since the denominator $p(\mathbf{q} \mid \mathcal{U})$ is fixed for a given query and user, we can ignore it for the purpose of ranking documents, and define the relevance $\rho_{\mathbf{q}}(\mathbf{d})$ of a document to a query as

$$\rho_{\mathbf{q}}(\mathbf{d}) = \underbrace{p(\mathbf{q} \mid \mathbf{d}, \mathcal{U})}_{\text{query-dependent}} \quad \underbrace{p(\mathbf{d} \mid \mathcal{U})}_{\text{query-independent}}. \tag{2}$$

Equation (2) highlights the decomposition of relevance into two terms: first, a query-dependent term measuring the proximity of $\mathbf{d}$ to $\mathbf{q}$, and second, a query-independent or "prior" term, measuring the quality of the document according to the user's general preferences and information needs.

The documents in the TREC-8 evaluation comprise a rather "well-behaved" collection, insofar as very few are completely implausible candidates for any possible query. Therefore, we expect there to be little harm in taking $p(\mathbf{q} \mid \mathcal{U})$ to be uniform. However, we imagine that in retrieval systems using real-world document collections, a non-uniform prior will be crucial for improved performance, and for adapting to the user's needs and interests. At the very least, the document prior can be used to discount short documents, or perhaps documents in a foreign language.

We give a detailed formulation of one $p(\mathbf{q} \mid \mathbf{d})$ model below, but here we will briefly outline the strategy for constructing the model. We start with a corpus of $(\mathbf{d}, \mathbf{q})$ pairs, where each pair consists of a query and a document relevant to the query. Given this data, one can construct a translation model $p(\mathbf{q} \mid \mathbf{d})$, which assigns a probability to the event that $\mathbf{q}$ is a distillation of (a translation of) $\mathbf{d}$.

Retrieval as translation

High-performance document retrieval systems must be sophisticated enough to handle synonomy and polysemy—to know, for instance, that `pontiff` and `pope` are related terms, and that `suit` can refer to clothing or a venue for legal grievance. The field of statistical translation concerns itself with how to mine large text databases to automatically discover such semantic relations. Brown *et al.* [4, 6] showed, for instance, how a system can "learn" to associate French terms with their English translations, given only a collection of bilingual French/English sentences. We shall demonstrate how, in a similar fashion, an IR system can, from a collection of documents, automatically learn which terms are related, and exploit these relations to better rank documents by relevance to a query.

By "translation model," we mean a conditional probability distribution $p(\mathbf{f} \mid \mathbf{e})$ over sequences of source words $\mathbf{f} = \{f_1, \ldots, f_m\}$, given a sequence of target words $\mathbf{e} = \{e_1, \ldots, e_n\}$. In the context of French-to-English translation, the value $p(\mathbf{f} \mid \mathbf{e})$ is the probability that, when presented with the English word sentence $\mathbf{e}$, an expert translator will produce the French sequence $\mathbf{f}$.

Brown *et al.* [4] introduce the idea of an ***alignment*** $\mathcal{A}$ between sequences of words, which captures how subsets of English words conspire to produce each French word. They also introduce the idea of a ***null word***, an artificial word added to position zero of every English sentence, whose purpose is to generate those French words not strongly correlated with any other words in the English string.

Using $\mathcal{A}$, we can decompose $p(\mathbf{f} \mid \mathbf{e})$ as

$$p(\mathbf{f} \mid \mathbf{e}) = \sum_{\mathcal{A}} p(\mathbf{f}, \mathcal{A} \mid \mathbf{e}) = \sum_{\mathcal{A}} p(\mathbf{f} \mid \mathcal{A}, \mathbf{e}) p(\mathcal{A} \mid \mathbf{e}) \tag{3}$$

The IBM family of translation models is predicated on the simplifying assumption that exactly one English word is responsible for a given French word. We can therefore write

$$p(\mathbf{f} \mid \mathcal{A}, \mathbf{e}) = \prod_{i=1}^{m} t(f_i \mid e_{a_i}) \tag{4}$$

Here e_{a_i} is the English word aligned with the ith French word, and $t(f \mid e)$ is a parameter of the model—the probability that the English word e is paired with the French word f in the alignment.

We use the convention that boldface roman letters refer to collections of words such as documents or queries, while italic roman letters refer to individual terms. Thus $p(q \mid \mathbf{d})$ refers to the probability of generating a *single* query word from an entire document $\mathbf{d}$.

If $\mathbf{f}$ contains m words and $\mathbf{e}$ contains $n+1$ words (including the null word), there are $(n+1)^m$ alignments between $\mathbf{e}$ and $\mathbf{f}$. The most basic member of this family of models, Model 1, simplifies matters dramatically by assuming that the translation probability $p(\mathbf{f} \mid \mathbf{e})$ does not depend on the order in which the words appear in the sentences. Thus we can write

$$p(\mathbf{f} \mid \mathbf{e}) = \frac{p(m \mid \mathbf{e})}{(n+1)^m} \sum_{A} \prod_{i=1}^{m} t(f_i \mid e_{a_i})\,. \tag{5}$$

Given a collection of bilingual sentences $\mathcal{C} = \{(\mathbf{f}_1, \mathbf{e}_1), (\mathbf{f}_2, \mathbf{e}_2), (\mathbf{f}_3, \mathbf{e}_3) \ldots$, the likelihood method suggests that one should adjust the parameters of (5) in such a way that the model assigns as high a probability as possible to $\mathcal{C}$. This maximization must be performed, of course, subject to the constraints $\sum_f t(f \mid e) = 1$ for all e. Using Lagrange multipliers,

$$t(f \mid e) = \lambda^{-1} \sum_{\mathcal{A}} p(\mathbf{f}, \mathcal{A} \mid \mathbf{e}) \sum_{j=1}^{m} \delta(f, f_j)\delta(e, e_{a_j}), \tag{6}$$

where δ is the Kronecker delta function.

The parameter $t(f \mid e)$ appears explicitly in the lefthand side of (6), and implicitly in the right. By repeatedly solving this equation for all pairs f, e (in other words, applying the EM algorithm), one eventually reaches a stationary point of the likelihood.

Equation (6) contains a sum over alignments, which is exponential and suggests that the computing the parameters in this way is infeasible. In fact, this is not the case, since

$$\sum_{\mathcal{A}} \prod_{i=1}^{m} t(f_i \mid e_{a_i}) = \prod_{i=1}^{m} \sum_{j=0}^{n} t(f_i \mid e_j) \tag{7}$$

This rearranging means that computing $\sum_{\mathcal{A}} p(\mathbf{f}, \mathcal{A} \mid \mathbf{e})$ requires only $\Theta(mn)$ work, rather than $\Theta(n^m)$.

Brown *et al.* propose a series of increasingly complex and powerful statistical models of translation, the parameters of which are estimated by a bootstrapping procedure. We have described here only that portion of the IBM translation approach that is directly relevant to the retrieval method described below. For further details on statistical machine translation, we refer the reader to two articles [4, 6].

A Model of Document-Query Translation

Suppose that an information analyst is given a news article and asked to quickly generate a list of a few words to serve as a rough summary of the article's topic. As the analyst rapidly skims the story, he encounters a collection of words and phrases. Many of these are rejected as irrelevant, but his eyes rest on certain key terms as he decides how to render them in the summary. For example, when presented with an article about Pope John Paul II's visit to Cuba in 1998, the analyst decides that the words `pontiff` and `vatican` can simply be represented by the word `pope`, and that `cuba`, `castro` and `island` can be collectively referred to as `cuba`.

We consider the simplest of the IBM translation models for the document-to-query mapping. This model produces a query according to the following generative procedure. First we choose a length m for the query, according to the distribution $\psi(m \mid \mathbf{d})$. Then, for each position $j \in [1 \ldots m]$ in the query, we choose a position i in the document from which to generate q_j, and generate the query word by "translating" d_i according to the translation model $t(\cdot \mid d_i)$. We include in position zero of the document an artificial "null word," written `<null>`. The purpose of the null word is to generate spurious or content-free terms in the query (consider, for example, a query $\mathbf{q} =$ `Find all of the documents...`).

Let's now denote the length of the document by $|\,\mathbf{d}\,| = n$. The probability $p(\mathbf{q}\,|\,\mathbf{d})$ is then the sum over all possible alignments, given by

$$p(\mathbf{q}\,|\,\mathbf{d}) = \frac{\psi(m\,|\,\mathbf{d})}{(n+1)^m} \sum_{a_1=0}^{n} \cdots \sum_{a_m=0}^{n} \prod_{j=1}^{m} t(q_j\,|\,d_{a_j}). \tag{8}$$

Just as the most primitive version of IBM's translation model takes no account of the subtler aspects of language translation, including the way word order tends to differ across languages, so our basic IR translation approach is but an impressionistic model of the relation between queries and documents relevant to them. Since IBM called their most basic scheme Model 1, we shall do the same for this rudimentary retrieval model.

A little algebraic manipulation shows that the probability of generating query $\mathbf{q}$ according to Model 1 can be rewritten as

$$p(\mathbf{q}\,|\,\mathbf{d}) \quad = \quad \psi(m\,|\,\mathbf{d}) \prod_{j=1}^{m} \left(\frac{n}{n+1} p(q_j\,|\,\mathbf{d}) + \frac{1}{n+1} t(w\,|\,\texttt{<null>}) \right)$$

where

$$p(q_j\,|\,\mathbf{d}) = \sum_{w} t(q_j\,|\,w)\, l(w\,|\,\mathbf{d})\,,$$

with the *document language model* $l(w\,|\,\mathbf{d})$ given by relative counts. Thus, we see that the query terms are generated using a *mixture model*—the document language model provides the mixing weights for the *translation model*, which has parameters $t(q\,|\,w)$. An alternative view (and terminology) for this model is to describe it as a Hidden Markov Model, where the states correspond to the words in the vocabulary, and the transition probabilities between states are proportional to the word frequencies.

3 Architecture of Weaver

As described in the previous section, Weaver assigns relevance rankings to documents according to a probability $p(q \mid d)$ that a user would distill the document into the query. To rank documents, Weaver doesn't employ a reverse index, but instead visits each document d in the collection and computes (8) for each. This is a tremendously expensive operation, but one can reduce the work somewhat by first performing a *fast match*: eliminate all documents which share no words in common with the query.

Phrases

Although Weaver takes a non-traditional approach to retrieval, it still relies on a standard independence or "bag of words" assumption, ignoring word order within documents and queries. As a modest step in the direction of context-awareness, Weaver does recognize a select set of two-word phrases.

We identified phrases using the statistical measure of mutual information. Within the corpus $\mathcal{C}$ of documents appearing on TREC disks 4 and 5 (excluding the Congressional Record documents), we ranked all pairs of words x, y according to their mutual information, and identified the highest-scoring pairs as phrases. In this context, the mutual information between two words is the reduction in uncertainty about the presence of y that results from knowing whether x was the preceding word.

FINANCI TIME	SAN DIEGO
UNIT STATE	WHITE HOUS
WALL STREET	FISCAL YEAR
PRIME MINIST	GEORG BUSH
SOVIET UNION	PRIVAT SECTOR
HONG KONG	NUCLEAR POWER
SAN JOSE	CHIEF EXECUTIVE
STOCK MARKET	PENSION FUND
SAN FRANCISCO	BILL CLINTON

Table 1: A subset of the 11,644 phrases automatically discovered from the newswire documents in TREC disks 4 and 5 using the mutual information criterion.

Define

$$\begin{aligned} p(y) &= \frac{\text{count}(y)}{\sum_w \text{count(w)}} \\ p(x,y) &= \frac{\text{count}(x,y)}{\sum_{v,w} \text{count(v, w)}} \end{aligned}$$

as the frequency of the word y and the frequency of the bigram x, y, respectively. Furthermore, define

$$\begin{aligned} H(y) &= -p(y)\log p(y) - (1-p(y))\log(1-p(y)) \\ H(y \mid x) &= -p(x,y)\log\frac{p(x,y)}{p(x)} - (1-p(x,y))\log\left(1-\frac{p(x,y)}{p(x)}\right) \end{aligned}$$

as the ***entropy*** of the word y and the bigram x, y respectively. Putting these definitions together, the mutual information score of the bigram x, y is

$$I(x;y) \stackrel{\text{def}}{=} H(y) - H(y \mid x)$$

Intuitively, $I(x;y)$ measures the reduction in uncertainty about whether y will be the next word in a sequence of text, given that x was the previous word. In practice, bigrams like `Hong Kong` tend to exhibit high mutual information. Table 3 lists a selected subset of the 11,644 phrases automatically extracted from the *News* portion of TREC disks 4 and 5.

Vocabulary issues

We elected to use the Porter stemmer to canonicalize English surface forms. This stemmer's deficiencies are well known—it aggressively conflates words, a characteristic which can sometimes be a liability: `policy` and `police`, for instance, are mapped to the same stem. On this first large-scale trial of Weaver, however, we decided to err on the side of a smaller active vocabulary.

We employed the 571-member SMART stopword list to eliminate common words from documents and topics. After applying stemming, pruning stopwords, and adding statistical phrases, we partitioned the collection of active documents into two portions: Federal Register documents and news documents, as summarized in Table 3. We selected the most common 100,000 words from each corpus and built separate models on each corpus.

When searching for relevant documents, we scored Fereral Register documents according to the models trained on this portion of the data and news documents according to their own model. Combining rankings across the two partitions was a simple matter: each model produces a probability estimate $p(\mathbf{q} \mid \mathbf{d})$, and these estimates are comparable across models.

model	**corpus**	**size (in documents)**
FR	*Federal Register (1994)*	55,630
News	*Financial Times (1991-1994)* *Los Angeles Times* *Foreign Broadcast Information Service*	472,525

Table 2: Since the content (and presumably occurrence statistics) of the Federal Register data appeared markedly different from the rest of the TREC-8 collection, we elected to separate out this data and process it independently of the rest.

Synthetic data

The translation model is parametrized in terms of $t(q \mid w)$, the probability that a word w in a document will "generate" the word q in a query for which that document is relevant. Before using these models, one needs to assign a value to each of these parameters. We compute maximum-likelihood values for the model parameters from a collection of queries and documents relevant to those queries. Given such a model and a new query $\mathbf{q}'$, assigning relevance judgments is a matter of computing $p(\mathbf{q}' \mid \mathbf{d})$ for each $\mathbf{d} \in \mathcal{C}$.

One could imagine using relevance judgments from previous TREC evaluations as the training data from which to learn model parameters. However, the number of parameters in the models we use is sufficiently large (the square of the number of recognized words) that just a few hundred topics won't suffice to estimate the parameters accurately. In fact, we know of no publicly-available collection of relevance judgments of suitable heft. Therefore, we synthesize training data as follows: from a TREC document, select words randomly to create a query, and take the document to be relevant to the query. For details on generating synthetic data by sampling, we refer the reader to [1].

For the TREC-8 experiments, we generated, for each of the two partitions of the data, one million synthetic queries of 15 words each.

System configuration

We ran the TREC evaulation on one of six UltraSPARC II 248 Mhz processors belonging to a Sun UltraEnterprise 3000 machine containing 1.5GB of physical memory, running the SunOS 5.5.1. operating system. Other than the fast match described above, we performed no optimization for speed or memory usage. Parameter estimation and document ranking required several days to complete.

Smoothing

For statistical models of this form, *smoothing* or interpolating the parameters away from their maximum likelihood estimates is crucial. We used a simple linear mixture of the background unigram model and the EM-trained translation model:

$$p_\alpha(q \mid \mathbf{d}) \quad = \quad \alpha\, p(q \mid \mathcal{D}) + (1-\alpha)\, p(q \mid \mathbf{d})$$

$$= \alpha\, p(q \mid \mathcal{D}) + (1-\alpha) \sum_{w \,\in\, \mathbf{d}} l(w \mid \mathbf{d})\, t(q \mid w)\,.$$

The weight was empirically set to $\alpha = 0.05$ by optimizing performance on a different dataset: a portion of the 1998 TREC *Spoken Document Retrieval* (SDR) data. Figure 3 shows the behavior of the system on the TREC-7 evaluation as a function of α.

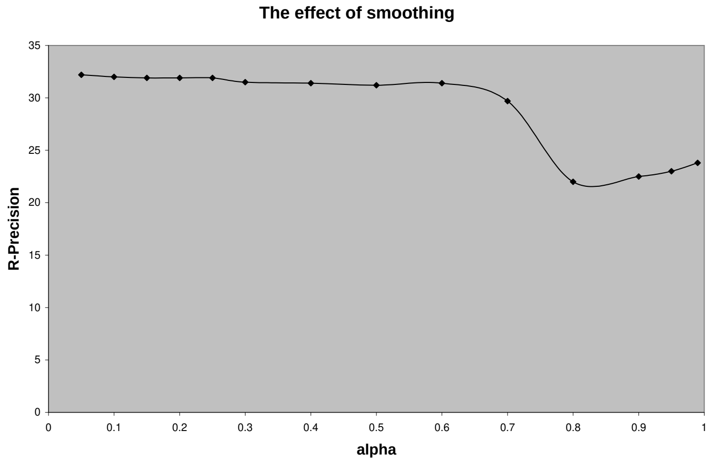

Figure 1: Retrieval performance of Weaver on the TREC-7 ad hoc retrieval task, as a function of α, the weight of the corpus-wide ("back-off") unigram language model in $p(\mathbf{q} \mid \mathbf{d})$. Although we set $\alpha = 0.05$ for the TREC-8 evaluation, it appears that the system is quite insensitive to the exact degree of smoothing, at least within a reasonable range.

4 TREC-8 Performance

Table 4 contains the precision/recall results for the Weaver system within the TREC-8 ad hoc automatic evaluation, as reported by NIST. Figures 2 and 3 display the performance of Weaver relative to all other systems participating in the TREC-8 ad hoc automatic evaluation.

Precision:	
0.00	0.6426
0.10	0.4673
0.20	0.3671
0.30	0.3179
0.40	0.2804
0.50	0.2363
0.60	0.1887
0.70	0.1574
0.80	0.1157
0.90	0.0753
1.00	0.0355
Average :	0.2447

Precision at:	
5 docs:	0.4480
10 docs:	0.4120
15 docs:	0.3600
20 docs:	0.3370
30 docs:	0.3100
100 docs:	0.2134
200 docs:	0.1554
500 docs:	0.0940
1000 docs:	0.0588
R-Precision:	0.2696

Table 3: Performance of the Weaver system in the TREC-8 automatic ad hoc evaluation. The topics consisted of titles and descriptions.

We report our results on the topics consisting of titles and descriptions only, but note in passing that the system performed slightly worse (roughly two percentage points in overall

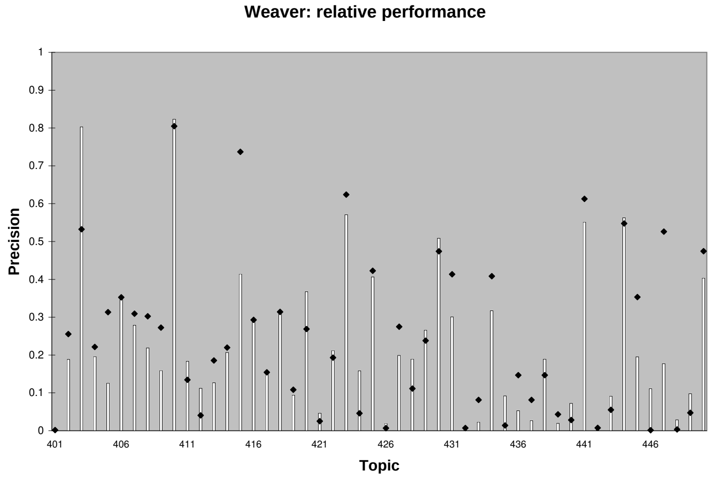

Figure 2: Performance of the Weaver system (represented by points) relative to the median performance of all automatic ad hoc TREC-8 systems (represented by vertical lines).

precision) when provided with the narrative portion as well. This is a somewhat unsuspected result—we observed the narratives to aid performance in internal experiments on earlier TREC datasets—which we plan to explore further.

5 Conclusions

TREC-8 marks the first large-scale evaluation of the retrieval-as-translation paradigm. Along with the language modelling approach put forward by the University of Massachusetts [12] and the Hidden Markov Model system deployed by BBN in TREC-7 [10], Weaver represents a departure from the traditional *tfidf*-based retrieval architecture. Its major asset is a strong theoretical grounding in probability; its major weakness is the computational burden it incurs.

Our immediate future plans will focus on ways to reduce this burden without compromising accuracy. Equipped with a faster retrieval engine, we hope to be able to conduct more experiments to understand how best to extend and improve the Weaver system. We are also investigating ways to incorporate pseudo-feedback into the probabilistic framework.

Acknowledgements

This research was supported in part by NSF KDI grant IIS-9873009, an IBM University Partnership Award, an IBM Cooperative Fellowship, and a grant from Justsystem Corporation. The first author gratefully acknowledges the support of Claritech Corporation throughout the TREC-8 evaluation.

References

[1] A. Berger and J. Lafferty (1999). "Information retrieval as statistical translation," *Proceedings of the ACM SIGIR*, pp. 222-229.

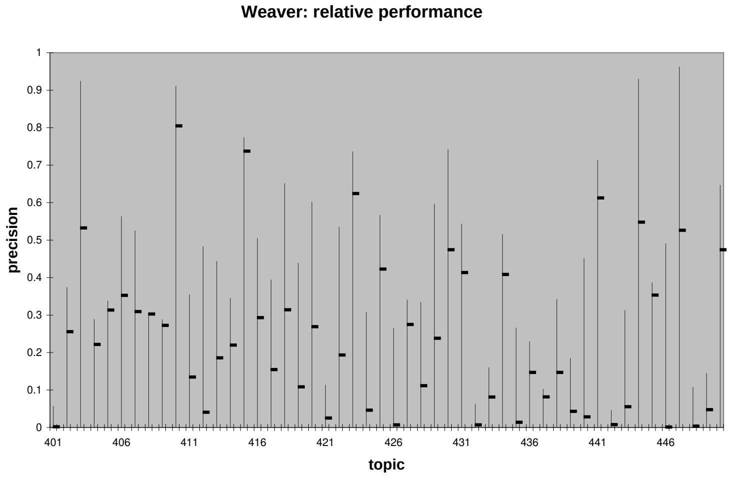

Figure 3: Performance of the Weaver system (represented by hash marks) relative to the best and worst performances among all automatic ad hoc TREC-8 systems.

[2] A. Bookstein and D. Swanson (1974). "Probabilistic models for automatic indexing," *Journal of the American Society for Information Science*, **25**, pp. 312–318.

[3] A. Broder and M. Henzinger (1998). "Information retrieval on the web: Tools and algorithmic issues," Invited tutorial at Foundations of Computer Science (FOCS).

[4] P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer, and P. Roossin (1990). "A statistical approach to machine translation," *Computational Linguistics*, **16**(2), pp. 79–85.

[5] A. Berger, P. Brown, S. Della Pietra, V. Della Pietra, J. Gillett, J. Lafferty, H. Printz and L. Ures (1994) "The Candide system for machine translation" *Proceedings of the ARPA Human Language Technology Workshop*, Plainsborough, New Jersey.

[6] P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer (1993). "The mathematics of statistical machine translation: Parameter estimation," *Computational Linguistics*, **19**(2), pp. 263–311.

[7] W.B. Croft and D.J. Harper (1979). "Using probabilistic models of document retrieval without relevance information," *Journal of Documentation*, **35**, pp. 285–295.

[8] A. Dempster, N. Laird, and D. Rubin (1977). "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society,* **39**(B), pp. 1–38.

[9] W. Gale and K. Church (1991). "Identifying word correspondences in parallel texts," in *Fourth DARPA Workshop on Speech and Natural Language*, Morgan Kaufmann Publishers, pp. 152–157.

[10] D. Miller, T. Leek and R. Schwartz (1999). "BBN at TREC-7: Using Hidden Markov Models for Information Retrieval," In *Proceedings of the Text REtrieval Conference (TREC-7)*, Gaithersburg, Maryland.

[11] J. Ponte (1998). *A language modeling approach to information retrieval.* Ph.D. thesis, University of Massachusetts at Amherst.

[12] J. Ponte and W. B. Croft (1998). "A language modeling approach to information retrieval," *Proceedings of the ACM SIGIR*, pp. 275–281.

[13] S.E. Robertson and K. Sparck Jones (1976). "Relevance weighting of search terms," *Journal of the American Society for Information Science*, **27**, pp. 129–146.

[14] S. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau (1992). "Okapi at TREC," In *Proceedings of the Text REtrieval Conference (TREC-1)*, Gaithersburg, Maryland.

[15] G. Salton and C. Buckley (1988). "Term-weighting approaches in automatic text retrieval," *Information Processing and Management*, **24**, pp. 513–523.

[16] H. Turtle and W.B. Croft (1991). "Efficient probabilistic inference for text retrieval," *Proceedings of RIAO 3.*

[17] W. Weaver (1955). "Translation (1949)," In *Machine Translation of Languages*, MIT Press.